ESST Master Thesis 2008

International Master Programme ESST
European Studies of Society, Science, and Technology

Graduate Program in the History and Philosophy of Science and Technology
National and Kapodistrian University of Athens and
National Technical University of Athens

October 2008

THEODORE LEKKAS

CONSIDERATIONS ON SOFTWARE PIRACY

and its role in the formation of the Greek software industry

CONSIDERATIONS ON SOFTWARE PIRACY

and its role in the formation of the Greek software industry

CONSIDERATIONS ON SOFTWARE PIRACY

and its role in the development of the software industry in Greece

ESST Master Thesis 2008
Typeset in *Times New Roman*
References listed in *APA Style*
20.758 words including footnotes and references

Written and submitted by:
Theodore Lekkas[1] [tlekkas@phs.uoa.gr] |
Personal Web Page: [http://www.phs.uoa.gr/ht/lekkas.html]
Graduate Program in the History and Philosophy of Science and Technology |
National and Kapodistrian University of Athens and National Technical University of Athens
Athens, August 2008

Specialization: Information Technology: Historical to Policy Considerations[2]
1st and 2nd Semester Institution: Graduate Program in the History and Philosophy of Science and Technology, National and Kapodistrian University of Athens and National Technical University of Athens
Supervisor: Aristotle Tympas [tympas@phs.uoa.gr]

Contact Details:
Theodore Lekkas
Kosti Palama 210, 184 52,
Nikaia, Greece

[1] Theodore Lekkas is an IT Technical Writer and Documentation Specialist at Greek software companies. His research interests include the history of software and the software industry in Greece. He is a member of the Society for the History of Technology (SHOT), the History of Science Society (HSS) and the IT History Society (ITHS).
[2] For further details, http://www.hpst.phs.uoa.gr/ESST/GPHPST-ESST_2007-2008_Specialization_INFORMATION______TECHNOLOGY-HISTORICAL_TO_POLICY_CONSIDERATIONS.doc

Synopsis

The aim of this thesis is to put software and software piracy into a historical context. Software piracy is connected to various aspects of the software production and use· ethical, legal, and economic. Software piracy is usually defined as the making and using of unauthorized copies of a software program. Seeking to elaborate on aspects of the software piracy phenomenon, in Part 1 of this thesis I introduce to the historical context of the emergence of software as a distinct technology in the mid-1940s, its development from machine codes to high level languages, and its various versions, from operating system to application software. In Part 2, I discuss some of the key features of software, in a manner that points to some of software's peculiar characteristics, which seem to differentiate it from other technological artifacts. As my argument goes, these characteristics -which include, most notably, the historical difficulty to set up a successful software production taylorist-type assembly line- may explain much of the software piracy phenomenon. In Part 3, I elaborate on this argument by focusing on a case-study, namely the role of software piracy in the history of the Greek software industry. The high piracy rates recorded in the Greek case make this case-study all the more suggestive. I will make specific references to the historical context of the development of software in Greece, to the role of the users in this country, and to relevant processes of localization-domestication of the software technology in Greece. My thesis concludes by suggesting that defining and identifying software piracy, and, moreover, interpreting its influence in the development of software and the software industry, is a much more complex affair than it is usually assumed. Especially when experiences from countries beyond the US and the north-west of Europe are also taken into account.

Keywords: software, history of software, computing technology, software crisis, software piracy, BSA, Greek software industry.

Table of Contents

Table of Figures

Introduction

Computing technology mediates now in the daily life of, almost, everybody. The rapid advance of computing technology has provided an excellent opportunity for the study of technological invention, innovation, transfer, diffusion and use. Computers are not just circuits, transistors, memories and wires. The functions of the computers are performed and controlled through what is now called 'software'· something that has no physical existence--at least not nearly as apparent as the physical existence of 'hardware'. Scholars have pointed out that it is software as embedded in sociotechnical systems that turns "computer" into "applications"[3].

This point was not so obvious for the earlier historiography of computing, which was restricted to the study of machines and engineers. Traditionally, the historiography of computing technology tended to focus on eminent figures, such as Charles Babbage. It also tended to employ an internalist-esoteric technical perspective, usually in reference to the hardware development of early computers, like the ENIAC and EDVAC. The emphasis was placed on the development of individual machines and on documenting contributions of early inventors.

In recent years, historians of computing technology have started to focus, also, on the commercial and scientific uses of computers, on the institutional and economic contexts of their development and use, on processes of diffusion, adaptation, domestication and localization of computing technology. There is now available a growing number of studies on aspects of the social history of computing.

[3] Thomas Landauer argues that "it is a mistake to view the computer as a single technology, more appropriately, each major application is a new technology harnessing information processing capability, much as the electric motor, the locomotive, and the jet plane all harnessed energy-transforming capability" (Landauer, 1995: 104).

The history of software is, relatively speaking, a less developed area in the history of computing technology. It has recently benefited from a ‘computers in use’ perspective (Cortada 1996; Hafner & Lyon 1996; Rochlin 1997; Campbell-Kelly & Aspray 2004). Under this perspective, there are now studies available on the social understanding the computer (Cortada 1993[4]) and computer software (Campbell-Kelly 2003) as a socially situated market phenomenon. The delayed attention to the history of the software industry has been explained by referencing to the limited size of this industry (during an initial period of computing) (Campbell-Kelly 1995). As the economic importance software product became undisputed, studies of the software industry started to appear, especially in the U.S[5].

When the electronic computer was first developed, only a few (if any) could anticipate that something like software would turn to be important and indispensible. The fact is that, by the 1990s, software overshadowed the hardware. The success of home computing at mid 1980s and the 1990s has created a gigantic market for personal software in particular. During this period, a great amount of software has been purchased by businesses, educational institutions and governments. At the same time, software has been increasingly purchased by individuals for use at home. As the demand for software has increased, a greater number of individuals have been copying software for their personal use.

The unauthorized production of exact duplicates has been relatively easy in regards to software programs (unlike the duplication of computer hardware, which is comparatively much more difficult). It demands low-level computer expertise, a computer set that is rather affordable,

[4] Cortada is seeking to understand the prehistory of the computer as a market phenomenon by examining what he refers to as the "proto-information processing industry", the office equipment industry that arose and flourished between 1865 and 1956.
[5] Actually, a number of them made specific recommendations on the intellectual property legislation and on efforts to combat unauthorized copying.

and, a copy of the program to be duplicated. It follows that the role of the computer user is very important in this case. Software products appear to be extremely vulnerable to unauthorized copying, because a second copy of the product contains all the information that is required by users in order make unauthorized copies. Such copies can be made using standard (and cheap) consumer equipment. In the past three decades, software producers have devoted significant resources in the effort to fight all forms of unauthorized copying. To be more effective in this effort, software manufacturers have come together in instituting trade organizations, which include the Software Publishers Association (SPA), the Association of Data Processing Service Organizations (ADAPSO), and, mainly, the Business Software Alliance (BSA).

My overview of the available historiography of the history of software has suggested to me that software piracy is not an issue that has been addressed by historians of software. Software piracy is a topic that has been studied only by scholars of business and management studies, and, by patent or law studies scholars. As a result, the approaches of software piracy have been limited to the study of copyright acts and provisions for their enforcement, to copyright laws, to the loss of revenues of the software industry and to the effect of this alleged loss on the software market, and, to the ethical dimensions of this practice. In regards to this last aspect of software piracy, specific criminological theories have been proposed (Willison and Siponen 2008). The available literature also includes studies of policies that could curb unauthorized copying. These policies range from lowering of prices to stricter law enforcements by governments and digital rights management authorities. The majority of the available studies assume the negative impact of software piracy on the computer industry. However, several economists have argued about the potential of a positive influence of software piracy in the size of publishers' profits due to the development of a large installed base (a large base of software uses). According to their arguments, software piracy initially produces a wide channel for the rapid flow and distribution of software, which may lead to an increase in sales, because it exposes to software individuals who may not have been aware of the product through normal distribution channels. In sum, the existing literature on software piracy discusses the phenomenon only from perspectives appropriate to managerial, governmental and other institutional initiatives that seek to reduce the 'problem' (on how political, economic and social factors direct technological developments, see Ceruzzi 1998).

There exist, also, empirical studies, that seek to record the varying piracy rates across countries and regions (Gopal and Sanders 1998; 2000) (Husted 2000). Software piracy has yet to attract the interest of historians of technology, including specialists in the history of software. I am also unaware of any scholarly study on software piracy about the Greek case, historical or other. To my knowledge, no rigorous Science, Technology, and Society (STS) theoretical framework has been used to explain this important phenomenon. In a historiographical paper, Aristotle Tympas (2006) has gone so far as to suggest that "for many actors of the early Greek PC computing scene (late 1980s) piracy was a functional prerequisite for the development of computing in the country".

My own research in the history of computing technology and software in Greece has suggested that "software piracy" activities go beyond the sphere of business and economics, because they reach into the political and the cultural sphere. To retrieve and interpret this reach, we need a historiographical emphasis on the 'technology-in-use' that is on the study of technology that includes the long-run end user (Edgerton 1999, 2006, Oudshoorn and Pinch 2003). As I argue over the course of this thesis, studying the software piracy in Greece may reveal various important tensions, including tensions between software producers and end users, between government and the software industry, and, between Greek and foreign (mostly US and European) software vendors. In regards to this last tension, some Greek software manufacturers were trying to "catch up" with the rest (for an account of the mass-market software and the IBM-clone systems, see Martin Campbell-Kelly 2003).

In this thesis, I introduce to software piracy as a factor that has affected the development of computing technology in Greece. In the context of doing so, I respond to questions like the following: Has unauthorized software copying affected the development of the Greek software industry? How? Has unauthorized copying facilitated or slowed down processes of software adaptation to local needs? What has the role of end users been? How end users argued over the software piracy issue? The central suggestion of this thesis is that the study of software piracy in a specific social context (e.g. in a country, in this case Greece) may be proven crucial in our attempts to understand the structure and development of computing technology and its relationship to society.

My primary sources are journalistic articles from the two longest running Greek home computing periodicals, *Ram* and *Computer for All*. These periodicals have been key intermediaries between the end user and the software producer (Guerreiro-Wilson et al. 2004). In addition, I have used press releases from organizations involved in the software industry; including organizations set up to address the issue of software piracy, including press releases by BSA Hellas. Finally, I have relied on surveys that reported on the state of the Greek IT industry in general and software piracy in particular.

I start by offering a synthetic account on how “software” has emerged, and, on its relationship to computer hardware. This is because the history of software cannot be treated separately from the history of computing, because no software development was independent from corresponding development in hardware. In addition, I will provide with an overview of the initial steps in establishing programming concepts. These steps have offered the medium for producing a more reliable and efficient software products from the 1970s onwards. My synthesis is limited by the fact that a good part of the history of software has focused on programming languages and systems software (Cortada 1990).

I have, however, benefited from the recent emergence of a historical literature on the development of the software industry (Campbell-Kelly 2003; Mowery 1996). Based on this literature, I move on to outline some of the central questions considering the development of software and its sale as a commercial product. This is followed by a section in which I introduce to efforts by Greek firms to “catch up” with Western European levels of information technology. These efforts resulted in the emergence of a competing Greek software industry, which even included exporters of software products. My historical outline of the emergence of the Greek software industry highlights a number of conflicts. Understanding these conflicts can helps us to elaborate on the software piracy issue. Naturally, I am mostly interested in the tension between users, who sought to easily copy software for free, and software companies, who wanted to make profits from their products.

In regards to the piracy issue, I introduce to the piracy rates in Greece, as recorded in surveys undertaken by the Business Software Alliance (BSA) and the market research group International Data Corporation (IDC). For some, these surveys indicated that even relatively low piracy rates can amount to huge industry losses. I also raise the following questions: Are these figures offered by these surveys reliable? Do they capture the software piracy phenomenon in Greece? How can these software piracy rates be explained? Has software piracy been only a problem or has it also been a prerequisite of the development of software in Greece.

The Historical Context

Computers, Computing and Machine Codes

It was back in 1936 when the British mathematician Alan M. Turing's (1912–1954) research into mathematics at Cambridge University led to his famous paper *On Computable numbers, with an application to the Entscheidungsproblem*. In this paper, Turing argued that, in theory, a machine could be constructed to prove that a mathematical theorem was true. He also argued that it would be possible to create a machine, now known as the 'Universal Turing Machine', which could solve all mathematical problems (for a definition see Davis 1957; for the developments in mathematic and logic that led to the notion of the 'universal machine' see Davis 2000). Attempting to resolve a long-standing debate over whether any method could prove or disprove all mathematical statements, Turing invoked the notion of a "universal machine" that could be given instructions to perform a variety of tasks. This idea was clearly stated also in his ACE Report [Proposal for Development in the Mathematics Division of an Automatic Computing Engine (ACE)] of 1945 (Peláez 1999). "This special property of digital computers, that they can mimic any discrete-state machine, is described by saying that they are universal machines," Turing (1936) stated characteristically.

Turing spoke of a "machine" only abstractly, as a sequence of steps to be executed (Peláez 1999). But his realization that the data fed into a system also could function as its directions cleared the way to the development of software. By realizing that a machine could be adaptable enough to execute a range of different tasks when supplied with the appropriate program rather than being constructed to just solve one problem, Turing's[6] was one of the first explicit formulation of modern computing (Copeland 2004). At this point, we should mention that historians of computing consider also other aspects as significant for this breakthrough, like the use of electronic components or the binary number representation (Hodges 2006). A binary number is a number written in the base two. This means that instead of using the digits zero through nine, like in base ten, only the digits zero and one are used. Each place is then equivalent to a power of two, for example the number five in binary would be 101. The binary number representation was crucial in performing logical operations by representing zero as "false" and one as representing "true". This was the concept behind the 'first generation of codes' (machine language or machine code) used to program a computer. This sequence of 0s and 1s was interpreted by the first electrical computers as instructions. These machine codes were entirely numeric, resulting in programming that was very tedious and faced serious problems, like poor readability and poor modifiability.

As Copeland points out, the first digital computing machines came to existence only a decade after Turing wrote 'On computable numbers'. At the time, the term 'computer' still meant to many a person performing a computing task (Ceruzzi 2003)(Copeland 2004)[7]. While Turing was clear about the universal machine as being the basis for the digital computer, he did not identify the common storage of data and instructions as its crucial feature. In 1945, von Neumann became the first to outline a 'practical version' of the universal machine. Von Neumann's interest in computers differed from that of his colleagues because of his perception of computers as

[6] "I believe that in about 50 years time it will be possible to program computers…to make them play the imitation game so well that an average interrogator will not have more than 70% chance of making the right identification after five minutes of questioning" (Turing, 1950).
[7] The question about the invention of the digital computer is a very complex one. A popular answer is Charles Babbage in 1834. However, the historiography of computing focuses on the intention of the general purpose digital computer, arguing that it was Eckert and Maunchly, with the construction of the ENIAC machine.

usable in applied mathematics for specific problems, rather than as usable, only, to the development of tables[8] (Aspray 1990).

War Needs and Stored 'Programs'

Before the development of the electronic computer, computing technology was dominated by electro-mechanical machines designed primarily for office automation. Punched-card machines offer a typical example. There were machines used for punching, copying, and sorting data cards. It wasn't until the 1940s that electronic devices that we recognize as being similar that modern computers began to appear. In the 1940s, the war needs for computations in firing and ballistic tables lead to the development of computers for programming the desired operations, and, for entering the numbers used in the calculations. These were the Howard Aiken's Harvard Mark I (IBM Automatic Sequence Controlled Calculator [ASCC])[9] and the ENIAC (Electronic Numerical Integrator and Computer)[10]. The military needs,[11] along with a strong consumer market (IBM's concentration on a commercial rather than a military market and its domination on the latter are crucial here), provided the climate for the eventual flourishing of computing technology (Ceruzzi 2003).

[8] By tables, we mean two things: those that present the results of a mathematical calculation, and those that present data. The technology of punched cards helped to process large amounts of information and, as technology, was the predecessor of the calculating machines, and the electronic computer (for an overview on the development of the tables and their relation to the computing technology see M. Campbell-Kelly et al., 2003)

[9] Mark I is thought to be the first programmable computer (but not a stored program computer), because it used punched paper tape (consisted of a row of holes) for programming, and vacuum tubes and relays to calculate problems (Ceruzzi 2003). Ceruzzi argues that "thus began the practice of computer programming in the United States" (Ceruzzi 2003: 82)

[10] The ENIAC was designed during WWII at the University of Pennsylvania to compute ballistics tables for the Army. However, ENIAC, a room-size machine with 18,000 vacuum tubes, was finished too late to help with the war effort.

[11] The U.S. Navy was that ordered Grace Murray Hopper, a professor of mathematics at Vassar College, to assist Howard Aiken with programming the Mark I.

In the context of WWII, the "first generation" computers were built using vacuum tubes to perform their calculations. In regards to software, the acknowledgement of the need for an internally stored program was an important step[12]. In 1945, John von Neumann wrote a paper entitled ‘First Draft of a Report on the EDVAC’, in which he described how a binary program could be electronically stored in a computer (Neumann and Williams 1993; Peláez 1999). This program would enable the computer to alter the operations to be performed, depending upon the results of previous steps. This concept introduced the basic elements of the ‘stored program’ idea. Only years later this concept was also introduced in the industry.

A similar approached characterized the work of Konrad Zuse. After he has completed the first fully functional program-controlled electromechanical digital computer, the Z3, in 1941, he began to work on the ‘Plankalkul’ (plan calculus), the first algorithmic programming language, aiming to create the theoretical preconditions for the formulation of problems of a general nature (Giloi 1997). Konrad Zuse sought to offer a concept that captured how computational machines can be programmed in a powerful way. As a result, he wrote many sample programs. He focused on the types of logical problems that had to be solved. His ideas were never implemented (Ceruzzi 2003).

The ‘stored program’ concept[13] breakthrough is credited to a June 21, 1948 event, which occurred at the University of Manchester during using a test computer called Small-Scale Experimental Machine, known as SSEM or ‘The Baby’. The EDVAC (Electronic Discrete

[12] Eloina Peláez argues that the concept of the stored-program computer can be found on the combination of two different traditions: the engineering tradition (hardware) and the mathematical-logical tradition (software) (Peláez 1999).

[13] By ‘stored program’, is meant a sequence of machine language instructions stored as binary values in the memory of the computer that executes it. These instructions can be loaded from the tape more than one time, if needed, by the computer. Thus, the programmer can ‘write’ the most frequently used tasks, making the whole operating procedure much more efficient.

Variable Automatic Computer) was the first computer that utilized the idea of an electronically-stored program and the first to use to use a magnetic tape that tried to replace the inflexible plug-in system of ENIAC's with something less laborious. To solve this issue, Eckert developed a storage medium capable of holding both instructions and data[14].

This was a critical point in the development of both computers and software, as preceding computers had to be re-programmed by re-wiring whereas EDVAC could have new programs loaded of the tape. Some have, however, argued that the idea of 'programming' the computer can be found in ENIAC's creation, in the use of the 'set-up' operations, which meant the plugging and unplugging a batch of cables. This 'set-up' procedure was extremely time consuming. It is estimated that, for the calculation of a single ballistic table, two days of human labor practice were required (Peláez 1999). The plugging and unplugging practice also confined the purpose of the machine to one specific problem. Any other problem would require a different reconfiguration and a new set up of instructions. It seems safe to consider that the concept of storing both instructions and data gave birth to the establishment of 'programming' (and later software). The stored program concept attempted to eliminate the delays involved in setting up a machine by enabling the machine to modify the program in a way that was not possible where the instructions were contained on punched cards or a paper tape, as before.

[14] Its creators, Eckert and Mauchly, were clear in their description: "An important feature of this device was that operating instructions and function tables were stored exactly in the same sort of memory device as that used for numbers" (Ceruzzi 2003). Magnetic tape was introduced as a data-storage medium in 1951, when it was used in the auxiliary memory of UNIVAC I, the first digital computer produced for commercial use. For about the next 10 years nearly all computers employed magnetic tape storage units.

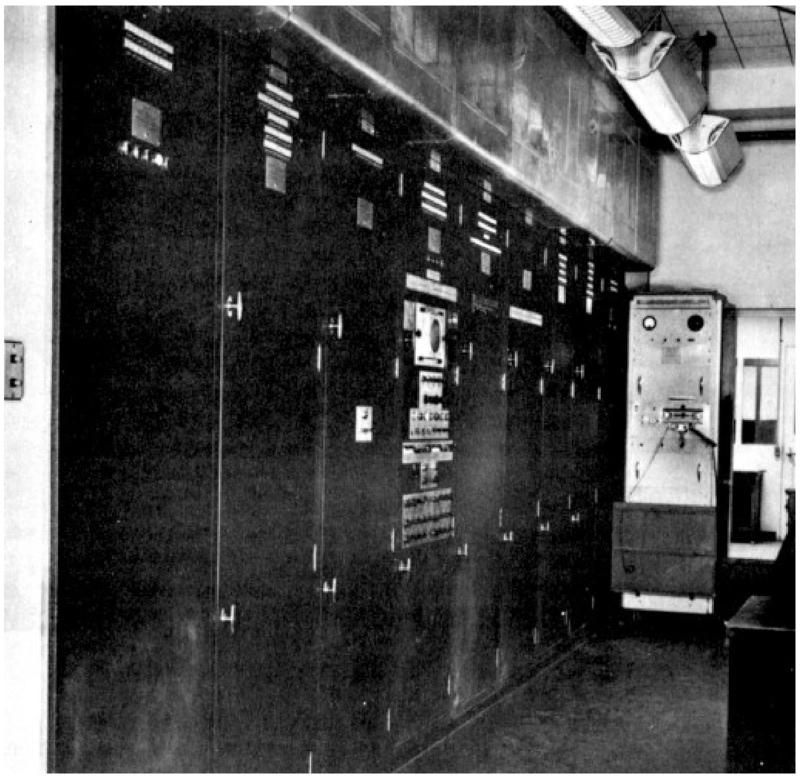


Figure 1: EDVAC

By the latter years of World War II, the limited speed and memory capacity of these computers forced programmers to exploit the connectivity among their fundamental working units and thus to deploy the assembly language. The assembly language had a very close relationship to the machine's central processing unit (CPU) operations at the bit and byte level. It seemed to provide a more human-readable method of writing programs than writing them in binary bit patterns. More specifically, assembly language was consisted of a set of symbolic instructions that each told the machine's processor to perform one simple operation, such as adding two numbers. An assembler program translated these instructions into a binary form, the 'machine language'. Machine language could reside in a random-access memory (RAM) for speedy access by the central processor in performing certain tasks. But, assembly language did not provide prewritten functions that could perform common or repeated tasks. Accordingly, it proved to be very effort-demanding and error-prone, because it was written by hand (Backus, 1978). The assembly language is usually called the 'second generation of code'.

At this point of the software development, the first explicit thoughts for a business exploitation of all this effort emerged. It was Eckert and Mauchly who tried to build a machine (the BINAC) that could store information on magnetic tape rather than on punched cards. In the late 1940s, John Mauchly suggested the 'Short Order Code', which was implemented by William Schmitt for the BINAC computer and for UNIVAC in 1950. This program provided the problem-solution procedure with greater flexibility and worked very well as the problems were small in scale (Norberg 2003). The motivation behind this effort was the need of speeding up the programming of mathematical problems for BINAC and the belief that this would be solved if a general program were written which would allow BINAC to interpret algebraic equations directly without the need to write machine code. This was the genesis of what came to be known as the UNIVAC Short Code, though the first efforts were directed at BINAC.

As Norberg has pointed out, historians of the computing technology focus mainly on what they characterize as the "major revolutionary development of the mid-1950s: higher-level languages such as FORTRAN and COBOL". They pay little or no attention to the previous decades as "little or no software development went on" (Norberg, 2003). Indeed, back then the notion of computers sharing program "software" (a term not yet invented) was not an issue—if you had a computer, you wrote programs specifically for it and no other machine used them. For computing machines like the ENIAC programming was not the primary issue and was almost always neglected (Bauer 1973). Jennifer Light has recently shown that software was actually important in the context of ENIAC. It was produced by women-computors, who worked behind the scene during public presentations of the ENIAC (Light 1999).

From Code to Languages

The third generation of code, called high level language or HLL begun to be developed in the 1950s. High-level languages were a significant improvement over machine and assembly languages because they allowed the programmers to use instruction that resemble the English language more closely (Ceruzzi 2003). FORTRAN (FORmula TRANslating) was the first higher level programming language. It was in place by 1954. It was used in IBM 704, which had index registers and floating point hardware. The IBM Preliminary Report of 1954 claimed that FORTRAN would virtually eliminate coding and debugging[15]. The success of the FORTRAN can be ascribed to two factors: First, the fact that machine code, developed by FORTRAN was as efficient as the code written by the programmers, and, second, the fact that it was accompanied by the very successful IBM's Model 704, on which FORTRAN was running (Ceruzzi 2003).

FORTRAN inaugurated a new era in the development of high-level languages, as users were given the opportunity to focusing to on solution of specific problems, not on the internal machine code and the system's complexity. A few years later, in the late 1950s, the U.S. Department of Defence expressed the desire to develop a common business language, which should meet specific requirements. This initiated a first attempt at standardization of programming languages. The goal was to design of a programming language that ought to look like simple English so that

[15] John Backus, leader of the group which developed FORTRAN, argues that the development of FORTRAN was influenced by what he calls 'the economics of programming'. By that, he means the high cost of programming which was equal, in most cases, to the cost of the machine itself (Backus 1978). This issue is high connected to the 'software crisis' problem. Backus himself argues that he proposed the FORTRAN project attempting to address the economic factor issue.

it would be easy to use, even if this means that it would be less powerful. This could broaden the base of computer users who would not have to deal with contemporary compiler problems. This resulted in the development of COBOL, which involved several major computer manufacturers, like IBM and Honeywell (through their contracts with the U.S. Government). COBOL turned to be significant milestone because it was easy to read and understand by persons, e.g. managers, who could not write code (Ceruzzi 2003). COBOL also proved to be a programming language well suited for creating business applications. Many other languages were developed during this period, like ALGOL (alongside an effort for a language independent of any specific hardware), JOVIAL (which was developed in connection with the SAGE air-defence system), LISP (List Processing) and SNOBOL (StriNg-Oriented symBOlic Language).

As Bauer has stressed, software was grossly underestimated during this period (Bauer 1973). By the 1960s there was a rapid increase in computing machines sales. The demand of programs to be delivered with them was correspondingly increased. In the 1960s there was an explosion in programming languages, with the development of hundreds of them. PL/I, designed in 1963-as supposedly, an all purpose one, combined features of FORTRAN, COBOL, Algol 60 and more. SNOBOL, Simula and BASIC (1964) [Beginner's All purpose Symbolic Instruction Code] were some of the rest.

BASIC represented an important effort towards an easy to learn and use programming language for non-science students, to facilitating homework. It is considered important because, among other things, it was well-suited for implementation on minicomputers, including that of Gates and Allen’s 4K Basic interpreter of the MITS Altair personal computer (circa 1975). The development of PL/M, on IBM’s PL /I, which run from large mainframe and minicomputers to microprocessor-based systems, enabled the development of Intel PL/M compiler, and, later, of the first personal computer (Ceruzzi 2003). The spread of the latter had to face the lack of a practical mass storage device, and, of an efficient way to write the application software. This was facilitated by the BASIC and the CP/M Operating System.

The decade of the 1970s saw the emergence of a number of influential languages: Pascal, Modula, CLU and C, which aimed for simplicity by reducing restrictions of the type system while allowing access to the underlying system interface with O/S – UNIX. In the 1980s, new concepts arose from the computer usage, which included functional programming (Scheme, ML, Haskell), logic programming (Prolog), and object-oriented programming (Smalltalk, C++, Eiffel). During the 90s, object-oriented languages (mostly C++) became widely used in practical applications. The Internet and the Web drove several phenomena, like adding concurrency and threads to existing languages, and, like increasing the use of scripting languages such as Perl and Tcl/Tk. Java is the most recent major programming language, developed at Sun in the early 1990s, with the original goal of a language for embedded computers.

The "high level languages" differ from their predecessors in a few very important characteristics: First of all, they didn't require knowledge of the machine code. They offered the flexibility of converting a program written in high level language for one computer to run on another computer. Moreover, one single statement in a high level language produced many machine code instructions as the notation for the language was problem oriented (Sammet 1981).

Although software cannot be restricted to the development of the programming languages (see for example the operating systems), the latter dominated the academic field of what was called 'computer science' (Ceruzzi 2003). This discipline arose in the 1950s in universities like

Stanford and Purdue, usually as a division of Mathematics or Electrical Engineering Departments. The 1960s found the discipline trying to define itself. An explicit description of the new field was offered in 1967, when Alan J. Perlis, Allen Newell and Herbert Simon asked and answered the following question: "Is there such a thing as computer science, and if there is, what is it?"[16]. They argued that 'computer' is something human-made and dynamic and should not be subject to the natural sciences but to computer science. The efforts for the definition of the computer science resulted in the ACM Curriculum 68, which outlined a coherent curriculum in computer science (ACM68; Gupta, 2007).

[16] Their article in *Science* is available online at http://akira.ruc.dk/~jv/cs.html.

Mapping the Framework: The Computing Technology and the 'Software'

Software Crisis and Professionalism

It is very interesting that Turing foresaw the 'software crisis' twenty years before it was declared, suggesting already in 1950 that surprises are inherent to computer programming (Copeland 2004). Around 1965, the so-called 'software crisis' came to the front, referring to a set of problems that were encountered in the development of computer software, as software systems begun to be larger because the size was thought to be a sign of excellence (Bauer 1973). This abundance of the software product cost money, resources, storage and system's flexibility. In addition, during the development process, software crisis was fed by an inherent uncertainty about delivery in time. Some blamed the lack of professionals for the crisis[17] (Ensmenger 2001). For them, special attention ought to have been paid to the training of the professionals, the creation of a professional code, and, the standardization of the whole process.

The software crisis was first acknowledged as a problem in a NATO-sponsored conference in 1968. This conference made clear that software development affected the human life in a very serious way but it lacked the tradition of certification and responsibility that characterized the other engineering disciplines (Ceruzzi 2003). A new discipline of software engineering was proposed. Eventually, 'software Engineering' was also proven unable to scale

[17] Bauer quote from Richard Hamming, statesman of the United States computing community, "Present programmers, with their inability to communicate, to deliver on time, to cooperate with others are so bad that they would not be accepted in any other kind of job" (Bauer 1973).

the production process in the same way that other engineering disciplines have, most notably, mechanical engineering, chemical engineering, civil engineering etc[18].

Software as a Business Activity

As suggested above, in the first two decades of electronic computing (1940s and 1950s), the scene was dominated by the hardware and the role of the engineers who built the machines. Bauer argued that this was an outcome of the society's appreciation for the machines and the instrument-makers. The engineer was one of them (Bauer 1978). The engineer was also able to program the machine by himself. This period is characterized by the belief that the computer could not possibly operate in a smooth way and, as a result, the attention of the community ought to focus on make it operate in the first place, not on how it could operate in a better and more efficient way. In other words, the primary goal was to just operate it. Gradually, it became clear that computer programming was something important for both the development of the computer development and for the users, who could better satisfy their needs through a code that could solve specific problems. We reviewed how the specification and the systematization of this pursue gave birth, later, to the development of programming languages, which stood at a higher level of analysis in comparison to the machine and assembly code.

There was no market for computer systems outside government circles in the second half of the 1940s (Norberg 2003). The diversity of the potential users of computer systems like the BINAC and the UNIVAC, and the prospect of developing a new business area, led the members of this community to focus their attention on the needs of prospective users. New software

[18] In all of these other fields "scale" refers both to being able to linearly apply more manpower while achieving linear or super-linear boosts in productivity, as well as producing less error prone and more predictable output.

departments were established, and, mathematicians and others interested in coding for the new computer systems were recruited. The computer companies supplied their computer systems with the system software while the customers were supposed to develop the application software that would meet their needs. The system software was thought to be a part of the computer and, thus, it was a part of the price of the machine (Ceruzzi 2003). As for the applications software, these programs were written in-house for every customer. Gradually, it became apparent that it is more efficient for a program to be modified according to the customer's needs, not to be written from scratch for each new application. This signified the beginning of the 'software package' concept (Haigh 2002). Software packages started to become very popular in the mid-1960s, with the development of general purposes packages for inventory management, file management and management information applications.

By the 1960s, when the sales of the computing machinery had increased, the need of providing programming software with the machines came to the front and boosted software development. The critical point was IBM's new marketing policy, which, starting in 1969, was based on charging separately for computer systems, computer programs, and customer education courses. This policy, known as 'unbundling', gave rise to a significant software and services industry. From that point on, software could be bought, sold and developed from independent vendors. IBM's decision can be interpreted as a move that was related to the 'software crisis' problem. 'Unbundling' enabled the creation of software houses, which aimed at the development of software in a more reliable and efficient way (Ceruzzi 2003).

In the mid-1960s, the computer market was dominated by large mainframes, sold mainly by IBM and its few competitors, mostly those forming the 'BUNCH' group (Burroughs, UNIVAC, NCR, Control Data, and Honeywell). All these manufacturers adopted the same model: they provided system and application software along with the hardware and the services. Aiming at the full circle of customers, these mainframes (like the IBM's System/360) were supposed to be supplied with programs that could work on larger models, thereby saving customer's money. Ceruzzi argues that IBM's 'unbundling' decision came as a result of the realization of the emergence of different industries for hardware and software (Ceruzzi 2003). This model changed only with the advent of personal computers, in the 1980s. In sum, the software industry began to grow significantly in the 1970s, initially as a consequence of IBM's 1969 unbundling decision, and, later, in interaction to the rise of the personal computer.

Some argues that the development of application software, e.g., software products like the VisiCalc spreadsheet and the WordStar word processor of 1979 and 1980, provided the soil for the introduction of the personal computers into the offices (Campbell-Kelly, 2001). In August of 1981, IBM announced the availability of the IBM PC, thereby legitimating the whole industry. From this point to the mid 1990s software became much more important than hardware. This has resulted in the emergence of gigantic software manufacturers, most notably Microsoft.

Definitions, Categories and Restrictions

I am now prepared to provide with a more appropriate historical map of how software has been understood. Software seems to be a rather general term, which is used to describe the programs or other "instructions" that a computer needs in order to perform specific tasks, which shall be done in a specific order and sequence (Ceruzzi 2003). In the 1950s, the term 'software' was not yet in use even though the term 'hardware' was used for representing the computer equipment (Haigh 2002). We also saw that, in the 1960s, the newly introduced term 'software' was used to describe a variety of things, often according to the particular interest of the user of the term (Haigh 2002). At the same period, software gradually started to mean, mainly, the systems software or operating systems. The definition of the software remained problematic. This was apparent at the Rand Symposium of 1963, where participants who invited to think of software production as an engineering activity realized that they didn't share a common definition of software (Haigh 2002).

According to a broader definition, software can also be described on the basis of its delivery: as a product (with or without large or small scale hardware) or as a service through a time-sharing network. Examples of software include word processors, e-mail clients, web browsers, video games, spreadsheets, accounting tools and operating systems. According to a more recent definition, the term software stands for "the written programs or procedures or rules and associated documentation pertaining to the operation of a computer system and that are stored in read/write memory"[19].

Software is often divided into two major categories, systems and applications software (Bauer had argued that, from the 1960s, with the launch of the software industry, independent software firms move from the systems software to the applications software (Haigh 2002)):

- *Systems Software*: System software typically refers to operating systems, network operating systems, database managers and development tools for building other programs such as assemblers, compilers, etc. These software programs are used to control the execution of other programs, the application programs.

[19] Wordreference.com: WordNet® 2.0. Princeton University, Princeton, NJ.

- *Applications Software* or *Application Programs*: Applications software (also called *end-user programs*) includes, for example, word processors and spreadsheets. It is obvious that the applications software cannot 'run' without the operating system or system software.

From a business scope perspective, software is divided into two major categories:

- *Proprietary Software*: Proprietary software is called the software that is owned by an individual or a company (usually the one that developed it). There are almost always major restrictions on its use, and its source code[20] is almost always kept secret.

- *Freeware* (not to be confused with free software) is a type of proprietary software that is offered for use free of monetary charges. However, as is the case with other types of proprietary software, there are generally severe restrictions on its use and the source code is kept secret. Examples of freeware include Adobe's Acrobat Reader and Microsoft's Internet Explorer web browser.

The principal question in the open source software literature is the identification of the economic and non-economic motivations for individual developers to write open source software (Lerner and Tirole 2002; Hann et al. 2002). As for the comparison between the quality of software under open source and closed environments, many researched have shown that open source software is not necessarily inferior in quality to proprietary software (Mishra et al. 2002).Another relevant distinction is the following:

[20] Source code is the form in which a program is originally written by a human using a programming language and prior to being converted to machine code which is directly readable by a computer's CPU (central processing unit).

- *Non-Proprietary Software*: Software that is not proprietary includes free software and public domain software:
- *Free software*, which is generally the same as *Open Source Software* (OSS), is available at no cost to everyone, and it can be used by anyone for any purpose and with only very minimal restrictions (OSI 2006). The “Free Software” concept was developed by Richard M. Stallman in 1982. He founded the Free Software Foundation (FSF), an organization set for the purpose of promoting free software. The most commonly used license, the GNU Public License (GPL), requires that if a modified version of the software is distributed, the source code for such modified version must be made freely available (Free Software Foundation, http://www.fsf.org/licensing/education). The best known example of software licensed under the GPL is Linux.
- *Public domain software* is software that has been donated to the public domain by its copyright holder. Thus, it is no longer copyrighted. Consequently, such software is completely free and can be used by anybody, for any purpose, without restriction.

The restrictions on the use of proprietary software are usually enumerated in the end user license agreements (EULAs) that users must consent to. The EULA describes a legal agreement between Microsoft and the end user, who may install, use, or copy the product--Microsoft refers to their software as a product. The EULA grants the licensee the right to install, use, access, and display only one copy of the product on a single computer, unless a license pack is acquired. The licensee is not allowed to copy, modify or reverse engineer the product. Unlike open source software, the source code is never available for Microsoft products.

According to the Open Source Software concept, its use may help with the following:

1. The freedom and the equality of the information technology and communications users, because the OSS allows every user to use, study, modify and distribute the software, independently of social or economic limits. This means that the OSS give to everybody the same rights with its initial 'author', thus overcoming much of the patent and IP issues that are also connected with the software piracy.
2. The reduction of expenses for the possession the use of the OSS. A lot of studies stress the reduction of expenses that involves the investment in software of open source and this is a very important motive, specifically for the governments. E.g. most Linux distributions can be downloaded from the internet with no fee.
3. The stability and the reliability of the product. Even if the process of OSS development appears chaotic, the final product is often of higher quality and reliability compared to proprietary software.
4. The shortage of the delivery time and the independency from software vendors that bind the public administrator with costly and long-term contracts, with questionable quality of the delivered software (e.g. Greek Taxisnet by Siemens). Moreover, most OSS applications are compatible with international standards, in contrary with proprietary software applications that are compatible only with other applications of the same producer.

There are several studies by reliable organizations that prove superiority of OSS usage by the public administration[21]. Of course the importance of the OSS is not restricted to the

[21] Indicatively, "Economic impact of open source on innovation and the competitiveness" of Information and Communication Technologies sector of European Union - (pdf), Detailed numeric data that prove Free Software

'computerization' of the public sector. For our study, the public administration and its connection to the software is important because it is one of the key elements of the proper understanding of the software piracy phenomenon.

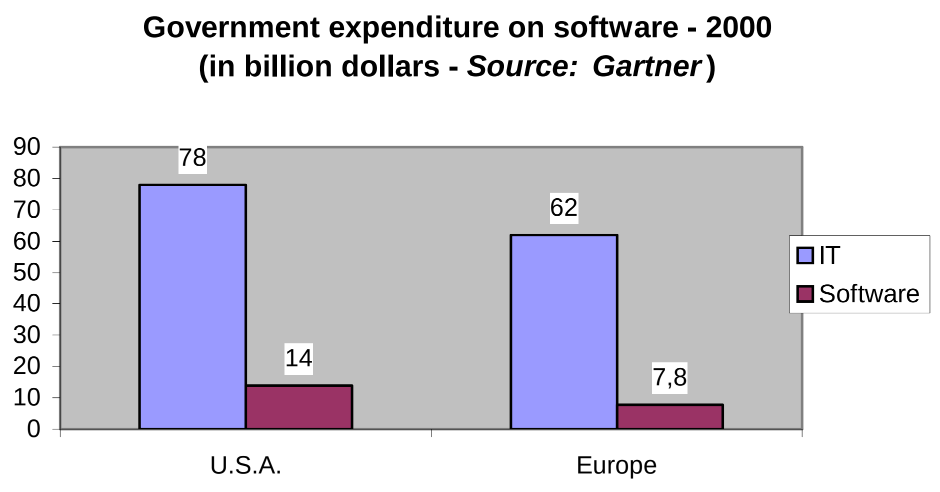


Figure 2: Government Expenditure on Software in Billion Dollars for 2000

More specifically, concerning the electronic services to the citizens, the Greek public administration is characterized by the following elements:

- Low budgets for the purchase of Licenses of Use and thus inability of acquisition of the needed software for the public administration needs.
- Low adaptability and security problems of the used 'closed' software. Gradually this results to the restriction of its usage and to alternative solutions that can be found 'free' and easily by the clerks.

superiority in issues like reliability, performance, scalability, security and total costs of ownership - (link) and "Open Source Software in e-Government" of Danish Board of Technology - (pdf).

- Agreements with specific vendors that force the public services and the citizens to be customers of this supplier (e.g. Microsoft), without alternative solutions.
- Limited percentage of public employees educated in the use of new technologies.

The result seems to be the lack of an ‘open’ philosophy, both in the planning and the organization of information technology infrastructures by the public administration. This result in a linear model of the IT market and services, where only the proprietary software is considered as ‘software’ and in the cases that a specific product of the latter does not serve the needs of the users, or it can be purchased due to its increased cost, the only remaining solution is to be pirated.

OSS could also help the software industry of a country like Greece because those who work on free and open software can acquire the know-how, improve on the software so as to produce additional value. This type of software can allow many software producers to create innovative services. For example, many of the services in the Web as Google, Yahoo and other use free and open software. ‘Google maps’ (http://maps.google.com/) offer an excellent because they are based an open architecture. Many others, professionals or not, build on this architecture and create services of additional value. An example is the identification of points of interests (restaurants, banks, public services, e.t.s.) on maps.

Moreover, the usage of the OSS can boost innovation because open source software is developed actively, on a 24-hour base, by a large number of software engineers. The fact that many different individuals and unions intervene in this software development process leads to a kind of multicultural and multi-organizational software configuration, which is more likely to result into innovative practices. In addition to that, it can promote healthy competition in markets like that of Greece, as the OSS discourage the ‘subjection’ of the users to one or certain software vendors, which could potentially use monopolistic practices in order, for example, to manipulate the software prices.

Beyond the OSS and with regard to the free software is found a wider idea: the open systems. We can think as example how the train networks operate: somebody can take a train from Athens and reach with the same train almost any other European city because the lines, the systems of electrification and the signals are made in order to allow human people to travel in the same way. The same approach would also affect the way that computer users (both individuals and organizations) use the technology in Greece. This can happen due to the “interoperability” between applications, because an application used by one Ministry can comply and operate smoothly with an application of other Ministries or local (e.g.,municipal) authorities. Relative to this is the idea of ‘transferability’, meaning that the applications that accompany the hardware provided by third party vendors can collaborate with hardware installations by other vendors. This means that the user is not bound to a specific or a few only vendors (e.g. Observatory for the Information Society and Siemens[22]), which afterwards can claim additional and increased cost for the support of their own systems.

[22] SIEMENS is the primary contractor of various projects and sub-projects of operational program “Society of Information” that corresponds to 40.000.000 Euros (Πίκουλας 2008).

Pirates and Thieves

Larry A. Welke, a former ADAPSO chairman, argued, in a recently interview, that, “in the 1960s, the most basic legal and commercial issues lacked clear definition. ‘If I write a program for you and you pay me for it, who owns the program? Do you or do I? Or can I write the program and let you use it but I retain the ownership of the thing.’ Well nobody had raised the question before”[23]. As we already saw, when personal computers first came to the market, alongside the ‘unbundling’ of the software, software was not protected as a published entity or as intellectual property. In fact, as we also saw, it the beginning software was hardly though as a discrete entity.

The beginning of the so-called software piracy[24] may be placed in the mid-1960s, when computer programs were distributed with mainframe hardware by the hardware manufacturers. As already described, at the time manufacturers begun selling their software separately from the hardware (‘unbundling’) and, thus, the software (in every form, system software or application software) became a stand-alone product that could be sold and bought. But, even then, the software patents were not allowed. Everything was based on trade secrets rather than patents (Campbell-Kelly 2005). When personal computers first appeared in the computer market, in the late 1970s, software was not recognized as intellectual property[25], so its reproduction was something legal. Only with the Software Act of the late 1980s (1988), we have the first systematic attempt for a form of intellectual property protection. Software was recognized as

[23] Reproduction from Haigh, 2004.
[24] Software piracy as the unauthorized duplication, distribution or use of computer software.
[25] Actually, the term "intellectual property" is not clearly defined. IP (internationally) has five acknowledged forms: patents, copyrights, trademarks, trade secrets, and industrial designs (Letterman 2001).

‘literary work’ and it was recommended that "rightful possessors" of copies of a computer program are authorized to make copies of that program under certain circumstances.

Software copying exceeded all estimations when Microsoft released Windows 95 which required few computer literacy skills from the user. Unauthorized copying did not really become a great issue for the software industry before the personal computer industry boom of the mid - 980s. Until then, computer software was for large corporations and only a few individuals were lucky enough to have their own computer at home. These users formed ‘computer clubs’, in universities and elsewhere, which offer them a way to share the software that they have written.

As presented so far, the discussion about the unauthorized copying of software initiated when the latter became a popular business product and the users were capable of acting energetically by acquiring and using personal computers and software. At the same time, software piracy became recognized as the process of copying and using commercial software which has been purchased by another person. Common forms of software piracy include ‘counterfeiting’, ‘Internet piracy’, and ‘softlifting’. Both counterfeiting and Internet piracy involve creating bootlegged copies of licensed software for sale or distribution. Internet piracy makes use of the Internet to distribute the software. Softlifting involves installing software with a single-user license on multiple machines. It has been the most common example of software piracy within businesses. We can also trace two types of piracy: the ‘end user piracy’ and the ‘retail piracy’. In the first type of piracy, the pirates are the end user, while in retail piracy those who make unauthorized copies sell them to other users with a profit motive.

Being a ubiquitous technology, software can be both a work of authorship and a business process. This is why it comprises rights that are protectable under three bodies of law: ‘trade secret and contract' law (effective protection if the software is a secret), 'patent' law (the equivalent of a monopoly on a computer program), and 'copyright' law. Trade secrets are a form of intellectual property that can be used to protect software. This kind of protection runs against reverse engineering by anyone who is not restricted by a licensing agreement. A patent is an exclusive right to use an invention that is granted by the state. Using the invention without license from the patent holder is called 'infringing' the patent.

For years, software was not believed to be patentable. This changed in the mid 1990s (for an overview of the history of patenting software, see http://www.bitlaw.com/software-patent/index.html). A fundamental distinction between patent protection and trade secret protection is the requirement that the owner of a trade secret use efforts to maintain the secrecy of information. Applying copyright protection on software means that the literal 'text' of a program is protected against replication (Weil 1988). Recently, it has been argued by the courts that making a copy in order to do reverse engineering infringes the copyright. Copyright law prevents anyone from copying source code without the owner's permission, but it does not prevent third parties from independently writing software that performs the same functions as the copyright software. Most countries protect software copyright using copyright and intellectual property rights legislation. One of the main factors of the patent policy is the expectation that intellectual property concerns both the society and the scientific research, and, the belief that a patent and copyright policy will promote science and practical applications, such as computer software (Weil 1988).

The Business Software Alliance (BSA) publishes yearly a piracy study that contains estimates for the piracy rate, i.e., the ratio of the number of pirated software units to the total number of installed software units, for different countries of the world, as well as for different regions of the world. The 2007 BSA *5th Annual Global Software Piracy Study*[26] claims that the worldwide piracy rate jumped from 35% in 2006 to 38% in 2007, costing the industry about US$11 billion in lost sales. Armenia headed the list of malefactors, with a "piracy rate" of 93%. It is worth noting the BSA estimates that 20% of business software programs in the US are illicit. The BSA surveys suggest that both the commercial and non-commercial forms of software piracy are of considerable economic significance.

Software companies do not depend only on the copyright and patent laws in order to prevent software piracy. Both the governments and software companies have been involved in other efforts to combat piracy. In the U.S., the two major groups of software manufacturers involved in this effort are the Software Publishers Association and the Business Software Alliance. These organizations suit companies involved in piracy. They also offer a hotline for reporting software piracy and web pages for online reporting of software piracy. In 1984, 1,200 members of the SPA formed the U.S. Trade Representative's (USTR) office in order to fight software piracy (Gopal & Sanders 2000). In addition to that, the Federation Against Software Theft (FAST) was set up in order to protect the intellectual property of software publishers through parliamentary lobbying, education and raising awareness amongst users. FBI has also set up a Computer Crime Squad. In addition to lobbying and suing the pirates, preventing users from pirating was sought in providing a high-level of support, including documentation and telephone service to the user (Brink 1986). Finally, to fight application software piracy, software publishers have been installing copy-protection features into their software products (Conner & Rumelt 1991). It does seem, however, that all such copying protections are vulnerable (Anckaert et al. 2004).

[26] http://global.bsa.org/idcglobalstudy2007/.

The literature offers a range of studies on the software piracy phenomenon. It is mainly focused on patent laws, software patents, trademark laws and copyright laws. Much of the debate over the effect and scope of the protection of computer programs has centered on identifying the aspects of computers programs that are subject to copyright protection, and, more fundamentally, on finding out if copyright is an appropriate form of intellectual property protection for computer programs (Nimmer 1985). Thus, the only available 'history' of software is a registry of a number of landmark software piracy legal cases.

Apart from the law perspective, there is also an economics and business literature that seeks to study the economic factors of the unauthorized copying. It tries to measure the relationship between the high price of software and software privacy growth (Shin et al. 2004), and, also, price discrimination (selling the same software at different prices to different consumers or at a price of zero) (Gopal & Sanders 2000; Slive & Bernhardt 1998). Some studies relate the high piracy rate with the low IT infrastructure, while others stress that the impositions of high tariffs increase the cost of software and, as a result, encourages piracy. In this context, researchers seek to explain the existence of the same piracy rates across nations, countries and regions (Holm 2003)

A number of researches focus on social and cultural factors, asserting that the social structure of a country and the attitudes shared by members of that society promote and facilitate the acceptance and use of the unauthorized copying. In the context of such analysis, some have suggested that the factors that affect software piracy intentions and behavior can be determined, explained and measured through mathematical models (Limayem et al. 2004). The motivation to pirate software may also have cultural dimensions, involving power asymmetries, gender, and individualism (Hofstede 1997). Related to the piracy motivation factors from an ethical point of view is the interpretation of piracy as an unethical behavior. It has lead to the suggestion that the software industry can benefit by increasing the awareness of the users (Simpson et al. 1994) A different approach is that referring to the 'network externality' (Katz & Shapiro 1986). According to this, a larger network size should be promoted because it increases software's utility to potential buyers. Pirates' work produces a positive network externality, thereby causing the market to expand and ultimately benefit the industry.

We can now categorize the factors that determine software piracy development under the following explanatory variables: economic, technical, regulatory, legal, and, social, cultural, and ethical. For the majority of the studies, software piracy is primarily a phenomenon with economic characteristics, which diminishes revenues, threatens the long-term viability of the software industry, and, "destroy the work of software developers, who invest millions dollars in software development projects" (Shin et al, 2004). It is dramatically described as a "major drain on the global economy" (Limayem et al. 2004). Currently, a good part of the debate the software piracy is taking place over the Internet, where the members of the open source community are against any barriers to software reproduction. Most of this debate is not taking place under an academic framework.

'Software' and its Piracy in Greece

As already mentioned, in the early 1980s mainframe-based programming and manufacturer-specific operating systems and languages gave way to workstation-based programming and standard operating systems and high-level languages. These changes modularized the programming function, i.e., programming could be done independently of the hardware platform and from the other functions of creating software, such as system design. This, along with the reduced costs of imported hardware and software (and many others), caused the Greek software industry to turn to supplying software programs. In the Greek case, there was no development in terms of new high-level languages or internal software development. The domination of the imported software and the special characteristics of the limited domestic market shaped a new industry, focused on the services, the delegation of imported software packages, and, less so, on the establishment of Greek software houses that could serve the needs of the local market. In the early 1990s, these software companies tried to exploit the E.C. regulations and programs in order to establish a communication with the European market in order to sell their products.

In this (second) part, I am going to map the emergence and the subsequent development of a computer software industry and software community in Greece,[27] and, the role that software piracy played in the latter. More specifically, the findings presented here come from a close study of the Greek home computing magazine *Computer for All* (*Computer Για Όλους*), which has been in circulation since 1983. *Computer for All* has not then been a publication only for IT

[27] For the significance of the study of enterprises in the local frame and under the local cultural characteristics, see Misa 1996 and Scranton 1996.

professionals. It could be read from any computer literate. It has hosted articles by both Greek and international authors, including translations of articles published in some of the most known international journals of the field. Via this account, I will relate the history of software in Greece to the broader context of the history of software offered by the available international historiography[28].

The Formation of a Software Industry in Greece

In the mid 80's (1985), a survey by Compupress Ltd revealed that, in Greece, only 9% of the respondents had access to a microcomputer in their working place, and, only 6% in their home. In the same research, 86% of those asked argued that microcomputers could enhance their productivity and 84% that microcomputers could contribute to the domestic economic growth ('Υπολογιστές και Νεοέλληνας', *Computer for All* 26, 90). At the same period, according to a European survey, almost 2 out of 3 Europeans had access to information technology in their working place. A few months later, another European survey reported on the prices of 21 home computers (like commodore 64 and Amstrad CPC 464). It was conducted in 10 state-members (except for Italy and Luxembourg). Greece was the second (after Portugal) more expensive country in Europe in regards to electronic equipment, 50% more expensive than Germany.

In the first years of 1980s, the idea of computerization emerged gradually in Greece, in reference to small offices, business companies and organizations of the public sector. This was, partially, a result of the growing collaboration and contact with international firms or institutions,

[28] For the history of software, see Campbell-Kelly 1995 and 2004; Ceruzzi 2003; Campbell-Kelly and Aspray 2004; Mowery 1996; Coopey 2004; Hashagen et al. 2002.

like the European Community. This is when Greece actually joined the European Community--following in the overthrowing of a dictatorship and the restoration in democracy in 1974, Greece joined the E.C. the 1st January of 1981. The international relations of Greece had since entered into a new phase.

Computing technology in Greece during this early period included mainframe computers, minicomputers, workstations and microcomputers. Workstations and, mainly, microcomputers increased their market share dramatically in the mid 1980s (as in the rest of the world at this time). This turn affected software production for these platforms. One of the major problems that the domestic software industry faced stemmed from the fact that the majority of workstations and microcomputers, and their software, were imported. This meant that the software was neither developed for the Greek needs nor adjusted to the local linguistic or economic characteristics (*Computer for All* 3, 14). Greek market actors tried to give a solution through the creation of software houses. The domestic software industry has been extremely fluid and diverse. It can be categorized into: service bureaus for major international manufacturers like IBM, Apple, etc, independent software vendors, providers of computer services, dealers of international (hardware and software) firms with a separate software department, computer shops and independent users who started by trying to adapt foreign software to their needs only to end up, in many cases, launching a self-made software product. The Greek software houses were, in reality, a combination of some or all of the above.

The computer software industry in Greece, in the 1980s, was dominated by micro-software houses[29], simply because microcomputers dominated the domestic and the world market. They offered an affordable solution to small and medium businesses, which have been the spine of the

[29] Software houses like Abc, Abacus, Scan, Sofragem, Sysco and more (Σώκος 1984).

Greek economy[30]. Worldwide, this widespread use of the microcomputer was due to the development of microprocessor and the associated drop of the cost (Mowery 1996). The rapid expansion of computing technology and the decreased cost of a microcomputer opened up new ways for more applications, in different fields, and, it created substantial business opportunities in the area of software. The first Greek software houses[31] were founded in the end of the 1970s, through initiatives by analysts-programmers who, after having completed their studies aboard, returned back to Greece[32]. These software companies were small businesses, with less than 10 employees; in many cases, less than 5[33].

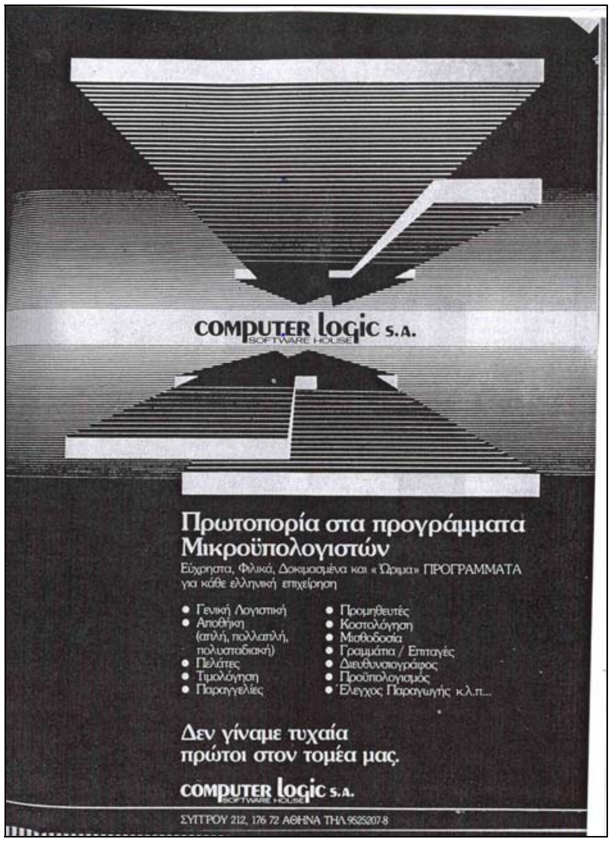
In 1980, the annual revenues of the Greek micro-software companies were estimated to 0.5 million drachmas. This jumped to 1.8 to 2 million for 1981, 7-10 million for 1982 and 50-70 million for 1983. It was expected to further rise to 300 million in 1984 and reach 1 billion by 1985[34]. This was due to the increasing use of the integrated software packages. However, in 1987 the Greek software market could only claim 0,2% of the European market (Βασιλειάδης 1990). The main reasons for the restricted software production had to do with the limiting of the demand to certain software products (mainly accounting packages and specific business applications), with the small size of the Greek software houses as compared to the European firms, with the increasing software piracy, and, with the use of the Greek language, which was an

[30] In 1987, an IT market research by Strategic International ltd showed that the domestic micro consumption increased 93% comparable to 1986, when big systems (like mainframes and mini) increased by 38% (*Computer for All*, 57).
[31] One of the first was M-DATA, which was founded in 1978 by Mr. Eleftherios Michaelides representing well-known IT companies for United States (like Cullinet Software INC). M-DATA also distributed and supported software package for mainframe computers.
[32] It should be noted that the information technology invaded the Greek universities in 1981 (Δατσέρης 1988).
[33] In 1985, according to a research conducted from the perspective of the Integrated Mediterranean Programs and published in *Computer for All* 55, the 63 of the 74 Greek software houses had less than 10 employees (Δατσέρης 1988).
[34] It is important to note that the size of the Greek software market didn't afford an official research, but estimations from those who just 'knew the field' (Σώκος 1984). Official market surveys are published towards the Greek participation in European programs, like the Integrated Mediterranean Programs.

obstacle to the easy adaptation of the foreign software (Βασιλειάδης 1990). As late as in 1989, a very significant share of the software production was developed by self-employees. This had a considerable impact on the quality of the final product and its support.

The traded software handled by the Greek software houses can be split into operating systems, application tools and application solutions, such as accounting and word processing; the software products into as customized software and packaged software. At the Index Exhibition of 1982, Unisoft presented the first micro computers network with complete Greek software. Obviously, the packaged software category was the prevailing one, as the Greek hardware industry was almost absent[35]. In 1985, the Greek software production accounted for more that 40% of the domestic software consumption whereas hardware production accounted for no more than 5% (Δατσέρης 1988). By 1984, a significant number of Greek software houses began to realize that they should focus mainly on the software development. These software houses followed the example of the international software vendors by providing total office automation solutions and by developing, simultaneously, their own integrated software packages. In a 1984 advertisement, the integrated software packages were advertised as "mature programs for every Greek enterprise" (Figure 3).

[35] The first Greek company which assembled a computer and distributed it in the Greek market was *Gigatronics*, which was founded in April 1980. Gigatronics also tried to establish its own 'bundled' software, which was supposed to be superior to the respective from aboard, according to the managing director Mr. Garyfallos. The accompanying software was consisted of the operating system G/DOS, an assembler, a Disassembler, a Linker, the Super Basic, a Data Base, a program for word processing and almost 30 utilities.


COMPUTER logic S.A.
SOFTWARE HOUSE

Πρωτοπορία στα προγράμματα Μικροϋπολογιστών

Εύχρηστα, Φιλικά, Δοκιμασμένα και «Ώριμα» ΠΡΟΓΡΑΜΜΑΤΑ για κάθε ελληνική επιχείρηση

- Γενική Λογιστική
- Αποθήκη (απλή, πολλαπλή, πολυσταδιακή)
- Πελάτες
- Τιμολόγηση
- Παραγγελίες
- Προμηθευτές
- Κοστολόγηση
- Μισθοδοσία
- Γραμμάτια / Επιταγές
- Διευθυνσιογράφος
- Προϋπολογισμός
- Έλεγχος Παραγωγής κ.λ.π...

Δεν γίναμε τυχαία πρώτοι στον τομέα μας.

COMPUTER logic S.A.
SOFTWARE HOUSE

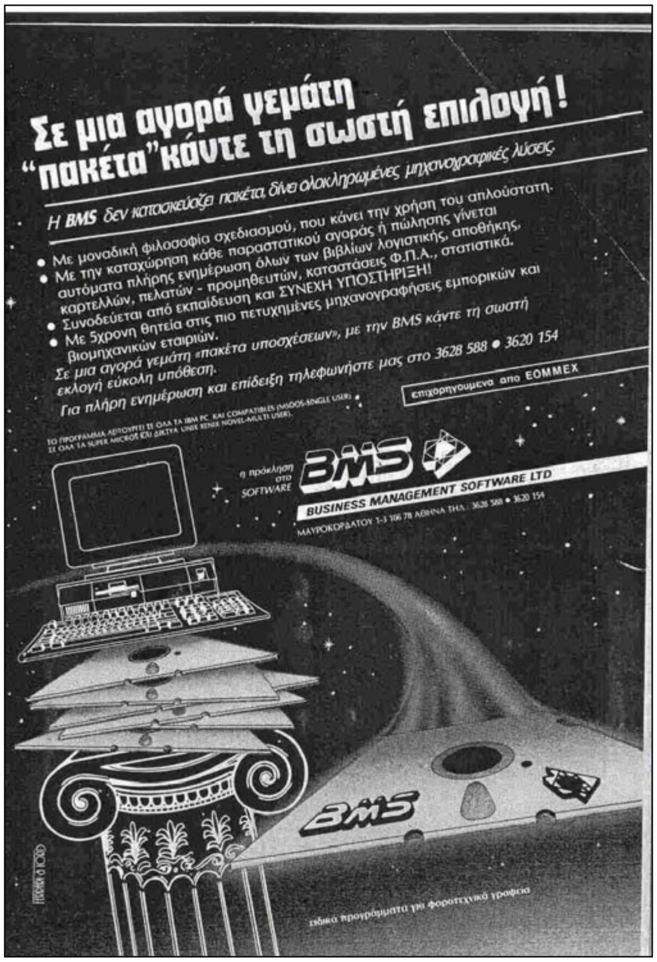
ΣΥΓΓΡΟΥ 212, 176 72 ΑΘΗΝΑ ΤΗΛ.9525207-8

Figure 3: The First Micro-Software Company Advertising Itself as a Developer of "Mature" Programs (*Computer for All* 1984, 87)

The "premature" programs thought to be the customized software ones, which could serve only one customer and may were full of bugs. During this period, the Greek software houses began to adjust their products to the general need for a Greek user friendly interface, which was the major handicap of their international competitors. These adaptation problems go a long way in explaining the high share of the domestic software industry in the Greek software market of the 1980s. The domestic computer software industry identified itself as the advocate of the local needs, which stood in contrast to vendors of international software packages that were addressed all of the world and couldn't be customized for a small European market like that of Greece.

Unlike the software packages developed by international firms like Lotus, WordPerfect, Ashton-Tate and Microsoft, the Greek software houses launched a series of packages for almost all the fields of activity of the small and medium businesses. Large businesses used minis and mainframes and software provided by the computer manufacturers, mostly by U.S. firms like IBM[36]. In this range there was little room for the domestic software production. Domestic production focused all energy and creativity on personal computers and microcomputers. In this context, Greek software companies made sure that they advertised the "greekness" of their products. Below we see a representative advertisement, which comes from 1987 (Figure 4)[37].

[36] According to surveys from Dataquest and Strategic International, IBM dominated Greek market, especially major customers (public sector and multinational enterprises) (Βασιλειάδης 1990).
[37] Note the use of specific symbols: the Greek pillar represents the "greekness" of software, the diskettes the software itself and the IBM-PC/2 image the compatibility of the advertised software.


Σε μια αγορά γεμάτη
"πακέτα" κάντε τη σωστή επιλογή!
η πρόκληση στο SOFTWARE
BMS
BUSINESS MANAGEMENT SOFTWARE LTD
BMS

Η BMS δεν κατασκευάζει πακέτα, δίνει ολοκληρωμένες μηχανογραφικές λύσεις.

- Με μοναδική φιλοσοφία σχεδιασμού, που κάνει την χρήση του απλούστατη.
- Με την καταχώρηση κάθε παραστατικού αγοράς ή πώλησης γίνεται αυτόματα πλήρης ενημέρωση όλων των βιβλίων λογιστικής, αποθήκης, καρτελλών, πελατών - προμηθευτών, καταστάσεις Φ.Π.Α., στατιστικά.
- Συνοδεύεται από εκπαίδευση και ΣΥΝΕΧΗ ΥΠΟΣΤΗΡΙΞΗ!
- Με 5χρονη θητεία στις πιο πετυχημένες μηχανογραφήσεις εμπορικών και βιομηχανικών εταιριών.

Σε μια αγορά γεμάτη «πακέτα υποσχέσεων», με την BMS κάντε τη σωστή εκλογή εύκολη υπόθεση.

Για πλήρη ενημέρωση και επίδειξη τηλεφωνήστε μας στο 3628 588 ● 3620 154

επιχορηγουμενα απο ΕΟΜΜΕΧ

ΤΟ ΠΡΟΓΡΑΜΜΑ ΛΕΙΤΟΥΡΓΕΙ ΣΕ ΟΛΑ ΤΑ IBM PC. ΚΑΙ COMPATIBLES (MSDOS-SINGLE USER) ΣΕ ΟΛΑ ΤΑ SUPER MICROS ΚΑΙ ΔΙΚΤΥΑ. UNIX XENIX NOVEL-MULTI USER).

ΜΑΥΡΟΚΟΡΔΑΤΟΥ 1-3 106 78 ΑΘΗΝΑ ΤΗΛ.: 3628 588 ● 3620 154

ειδικά προγράμματα για φοροτεχνικά γραφεία

Figure 4: "The Right Choice is a Greek Software Package" (*Computer for All* 1987, 186)

By 1988, the domestic personal computer software production covered a large part of the business needs. We can find the known software applications – Accounting, Warehouse, Customers – as well as packages for Video Clubs, Car Rentals, Lawyers, Dental Clinics, Private Schools, Restaurants, etc. Greek software production could satisfy these needs with one requirement only, to have a PC. In the end of the 1980s, the Greek software industry reached a satisfactory level (Figure 5). The next step would be to break through national borders.

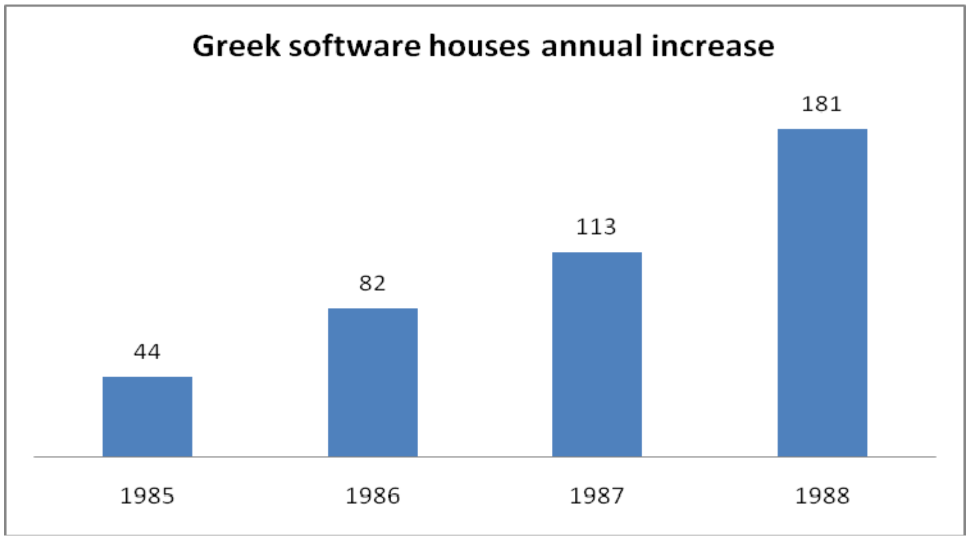


Figure 5: The Annual Increase of Greek Software Houses Number, according to the *Computer for All* Market Guide (years 1985-1988)

In 1985, the members of the Greek computer software industry formed a non-profit organization, the Federation of Hellenic Information Technology Enterprises (SEPE). The founding members of this Federation were the most eminent and oldest Greek software houses, alike ABC and Computer Logic[38]. Any Greek software house with a two years experience in software products manufacturing and annual revenues of 7 million drachmas could join the Federation. The software houses that were members realized that the software industry was not represented in European institutions. The Federation managed to become a member of the European Computer Service Association. Before the launching of the Federation, the Greek software houses were practically excluded from European efforts towards technological collaboration. They had almost no contact with other European software houses. This lack of cooperation explains, partially, why only one public institution -the Technological Institute in Crete- was accredited by the ESPRIT Program. We should mention here that EC had already initiated in the 1980s a series of collaborative programs for research and development (R&D) in

[38] The other initial members were SCAN, SOFRAGEM HELLAS and STEP (Καρατζιάς και Νταφώτης 1985)

state-of-the-art technologies. The cooperation covered programs like the European Strategic Programme for Research and Development in Information Technology (ESPRIT), R&D in Advanced Communications-Technologies in Europe (RACE), and, the European Research Coordination Agency (EUREKA).

In 1984, during the first Greek Information Technology Conference, Greek authorities expressed the will for a more vigorous involvement of the Greek state, in order to "catch the train of technology". This conference pointed to some of the fundamental features of the Greek IT market. One of the conference observations was that Greece didn't have any standard at all concerning information technology. Another major problem was the lack of a Greek language character set, which caused severe problems in the use of international software. The lack of standards also affected the terminology used, which caused severe problems in the communication between Greece and other countries on IT issues. Characteristically, Desyllas, Chairman of the Technical Chamber of Greece (TEE-TCG), translated the term 'software' as '*λογικό*' while Trapezanoglou, the representative of the Greek Software Company, used the current term "*λογισμικό*". The language issue was generating severe tensions between the Greek users and the international software products, as there also was no consensus about the proper and exact rendering of the IT terms in the Greek language[39].

[39] An attempt, but a novice one, for the terminology solution was the circulation of the Greek IT Glossary by the *Greek Computer Society* (GSC) in December 1985 (Τσιτσιμής 1986). We should wait until the 1992 for the creation of the Hellenic Society for Terminology (ELETO), a non-profit scientific association aimed to study, process, systematize, validate and develop the Greek terminology and terminological research. At the same period, EC was trying to promote European standards in data-processing and computer networking. The Standards Promotion and Application Group (SPAG) was such an effort. The group included the same twelve European giants (ICL, Bull, Siemens, Nixdorf, Olivetti, Philips, Thomson and Stet Italia) that had begun meeting in the Commission's Roundtables in early 1982. The goal was the European adoption of the Open Systems Interconnection (OSI) standards for allowing computers of different makes to communicate and share programs and files.

In this first Greek Information Technology Conference, it was argued that a national information technology industry should be formed around software, which was supposed to be easier to export than the hardware. This was an estimation derived from the fact that a significant number of software houses were already successful. It was rather clear that the technological advance of the United States at the hardware level left no space for a domestic (Greek) hardware industry. There was also consensus that Greece should gradually become independent from multinational vendors. It was thought that this could be done in the context of EC. As domestic investment IT programs were extremely scarce (if not altogether absent), the Greek IT policy turned to European partners for guidance and financial aid. Besides the ESPRIT Program, EC established the Integrated Mediterranean Programs for "the improvement of the socio-economic structures of certain southern regions of the Community, in Greece, Italy and France" (Council Regulation (EEC) N° 2088/85 of 23 July 1985). The Integrated Mediterranean Programs started to run in 1986 (to 1993). One of the 12 sections of the Program was the Integrated Mediterranean Program for Information Technology (IMP - Computer Science). Through this program, Greece tried to escape what was largely perceived as technological underdevelopment. In many official speeches, there was even a nationalistic tone about the potential of the Greek scientists and the lack of opportunities in the small Greek country (in comparison to the EC and the United States). The efforts to keep a pace with the other European countries were constant[40]. There was also a constant need for Greek IT products, as the international software couldn't yet be easily adopted by the final users and, mainly, the small and medium Greek businesses.

[40] For example, in 1986, a *Protocol for Scientific and Technological Collaboration* between Greece and West Germany was signed in Bonne. The Protocol was aimed at enhancing the exchange of know-how in various technological fields and especially in Information Technology.

Economic Tendencies of the Greek ICT[41] market

So far, we have seen aspects of the formation and development of software industry and market in Greece. However, we don't have detailed economic statistics for this field in Greece before the mid-1990s. According to the 'IT & Communication Directory '95' of IDC, this was the 1994s situation in Greece:

Activity	**Market Share**
PCs	26%
Software	19%
Multiuser Systems	15%
Professional Services	13%
Peripherals	13%
Support	7%
Telecommunications	7%

Figure 6: Greek IT Market 1994 (Source: Νακόπουλος 1995)

According, however, to the 2000 edition of the European Information Technology Observatory (EITO)[42], it had become clear that the domestic ICT market has been one of the most dynamic in the Greek economy, as compared to the global average. Referring to the ICT

[41] ICT stands for the Information and Communications Technology - or Technologies.
[42] The EITO is a survey of the information and communication technology (ICT) markets in many European countries and is funded collectively by European companies. For further details, http://www.eito.com/.

market annual growth (%) for the years 1996 – 1999: the ICT market in Greece grew from 12% to 24% (Figure 7). Compared to the total size of the EU ICT market in 1999, the domestic market was still found to be one of the smallest EU national markets, together with Ireland's and Portugal's. It constituted less than 2% of the total (Figure 8). As for the per capita spending on ICTs for the same period, while the average was €1100 for the EU countries, Greece has the lowest spending at about €500 (Figure 9).

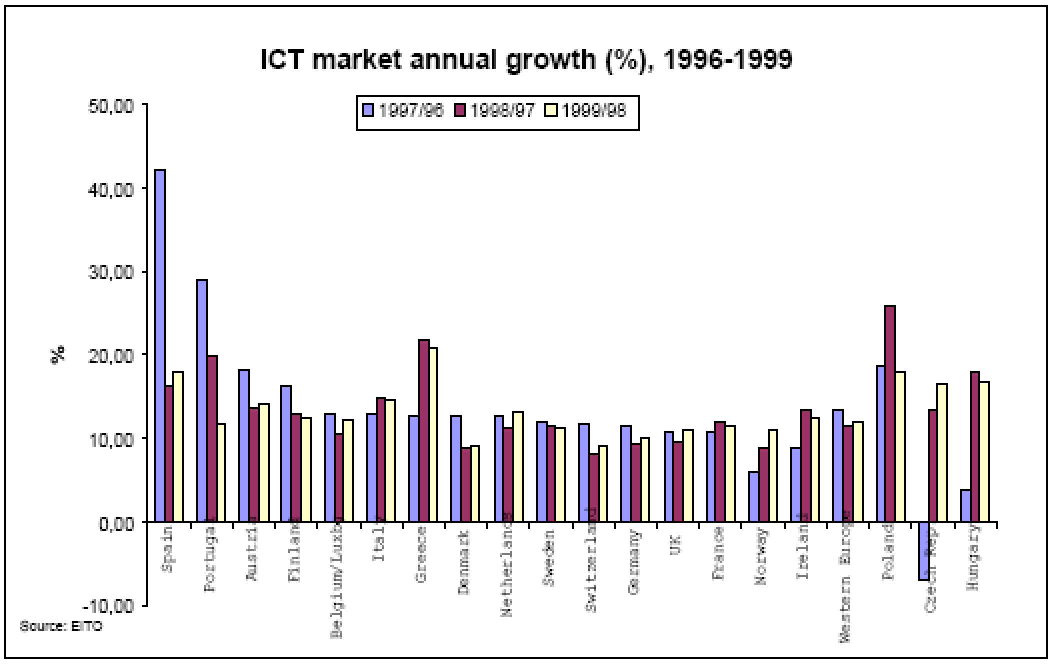


Figure 7: ICT Market Annual Growth, 1996-1999

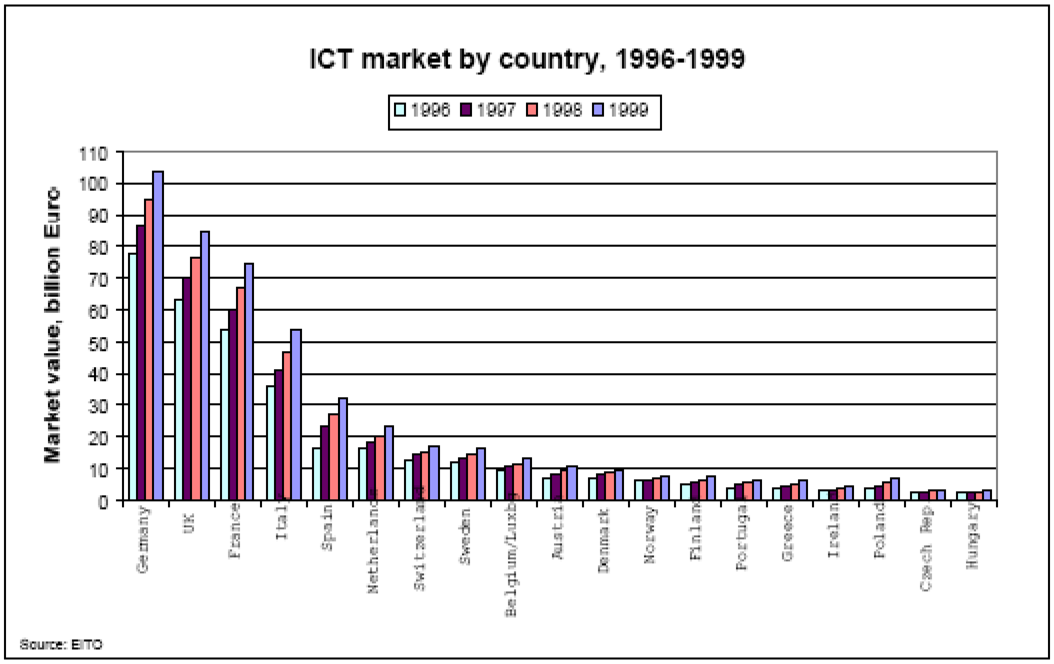


Figure 8: ICT Market by Country, 1996 – 1999

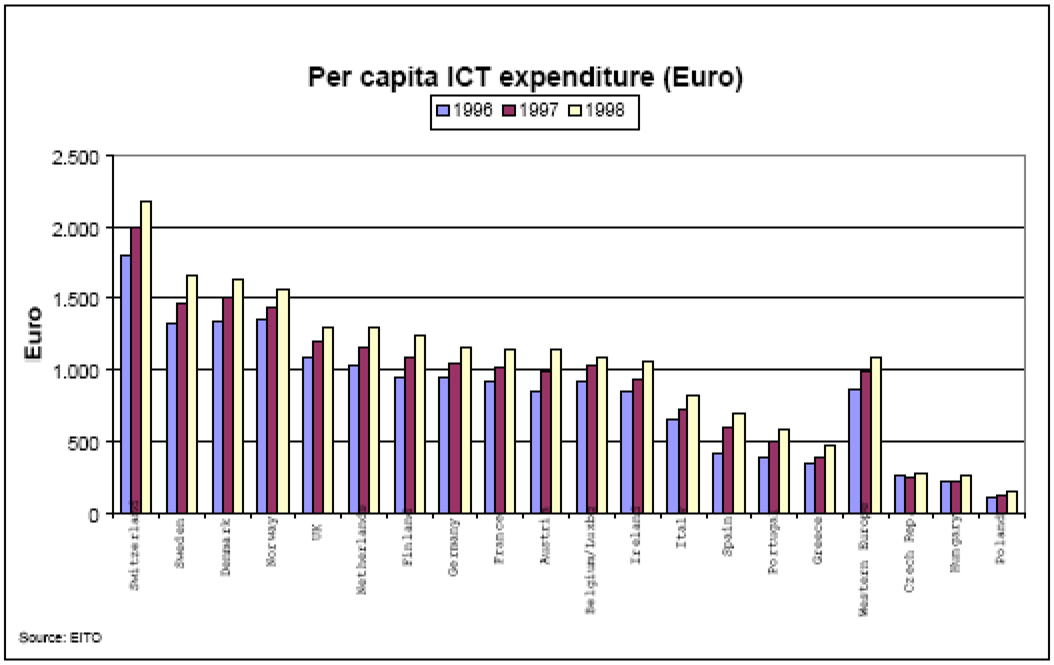


Figure 9: Per Capita ICT Expenditure, 1996 – 1999

As for the IT Market in Western Europe, EITO estimated that the expense for IT products was, in year 2000, higher than 247 billion €. This represented an increase of 11.5% over 1999. It is interesting to note that the highest prediction in regards to growth rates for the following two years referred to software:

IT European (*) Market (Million €)	1998	1999	2000	2001	2002
Hardware	€ 92.547	€ 99.643	€ 109.215	€ 118.350	€ 127.641
Software	€ 38.752	€ 43.928	€ 49.890	€ 56.969	€ 65.771
Services and Maintenance	€ 69.034	€ 78.353	€ 88.399	€ 99.494	€ 111.266
Total	**€ 200.333**	**€ 221.924**	**€ 247.504**	**€ 274.812**	**€ 304.678**
Growth Rates		1999/1998	2000/1999	2001/2000	2002/2001
Hardware		7,7%	9,6%	8,4%	7,9%
Software		13,4%	13,6%	14,2%	15,5%
Services and Maintenance		13,5%	12,8%	12,6%	11,8%
Total		**10,8%**	**11,5%**	**11,0%**	**10,9%**
CGR 2000-1998			**11,2%**		
CGR 2002-2000					**11,0%**
(*) Austria, Denmark, Finland, France, Germany, Greece, Ireland, Italy, Holland, Luxemburg, Norway, Portugal, United Kingdom, Spain, Sweden, Switzerland.					
Source: EITO 2001					

Figure 10: IT European Market and Growth Rates, 1998 – 2002

One intangible growth factor in the European IT market in 2000 was the full understanding, by both governments and companies, that IT had become crucial for future growth in Europe and for the improvement of the European competitiveness in international markets. This new perception led the European companies and governments to promote investments, to encourage the dissemination of technology and to improve the creation of IT skills, in order to rapidly fill the gap with the USA. Investments in new applications (CRM, ERP, SCM, data warehousing and e-business), and the need for companies to equip themselves with an integrated software infrastructure, prompted demand for software packages and services. The update of application software installations in the companies and the construction of Internet infrastructures, in-house and with technology suppliers (e.g. ASPs, ISPs), generated an intense demand for professional services.

Large countries, such as Germany, UK and France, grew at the same rate while Ireland and Greece presented the fastest growth. According to a report on the period 2001-2005, the sum of the companies that were not listed in the Greek Stock Exchange Market also showed good economic results (ICTplus 2007). Interestingly, the major part of the sales was done by IT companies and software houses that were not listed in the Greek Stock Exchange Market.

PARTICIPATION OF LISTED AND NON-LISTED COMPANIES IN NET RESULTS OF THE GREEK IT SECTOR 2001-2005

	NET PROFITS					**% PARTICIPATION IN NET PROFITS**				
	2005	**2004**	**2003**	**2002**	**2001**	**2005**	**2004**	**2003**	**2002**	**2001**
Listed	17.444	42.095	13.101	30.168	67.337	17%	28%	15%	38%	51%
Non-Listed	82.741	110.544	76.932	50.191	65.617	83%	72%	85%	62%	49%
TOTAL	100.185	152.639	90.033	80.359	132.954					
Source: ICTplus 2007										In thousands €

Figure 11: Participation of Listed And Non-Listed Companies in Net Results Of The Greek It Sector 2001-2005

As for the position of the multinational IT companies in the Greek market, it is apparent that their sales during the same period had increased from €466 million in 2001 to €579 million in 2005. The support by the Third Community Support Framework, the replacement of old infrastructures, the growth of new digital services, and the growth of use of broadband networks had promoted the position of multinational companies in the Greek IT market.

Tendencies of the Economic Results of the Multinational companies in the total Greek IT market 2001-2005						
	2005	**2004**	**2003**	**2002**	**2001**	**Tendency**
TOTAL ASSETS	15%	16%	16%	14%	16%	Steady
NET PROFITS	32%	27%	32%	24%	26%	Ascendant
Source: ICTplus 2007						

Figure 12: Tendencies of the Economic Results of the Multinational Companies in the Total Greek IT market 2001-2005

Software Piracy in Greece: Rates and Losses

In regards to software piracy in Greece, I start with reports on the size of unauthorized software use. In Greece, the only studies of the field are those conducted by the BSA Hellas and the IDC (International Data Corporation). Starting in 2003, both organizations deliver a joint Annual Global Software Piracy Study, which estimates the piracy rate in more than 100 countries, including Greece. These studies can be found in the BSA's web site (http://w3.bsa.org/hellas/events/). The BSA divides the world into six zones: Asia-Pacific, Central & Eastern Europe, Latin America, Middle East & Africa, North America, and Western Europe.

For Greece, the 2007 BSA report opens with some encouraging news. Piracy rate has dropped to 58%, presenting, for second straight year, a reduction of 3%.[43] In 2006 the piracy rate in Greece had reached 61%, the highest among the 27 European state-members (36% was the average piracy rate) according to the study of IDC[44]. It should be noted here that only Greece and five other countries (from a total of 108) managed to drop their piracy rate at 6% or more in the last two years. However, according to BSA, almost 6 of the 10 personal computer users in Greece had still used illicit software. Dimitris Sarafianos, legal adviser of BSA Hellas, argued recently that Greece has a piracy rate similar to that of Turkey (65%), Chile (66%), Bosnia (69%), Kuwait (66%), Costa Rica (66%) and Mexico (65%). He stressed that it is even higher than that of the countries that recently jointed the European Union (*Kathimerini* 2007). IDC's

[43] http://w3.bsa.org/hellas//press/newsreleases/greece_release_20070516.cfm.
[44] The study is available in the electronic address www.bsa.org/idcstudy. The study examines the repercussions of software piracy in the economic growth as well as the profits that result in the states that decrease the software piracy and promote the protection of intellectual property.

2007 study estimates that the information technology sector in Greece will see a 53% growth until 2009. Relating directly the decreasing of piracy rate with the economic growth (*Χρήμα*, 2007), another study estimates that a decrease of the piracy rate at 10% would increase the economic growth of the software sector to 60% in 2007 (representing a growth of 8,9% annually). In regards to the previous year (2006), the economic losses from the software piracy were estimated to be €125 millions. This increase of the economic losses is attributed to the enlargement of the information technology market and to the depreciation of dollar in relation to other currencies[45].

The 2005 BSA study reported a piracy rate of 64%, exceeding the mean piracy rate of countries of Middle East (57%) and Asia (54%), while being almost double of the European mean (35%). A few years earlier, 2003, Greece was still on the top of the "black list" of the software piracy in the European Union. The piracy rate was slightly decreasing, but Greece was still having one of the bigger indicators of piracy rate in the world, double that of the EU average. According to the 2003 BSA study, the piracy rate in Greece was 63%, exceeding by far the mean of European Union. This percentage was the highest among the states-members (37%). The cost of illegally installed software was estimated to be €70.8 million ($86.9 millions).

From 1994 to 2002, according to the 8th Annual International Research of BSA, the piracy rate in Greece was dropped by 24%, bringing Greece closer to the other countries of Western Europe. According to the BSA research, in 2002 the software piracy rate in Greece descended to 63% (from 64% in 2001, 66% in 2000 and 86% in 1995), while the economic losses from the illegal software use were limited to $53.674.000 from $61.412.000. At the same

[45] The discoveries resulted from the fifth annually World Survey of Software Piracy of the BSA. The study includes 108 countries and is curried out by the IDC, an international company of analyses and forecasts in the sector of information technology.

time, the piracy rate decreased internationally by ten units in eight years (from 49% in 1994 to 39% in 2002). In Western Europe the decrease was even higher and amounted to 17% (from 52% to 35%).

Year	Software Piracy Rate in Greece
1995	86%
1996	78%
1997	73%
1998	74%
1999	71%
2000	66%
2001	64%
2002	63%
2005	64
2007	58%

Figure 13: Piracy Rates in Greece (Sources: BSA, *Ελευθεροτυπία* 2001)

According to BSA, the software piracy rate in Greece is still high, with an impact on many business sectors, and, also on tax revenues. This means that software piracy should not only concern the beneficiaries of intellectual rights, but, also, the state. Moreover, BSA Hellas estimated the cost of the illegally installed software to 128 million euro, while the economic losses of the software industry due to the use of illicit software, for year 2005, accounted to approximately 150 million dollars. According to a rough estimate by Dimitris Sarafianos (see more below), a 10 percent decrease in the piracy rate by 2009 would contribute €351 million in the Greek GNP, create over 1.300 new jobs, add €111 million income from taxes for the Greek state, and, increase the income of local industry at least by €223 million. A BSA study that was published in January 2008 estimates that, if the piracy rate will decrease by 1 percent by 2012, there will be more than 1.000 new jobs. It will contribute to the national economy €261 million ($385 million), and, it will bring €88 million ($130) in tax revenue--with total expenses of €1.8 billion ($2.7 billion) in the sector of information technology for 2007, Greece has almost 4.700 IT enterprises, employing 32.000 persons and contributing €2.2 billion ($3.2 billion) in tax revenue.

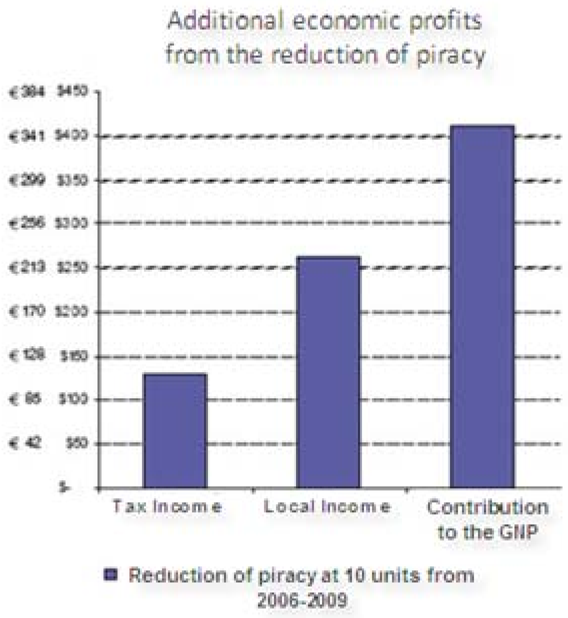
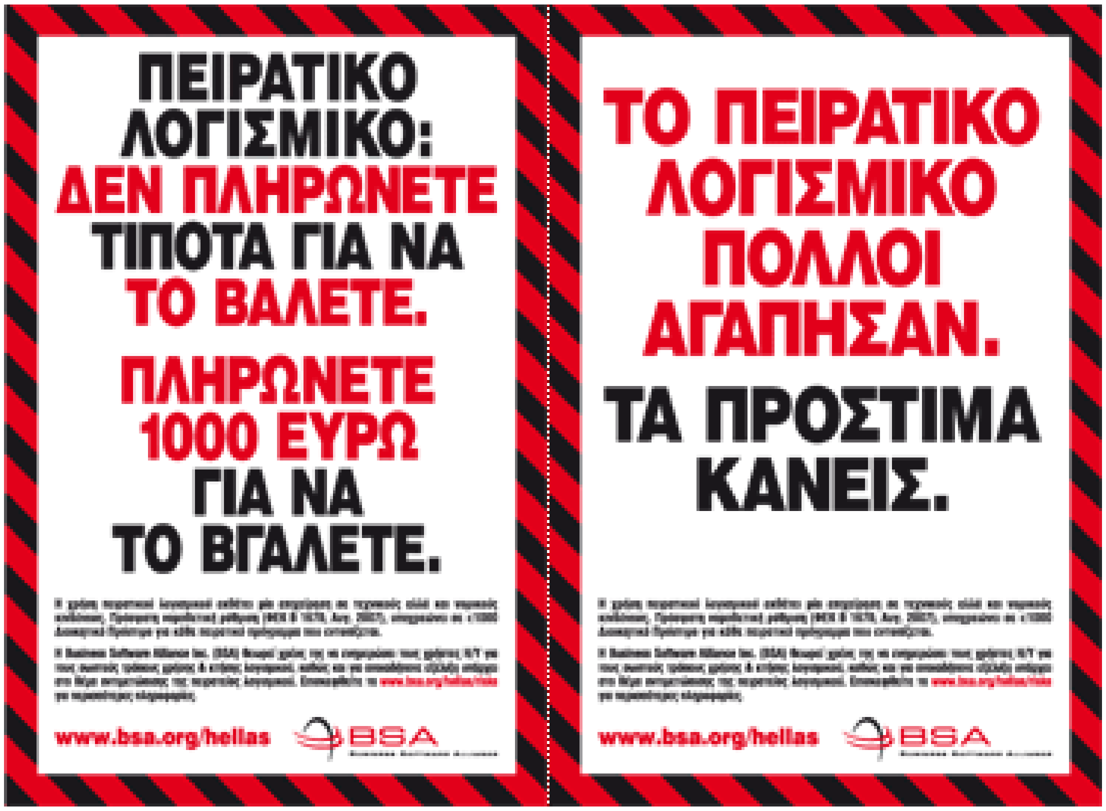


Figure 14: Additional Economic Profits from the Reduction of Piracy According to BSA's Estimations (Source: BSA)

Software Piracy in Greece: Inspections

How can this important decrease of the software piracy, from almost 88% in 1994 to 58% in 2007 explained? The aforementioned reports stress the importance of additional controls (inspections), and actions and fines by the state. The institutions that check on the software piracy issue in Greece are three: the Greek subsidiary of the Business Software Alliance (BSA), the Service of Special Inspections of the revenue department of the Ministry of Economics [YPEE] (the former [SDOE]) and the Police. Each institution employs different inspection methods. More specifically, BSA undertakes tens of inspections every year--hundreds if we include the letters that BSA writes to ask companies to check the software they have installed and to compile lists, so as to control if they use legal or illegal software.

BSA is active in Greece since 1992, aiming (according to BSA's declaration, http://w3.bsa.org/hellas/about/) at the protection from the illegal use of software, at the support of the growth of the Greek market for information technology, which is impeded by the illegal copying and distribution of computer programs. The activities of BSA Hellas were first supported by companies of computer and software production and distribution: Aco Hellas, Byte computer Applications, M-Data, Pouliadis & Co., and more recently, also, Microsoft Hellas (*RAM* 1994). BSA has three main tools: active advertisement, which seeks to warn users about the cost and the dangers of the use of unauthorized software (Figure 15), the BSA hotline, and, reports sent to the BSA Web site (Figure 16).



ΠΕΙΡΑΤΙΚΟ ΛΟΓΙΣΜΙΚΟ: ΔΕΝ ΠΛΗΡΩΝΕΤΕ ΤΙΠΟΤΑ ΓΙΑ ΝΑ ΤΟ ΒΑΛΕΤΕ.

ΠΛΗΡΩΝΕΤΕ 1000 ΕΥΡΩ ΓΙΑ ΝΑ ΤΟ ΒΓΑΛΕΤΕ.

[illegible]

www.bsa.org/hellas BSA

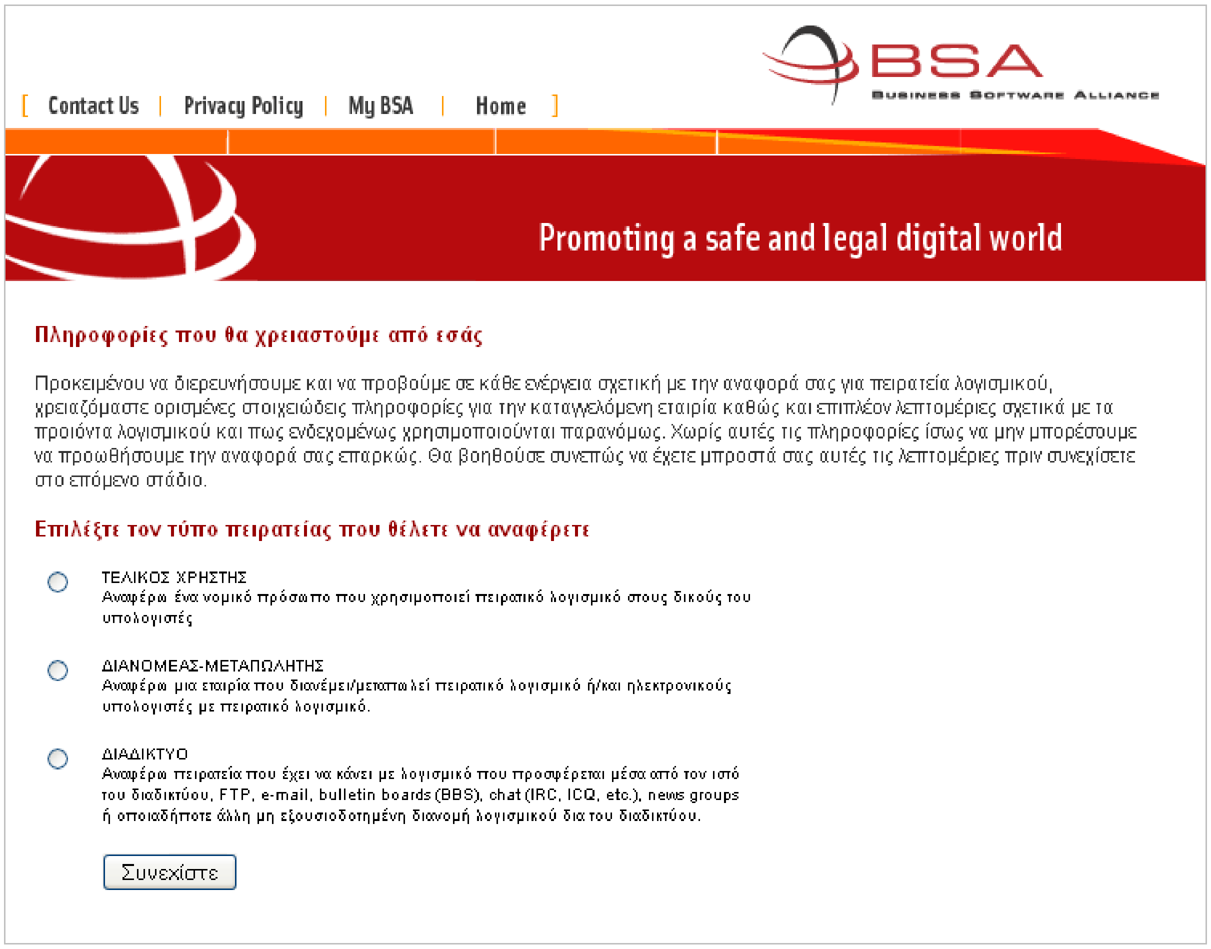
ΤΟ ΠΕΙΡΑΤΙΚΟ ΛΟΓΙΣΜΙΚΟ ΠΟΛΛΟΙ ΑΓΑΠΗΣΑΝ.

ΤΑ ΠΡΟΣΤΙΜΑ ΚΑΝΕΙΣ.

[illegible]

www.bsa.org/hellas BSA

Figure 15: BSA's 2008 Anti-Piracy Advertisements


[Contact Us | Privacy Policy | My BSA | Home]
BSA
Business Software Alliance
Promoting a safe and legal digital world
Συνεχίστε

Πληροφορίες που θα χρειαστούμε από εσάς

Προκειμένου να διερευνήσουμε και να προβούμε σε κάθε ενέργεια σχετική με την αναφορά σας για πειρατεία λογισμικού, χρειαζόμαστε ορισμένες στοιχειώδεις πληροφορίες για την καταγγελόμενη εταιρία καθώς και επιπλέον λεπτομέριες σχετικά με τα προιόντα λογισμικού και πως ενδεχομένως χρησιμοποιούνται παρανόμως. Χωρίς αυτές τις πληροφορίες ίσως να μην μπορέσουμε να προωθήσουμε την αναφορά σας επαρκώς. Θα βοηθούσε συνεπώς να έχετε μπροστά σας αυτές τις λεπτομέριες πριν συνεχίσετε στο επόμενο στάδιο.

Επιλέξτε τον τύπο πειρατείας που θέλετε να αναφέρετε

ΤΕΛΙΚΟΣ ΧΡΗΣΤΗΣ
Αναφέρω ένα νομικό πρόσωπο που χρησιμοποιεί πειρατικό λογισμικό στους δικούς του υπολογιστές

ΔΙΑΝΟΜΕΑΣ-ΜΕΤΑΠΩΛΗΤΗΣ
Αναφέρω μια εταιρία που διανέμει/μεταπωλεί πειρατικό λογισμικό ή/και ηλεκτρονικούς υπολογιστές με πειρατικό λογισμικό.

ΔΙΑΔΙΚΤΥΟ
Αναφέρω πειρατεία που έχει να κάνει με λογισμικό που προσφέρεται μέσα από τον ιστό του διαδικτύου, FTP, e-mail, bulletin boards (BBS), chat (IRC, ICQ, etc.), news groups ή οποιαδήποτε άλλη μη εξουσιοδοτημένη διανομή λογισμικού δια του διαδικτύου.

Figure 16: BSA: Report a Piracy Incident – Step 1

According to Dimitris Sarafianos, legal adviser of BSA Hellas, the latter asks for the reception of actuarial documents and for the issuing of a provisional order so as to order an analytic inventory of the computer programs installed in anyone's technical means (PC, laptop, cd, diskettes) (*Kathimerini* 05-12-07)[46]. Then, a juridical commissary engages experts in the implementation of the particular provisional order. There follows a visit to the company, during which experts ask the employees of the company to switch on the computers in order to check

[46] Available: http://portal.kathimerini.gr/4Dcgi/4dcgi/_w_articles_kathextra_10_05/12/2007_214309

which programs are installed. They record the program with its serial number and then they ask the employees or the legal adviser of company to show them the licensees or invoices they have regarding the purchase of the installed programs.

As already mentioned, checks in companies for use of piratic software are also carried out by the Service of Special Controls, which can impose fines of at least 1.000 Euro per infringement (*Eleftherotypia* 2008). As we already saw, according to the 2007 IDC study of the software piracy, the inspections of the last two years (2006-present) had resulted in the reduction of piracy rate by 6%, limiting it in from 58% in 2007 to the 64% in 2005. Since 2006 there have been dispatched invitations to Greek companies for submitting software inventory statements. More specifically, 3.100 invitations have been dispatched to companies that occupy more than 50 employees, 1.060 to companies that occupy 40 to 49 employees, 2.303 to companies that occupy 26 to 39 employees and 3.010 invitations to companies that occupy 20 to 25 employees (Press release of July 18th, available: http://www.ypee.gr/gr/enhmerosi/deltiatypou2008/dt_180708-2.htm).

Software Piracy in Greece: Legislative regulation

One of the main factors of the reduction of the piracy rate is considered to be the legislative regulation (N. 3524/07) that issues administrative fines of €1.000 for every illicit software installation. More specifically, Greece from 1993 has established a powerful legislative frame for the protection of intellectual rights and BSA and Greek authorities agreed that this resulted in the aforementioned decrease of the piracy rate. It also stressed that at the same period, the domestic software industry developed at 16%. According to the 3524/2007 Act, whoever has installed in his/her computer or server illicit software (that is to say without the legal license of use), in a possible audit he/she will pay a fine of €1.000 for each pirated software found. The same stands for everyone that distributes sells or reproduces piratical programs.

If the programs that will be found illegally installed are up to the fifty and the offender pay immediately the fine (eg. 4 programs x €1.000 = €4.000), then he/she is not persecuted. If, however, the illegally installed programs are more than fifty – no matter if he/she pays the fine or not – he/she is persecuted. Sanctions impose compensation of the beneficiary, which cannot be less than the double sum that the offender would pay, if he/she legally had bought the License of Use for that software. In 2007, the Greek companies paid compensations of €126.000 in the companies - members of BSA for using illicit software. More than the 60% of compensations came from companies of the construction and telecommunications sector.

More specifically, in Greece the computer programs are the subject of protection by the 2121/93 Act for the intellectual property (article 2, paragraph 3a), according to which beneficiaries of intellectual rights for software are companies that produce software as 'special successors' of rights on the programs, based on ipso jure or conventional transfer of the first beneficiaries of rights (Article 40 of Act 2121/93). Article 10, paragraph. 2 of 2121/93 Act also applies. According to this, the beneficiary of intellectual property for computer programs is the person whose name is legally presented on the material part of the product. The beneficiary holds the power of reproduction (installation of program in PCs, registration of program in magnetic or optical means, as disks, diskettes, CD, etc) and the right of placing the software programs (or copies of it) in circulation (distribution, marketing).

There are also urban sanctions (article 65, 2121/93 Act). The beneficiary can demand the recognition of his/her right, the lifting of offence, her/his omission in the future as well as compensation that cannot be inferior to the double economic return that the offender would pay, if he/she legally had supplied the licence for the use of the software program. As for the penalty, (article 66, 2121/93 Act), the offender can be punished with imprisonment of at least a (1) year and financial sentence 1 until 5 millions of drachmas. If the damage that suffered the beneficiary is especially extensive, it could result in imprisonment of at least two (2) years and pecuniary sentence of 2 to 10 millions of drachmas. According to the same article, if the guilty has copied or used unauthorized software as a profession, or, if the circumstances under which the action took place testify that the guilty is particularly dangerous for the protection of intellectual property or related rights, he/she is susceptible to imprisonment of up to 10 years and financial punishment of 5 to 20 millions of drachmas. There are also administrative sanctions, (Article 65A, 2121/93 Act), according to which whoever reproduces without rights (or sells or distributes) publicly computer programs, will pay a fine equal to €1.000 for each illegal copy of it. Responsible for the implementation of the provisions of the Act and the imposition of these sanctions are the Service of Special Controls, the Police and the Custom Authorities.

The first court trial of a software piracy case in Greece took place in 1995, when the BSA turned against two educational institutes, ΚΟΡΕΛΚΟ and COMPUTRONICA, accusing them for using and distributing unauthorized copies. The position of the defendant was that educational organizations should have this privilege and that the software companies proceed in the abuse of their right. The verdict was that the companies and the organizations that use software should purchase the latter along with all the necessary licences of use, even this software is used for educational purposes (Γ.Χ.Π. 1995).

Second Thoughts and Greek Particularities. Is the OSS the answer?

Having introduced to data on software piracy rates in Greece as derived by BSA and IDC studies, we move on to note that these surveys have been criticized as 'dodgy' (The Economist 2005). These yearly surveys were charged with using a rather 'flawed' methodology. To be more specific, piracy rates are calculated by estimating the total amount of software deployed in a market and then subtracting the amount of software sold in the same area over the same period. The 'piracy losses' is the difference. But the losses are not exactly losses; they are rather the estimated retail value of the pirated software. There is here a methodological gap because every pirated software product would not necessarily be purchased at its full retail price. Moreover, it seems that all numbers used into the various formulas are based on estimates, not hard data, and that the BSA does not adjust its figures for the software packages that would not be sold if piracy was not an option. For BSA's account, ten pirated copies of, e.g. Microsoft Office, automatically equate to 10 legitimate copies of Microsoft Office, no matter if OSS options are available or not. This is very important if we take into account the dynamic of the OSS software in the last decade, as presented briefly earlier in this thesis.

It is apparent that these studies assume that every unauthorized copy of the software is an actual lost sale. While there are undoubtedly some lost sales, these studies ignore the fact that many if not most people who have the unauthorized software installed on their computers would do without it rather than pay full price. Moreover, estimating the amount of software per PC and subtracting legitimate sales leaves a residue containing not only infringing copies of software but also legitimate software that was not sold. Losses due to competition from freeware and open source, in other words, are counted into losses due to piracy. In reference to users, BSA tries to estimate how much software a user should have on his/her computer. Then these surveys compare that to how much software products have been sold. If the numbers are different, they assume that this is pirated software and not that the user installed more software than came with the system, or that the system runs Linux. Finally, these studies make the assumption that, if the user hadn't copied that software, he/she would have bought the full product directly from the software producer at full retail price, instead of using OSS or freeware. Noticeably, in 1994, when the BSA study estimated the piracy rate in Greece to almost 90%, BSA members didn't publish their sales (*RAM* 2005).

IDC uses a different formula, taking the number of hardware shipments and multiply it by the average amount of the installed software. This average amount is calculated through surveys and local analyst research, which for Greece are, at best, scarce. In the mid 1990s there was only one research on the sold hardware in Greece! According to the BSA legal adviser (Marinos) of the same period, it was estimated that three software packages were installed (on average) per hardware (*RAM* 2005).

Characteristically, in the same research of the periodical *RAM*, Kyriazis, chairman of the Greek software house Singular argued that their software products were not pirated because his company provided support and continuous updates. In a 1993 by the Greek magazine *CHIP*, the readers were asked about the main causes for the distribution of illegal programs. The most popular answers were: 1. the incomplete and not reliable releases, 2. the non-stop launch of new versions of the same program under the title ‘upgraded’, 3. the fact that these ‘upgraded’ versions incorporates minor updates but it costs the same, or even more, 4. the Greek mentality.

As already mentioned, we can identify three types of business customers for software in the Greek market: Large organizations with many workstations and employees, small organizations with few workstations and employees and home users. Large organizations usually buy the software because they need the support that comes with purchased software and, also, avoid being blackmailed by fired employees who know that they used pirated software. However, in Greece, the available data shows that there is a number of large organizations that use illicit software. The largest organizations are, surprisingly, those of the Greek public sector (*RAM* 2005). This has been attributed to the fact that the computerization of the public services is comparably something new--in many cases it has not even started yet. Other reasons for the usage of unauthorized software are the absence of a central computerized system (each service has its own discrete system) and the presence of a bureaucracy, which halts any attempts for a total solution.

There is also the small business and home computer market. Here there is a strong tendency to use illegally copied software or legally downloaded freeware. Clearly, the legally copied freeware cannot be counted as a loss in sales by BSA’s surveys. Interestingly, in countries where piracy is at its highest (like Greece), Linux has the lowest penetration rate. The model shows that a company holding a monopoly like that of Microsoft can use piracy as an effective tool to price discriminate, and that piracy may even result in higher profits to Microsoft (Casadesus & Ghemawat 2006).

The BSA studies also seem to miss that piracy does not affect different products to the same extent. For example, Adobe Photoshop is generally pirated by people who have an interest in web/graphic design. Somebody could argue that chances are that these people will have no choice but to buy it. For similar reasons computer science students at university have access to cut price (and sometimes free) software so they learn to use it and get 'used' to it. A comparison of, e.g. Photoshop piracy and computer game piracy, is also suggestive. For games, the incentive does not exist to buy it if the gamer already have a pirated version - studies have shown that a large percentage of gamers would buy a copy if they couldn't get it for free. By mixing different types of software together (not even mentioning software in different price), the above surveys produce questionable results. As for the ‘tax evasion’ argument, we must consider that a significant number of software is purchased through the Web, which means that the Greek state cannot control the transactions and collect the respective VAT.

During the first two decades of the computer market in Greece, 1980s and 1990s, the support of the companies that bought software was very poor (*RAM* 1994) and thus these companies didn’t have the motivation to buy software by paying its full price. Only a few software producers provided user manuals and other documentation in Greek, offered help-lines, updates and, more importantly, software adjusted to the local needs. The usual practice was to offer packaged software as a ‘black box’. This reality characterized the Greek IT market and eased the spread of the use of the pirated software. The lack of decent support and additional services has been frequently cited as the main factor for the explosion of software piracy (*Ελευθεροτυπία* 2001). This pirated software, in turn, provide the necessary conditions for the computerization of the Greek economic life, and, for the appearance of many little software houses, which sought to cover customer needs in cheap and reliable software that could support Greek.

Some have argued that a public authority (YPEE) cannot turn into something like an employee of Microsoft and other big software houses who have their own mechanisms (BSA) for fighting piracy. According to this approach, software companies should fund the anti-piracy policy while the Greek state may (and actually do) provide them with all legal weapons in order to safeguard their interests (Μανδραβέλης 2000).

Having in mind the great needs of Greek bureaucracy in IT equipment and the high pirate rate in this sector, operating systems and software applications that do not require user licenses make it a lot easier to avoid piracy. Since, e.g. Linux does not require user licenses, it is a way to combat software piracy. This approach has been supported by several Greek Open Source user groups and communities like the Hellenic Linux User Group (HEL.L.U.G.) and the Free Software / Open Source Software (EL/LAK). The latter has started an initiative for the promotion of the Free Software, as a solution towards local growth (Figure 17).

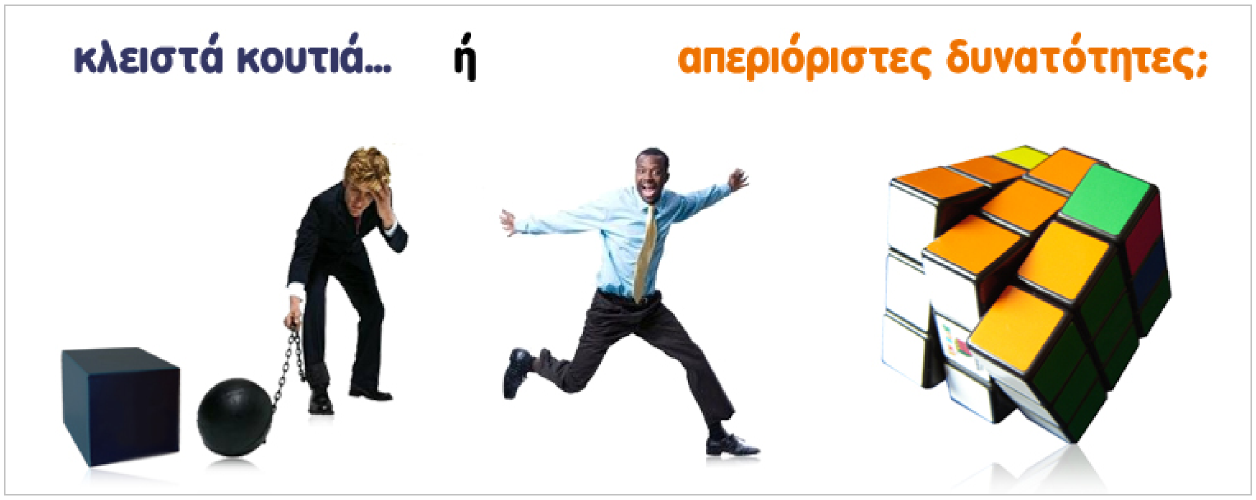


Figure 17: Campaign for the Promotion of the OSS in Greece. EL/LAK Initiative for the Promotion of Free Software - 2008[47].

[47] http://www.ellak.gr/index.php?option=com_content&task=view&id=6754&Itemid=1.

These communities argue that given the security, reliability, and low cost of, e.g. Linux deployments, Linux can benefit big companies or organizations of the public sector. Moreover, with the government's support, the IT market in Greece can be further sprout and not be confined in the proprietary software. According to this argument, if Greeks can make this industry big enough, then they can start reaching the goal of becoming a stronger and more powerful part of the global software industry. At present, a few public services are using OSS. To some, this is a significant improvement for both independence from vendors and curbing of software piracy. Indicatively, the National Printing House, which prints the bulletin with the new laws of the government is using platform CMS (Content Management System) PLONE (open software, even if runs in functional win32). The National Printing House also uses the open source SVG for the free depiction of issues of the bulletin. Moreover, the Prime Minister website (www.primeminister.gr) is using Apache in Win32 operating system, while the Ministry of Transport and Communication and the Ministry of Development use Apache and Red Hat Linux, the Ministry of Education and Religion Apache/2.2.2, the Ministry of Interior Apache-Coyote/1.1, and the Ministry of Tourism Apache/1.3.33 (Debian GNU/Linux). The success of the Open Source Software is recognized by the eEurope 2005 Action Plan. Its use by the public administration is promoted strongly[48]. The increased demand and the need for the adoption of the use of open source software are portrayed, also, in the programs that are financed by the European Union[49] and the open source policies issued by governments all over the world[50].

[48] Commission of The European Communities, COM(2002) 263 final, Brussels, 28.5.2002. Available: http://ec.europa.eu/information_society/eeurope/2002/news_library/documents/index_en.htm.
[49] For an overview of open source case studies in European state-members, see http://ec.europa.eu/idabc/en/chapter/470.
[50] At the third update of *Government Open Source Policies* (Jan. 2006) of the Center for Strategic and International Studies (CSIS), it was found "two hundred and sixty five open source policy initiatives. The majority of the policy initiatives are found in Europe (47.7%). Europe, Asia (27.7%) and Latin America (15.2%) account for 90% of the activity in open source policy. North America was a distant fourth, its share being only 6.4%."

Some Greek authors deal with the software piracy (and the piracy in general) under a different perspective. They suggest that the deeper reason of the phenomenon called "piracy" comes from the fact the technology itself changes dramatically the terms of distribution of intellectual work (Μανδραβέλης 2003). This is something that the companies do not deal with successfully, and this is why they have faced major problems. More specifically, they faced an explosion of parallel distribution of their products, which they tried to fight with police-type meters (e.g. BSA). For example, while raw material of off-prints was decreased dramatically (passing from vinyl to CD), the music industry companies kept price the same level. The enormous profits, however, became the motive for the entrance of other players in the market (even 'outlaws'). New technology has made the distribution of intellectual work even cheaper. The explosion of internet and Peer to Peer (P2P) (which facilitate the exchange of MP3 and other files) threatened both the music and software industry, and, the traditional illegal distributors of their products, excluding them both from the market. According to this approach, prosecutions will not stop the stream. On the contrary, the technology itself invites piracy (Μανδραβέλης 2003).

Conclusions

I have here argued that software is a technology with some discrete characteristics and thus should be approached in a very different way than traditional historiography does. The latter seems to limit itself to a narration of inventors, engineers and machines, thereby neglecting or excluding important elements of this account, such as the business implications, the role of the users, the interactions between social actors, the tensions between different interests, etc.

Trying to consider on the software piracy phenomenon and to relate it to a broader perspective on the role of technology in economic and social life, we first tried to provide a synopsis of the available literature and historiography on the computing technology, which included a focus on the development of the concept of 'software'. Through this enquiry, we came across the realization that software development is a much more complex process than assumed in the literature. For example, the available literature on software piracy has yet to relate it to the historical persistence of the 'software crisis' phenomenon.

In the first part of the thesis, I sought to identify some of the software's discrete characteristics, which make it especially susceptible to unauthorized copying. In the second part, I gave an introduction to the emergence and subsequent development of a computer software industry in Greece, which prepared me to discuss the role of software piracy in this history.

The Greek case is characterized by uniquely high pirate rates. A number of factors (most notably, the limited support by the software companies) facilitated the boom of illegal copying. Illegal copying has interacted with the computerization of the small and medium Greek enterprise, the explosion of the home computer market, the active participation of users into the software development procedure, and the proliferation of small software houses. This thesis made a modest step in considering how all the aforementioned interactions were crucial to the development of a software industry in Greece.

References

Secondary References

Anckaert, B., De Sutter, B., De Bosschere, K., (2004). Software piracy prevention through diversity. *Proceedings of the 4th ACM Workshop on Digital Rights Management, Washington*, DC, USA, 63–71.

Aspray, W. (1982). Pioneer Day `82: History of the Stored Program Concept in the *IEEE Annals of the History of Computing*, 4(4), 358-53.

Aspray, W. (1990). *John von Neumann and the Origins of Modern Computing*. London: MIT Press.

Bauer, F. L. (1973). Software and Software Engineering in the *SIAM Review*, Vol. 15, No. 2, Part 2: Anniversary Supplement, 469-480.

Backus, J.W. (1978). The History of FORTRAN I, II, and III in the *ACM SIGPLAN Notices*, 13(8), 165-18.

Brink, D. (1986). MLJ Computer Corner in *The Modern Language Journal*, 70, (3), 278-282.

Campbell-Kelly, M. (1980). Programming the EDSAC: Early Programming Activity in the University of Cambridge in the *Annals of the History of Computing*, 2 (1), 46-67.

Campbell-Kelly, M. (1995). Development and Structure of the International Software Industry in the *Business and Economic History*, 24(2), 73-110.

Campbell-Kelly, M. (2001). Not Only Microsoft: The Maturing of the Personal Computer Software Industry, 1982–1995 in the *Business History Review*, 75, 103-145.

Campbell-Kelly, M. (2003). *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry*. Cambridge, Mass.: MIT Press.

Campbell-Kelly, M. (2005). Not All Bad: An Historical Perspective on Software Patents in the 11 Mich. Telecomm. Tech. L. Rev. 191. *Available:* http://www.mttlr.org/voleleven/campbell-kelly.pdf

Campbell-Kelly, M., Croarken, M., Flood, R. & Robson, E. (Eds.). (2003). *The History of Mathematical Tables: from Sumer to Spreadsheets*. Oxford: Oxford University Press.

Campbell-Kelly M. & Aspray, W. (2004). *Computer: A History of the Information Machine*. Westview Press.

Casadesus-Masanell, R. & Ghemawat, P. (2006), Dynamic Mixed Duopoly: A Model Motivated by Linux vs. Windows in the *Management Science*, 52(7), 1072-1084.

Ceruzzi, Paul E. (2003). *A History of Modern Computing*. Cambridge, MA.: MIT Press.

Coopey, R. (Ed.). (2004). *Information Technology Policy: An International History*. Oxford: Oxford University Press.

Copeland, J. (Ed.). (2004). *The Essential Turing*. New York: Oxford University Press.

Conner, K., & Rumelt, R., (1991). Software piracy: an analysis of protection strategies in the *Management Science*, 37 (2), 125–139.

Cortada, James W. (1990). *A Bibliographic Guide to the History of Computing, Computers, and the Information Processing Industry*. Westport, CT: Greenwood Press.

Cortada, James W. (1993). *Before the Computer*. Princeton: Princeton University Press.

Cortada, James W. (1996). *Information Technology as Business History: Issues in the History and Management of Computers* (Contributions in Economics and Economic History, 177.). Westport, Conn./London: Greenwood Press,.

Cortada, James W. (2002). Studying the Increased Use of Software Applications: Insights from the Case of the American Petroleum Industry, 1950-2000 in the *Iterations: An Interdisciplinary Journal of Software History*, 1, 1-26.

Davis, M. (1957). The Definition of Universal Turing Machine in the *Proceedings of the American Mathematical Society*, 8(6), 1125-1126.

Davis, M. (2000). *The Universal Computer: The Road from Leibniz to Turing*. New York and London: W.W. Norton & Company.

Ensmenger, Nathan L. (2001). The 'Question of Professionalism' in the Computer Fields in the *IEEE Annals of the History of Computing*, 23(4), 56-74.

Giloi, Wolfgang, K. (1997). Konrad Zuse's Plankalkül: The First High-Level "non von Neumann" Programming Language in the *IEEE Annals of the History of Computing*, 19(2), 17-24.

Gopal, R. D., & Sanders, G. L. (2000). Global Software Piracy: You Can't Get Blood Out of. Turnip in the *Communications of the ACM*, 43(9), 82-89.

Gupta, Gopal K. (2007). Computer Science Curriculum Developments in the 1960s in the *IEEE Annals of the History of Computing*, 29(2), 40-54.

Hafner, K. & Lyon, M. (1996). *Where Wizards Stay Up Late: The Origins of the Internet*. New York: Simon & Schuster.

Hann, I-H.; Roberts, J.; Slaughter, S. & Fielding, R. (2002). Economic incentives for participating in open source software projects in the *Proceedings of the Twenty-Third International Conference on Information Systems*, pp. 365-372.

Haigh, T. (2002). Software in the 1960s as Concept, Service, and Product in *the IEEE Annals of the History of Computing*, 24(1), 5-13.

Haigh, T. (2004). Biographies in the *IEEE Annals of the History of Computing*, 26(4), 79-91.

Hashagen, Ulf; Keil-Slawik, R. & Norberg, A. (Eds.). (2002). *History of Computing: Software Issues*. Berlin: Springer.

Hodges, A. (2006). The Essential Turing, book review in the *Notices of the A. M. S*, 53, 1190-1199. Available: http://www.ams.org/notices/200610/rev-hodges.pdf.

Holm, H.J. (2003). Can Economic Theory Explain Piracy Behaviour? in the *Topics in Economic Analysis and Policy*, 3(1), 1082—1082.

Hofstede, G. (1997). *Cultures and Organizations: Software of the Mind*. New York: McGraw-Hill.

Katz, M., & Shapiro, C. (1986). Technology adoption in the presence of network externalities in the *Journal of Political Economy*, 94(4), 822–841.

Landauer, Thomas K. (1995). *The Trouble with Computers: Usefulness, Usability, and Productivity*. Cambridge, Mass.: MIT Press.

Lerner, J. & Tirole. J. (2002). Some simple economics of open source in the *Journal of Industrial Economics*, 50(2), 197-234.

Letterman, Gregory, G. (2001). *Basics of international intellectual property law*. Ardsley, NY: Transnational Publishers.

Light, J. (1999). When Computers were Women in the *Technology and Culture*, 40(3), 455-483.

Limayem, M.; Khalifa, M. & Chin, W. (2004). Factors Motivating Software Piracy: A Longitudinal Study in the *IEEE Transactions on Engineering Management*, 51(4), 1-12.

Misa, Thomas J. (1996). Toward an Historical Sociology of Business Culture in the *Business and Economic History*, 25(1), 55-64.

Mishra, B., Prasad, A. & Raghunathan. S. (2002). Quality and profits under open source versus closed source in the Proceedings of the Twenty-Third International Conference on Information Systems, 349-363.

Mowery, David C. (Ed.). (1996). *The International Computer Software Industry: A Comparative Study of Industry Evolution and Structure*, New York: Oxford University Press.

Neumann, John von. (1993). First Draft of a Report on the EDVAC in the *IEEE Annals of the History of Computing*, 15(4), 27-75.

Nimmer, T. R. (1985). *The Law of Computer Technology*. Warren, Gorham & Lamont.

Norberg, Arthur L. (2003). Software Development at the Eckert-Mauchly Computer Company Between 1947 and 1955 in the *Iterations: An Interdisciplinary Journal of Software*, 2, 1-26. Available: http://www.cbi.umn.edu/iterations/norberg.html

Guerreiro-Wilson, Robbie, Heide Lars, Kipping Matthias, Pahlberg Cecilia, van den Bogaard Adrienne & Tympas Aristotle. (2004). Information Systems and Technology in Organizations and Society: Review Essay. 'Tensions of Europe' Working Paper. Available: http://www.histech.nl/tensions/.

Pelaez, Eloina. (1999). The Stored-Program Computer: Two Conceptions in the Social Studies of Science, 29(3), 359-389.

Posner, Richard A. (2005). Intellectual Property: The Law and Economics Approach in *The Journal of Economic Perspectives*, 19(2), 57-73.

Rochlin, Gene I. (1997). *Trapped in the Net: The Unanticipated Consequences of Computerization*. Princeton, N.J.: Princeton University Press.

Sammet, J.E. (1981). History of IBM's Technical Contributions to High Level Programming Languages in the *IBM Journal of Research and Development*, 25(5), 520-534.

Scranton, Philip. (1996). The Significance of Spatial Theory for Business Historians in the *Business and Economic History*, 25(1), 65-71.

Shin, S.K., Gopal, R.D., Sanders, G.L., & Whinston, A.B. (2004). Global software piracy revisited in the Communications of the ACM, 47(1), 103–107.

Simpson, P. M., Bannerjee, D., & Simpson Jr., C. (1994). Softlifting: A model of motivating factors in the *Journal of Business Ethics*, 13(6), 431– 438.

Slive, J. & Bernhardt, D. (1998). Pirated for Profit in *The Canadian Journal of Economics / Revue canadienne d'Economique*, 31(4), 886-899.

Weil, V. (1988). Policy Incentives and Constraints on Scientific and Technical Information in the *Science, Technology, & Human Values*, 13, 17-26.

Williams, Michael R. (1993). The Origins, Uses, and Fate of the EDVAC in the *IEEE Annals of the History of Computing*, 15(1), 22-38.

Primary References

Act of Dec. 12, 1980 (1988), Pub. L. No. 96-517, 94 Stat. 3015, 3028 (codified at 17 U.S.C. §101, § 117.

[ACM68] ACM Curriculum Committee on Computer Science (1968). Curriculum '68: Recommendations for the undergraduate program in computer science. Communications of the ACM, 11(3), 151-197.

Brazil launches plan to raise PC use among poor, (2005, May 13) India Daily.

Βασιλειάδης, Κώστας. (1990, 76). 1989: Η χρονιά των απογοητεύσεων. *Computer for All*, p.p. 62-23.

Commission of The European Communities, COM(2002) 263 final, Brussels, 28.5.2002. Available:http://ec.europa.eu/information_society/eeurope/2002/news_library/documents/index_en.htm.

Γ.Χ.Π. (1995, September 10). Δεδικασμένο για την πειρατεία! Ελευθεροτυπία.

Δατσέρης, Νίκος. (1989, 65). Computer Logic: Η Εταιρία με τους μεγάλους…Ορίζοντες. *Computer for All*, p.p. 90-92.

Developing Open Source Software to Advance High End Computing (October 2000), President’s Information Technology Advisory Committee.

Dodgy software piracy data, *The Economist*, 2005-05-19. Available: http://www.economist.com/business/displayStory.cfm?story_id=3993427.

EITO (2000), European Information Technology Observatory, EEIG.

Η ΥΠΕΕ στο κυνήγι του πειρατικού Software, *Eleftherotypia*, (2008, July 21), available: http://www.enet.gr/online/online_text/c=114,id=96467936

Η πειρατεία λογισμικού πλήττει την ελληνική οικονομία και στερεί νέες θέσεις εργασίας, *Χρήμα*, (2007, Ιούλιος – Αύγουστος), p.334.

ICTplus (2007), Greek IT and Digital Telecommunication Market 2001-2005, Piraeus.

Καρατζιάς, Φ. και Νταφώτης, Γ. (1985, 26). Σ.Ε.Ε.Π.: Οι ελληνικές εταιρίες Πληροφορικής ενώνουν τις δυνάμεις τους και τη φωνή τους. *Computer for All*, p.p. 58-60.

Μανδραβέλης, Π. (2003, September 22). Οι πειρατές στην εποχή του διαδικτύου. Απογευματινή.

Μανδραβέλης, Π. (2000, November). Το πρόγραμμα «μην Δικτυωθείτε». Τύπος της Κυριακής.

Νακόπουλος, Σ. (1995, December 24). Το Who is Who της πληροφορικής στην Ελλάδα. Ελευθεροτυπία.

N.M. (2001, June 5). Πρωταθλητές στην Πειρατεία αλλά...., Ελευθεροτυπία, p. 61.

OSI. (2006). The Open Source Definition. Available: http://opensource.org/docs/osd.

Πειρατεία λογισμικού στην Ελλάδα, *RAM*, April 1994, p. 124.

«Πειρατές» λογισμικού 6 στους 10 Έλληνες χρήστες Η/Υ, Kathimerini, (2007, December 5), available: http://portal.kathimerini.gr/4dcgi/_w_articles_mc7_1_18/07/2008_214309.

Πίκουλας, Γ. (2008, July 10). Νέες γαλάζιες αναθέσεις στη Siemens, Έθνος, p. 4.

Study into the use of Open Source Software in the Public Sector (June 2001), Interchange of Data between Administrations, European Commission, DG Enterprise.

Symposium on use of Open Source Software in EU public administrations (February 2001), Interchange of Data between Administrations.

Σώκος, Χάρης. (1984, 18). Διεθνής Έκθεση Θεσσαλονίκης: Η μηχανογράφηση «προ των πυλών». *Computer for All*, p.p 68-73.

The History of Software Patents. Available: http://www.bitlaw.com/software-patent/index.html.

Τσιτσιμής, Γιάννης. (1986, 37). Τεχνολογική εξέλιξη και ελληνική γλώσσα: Το νέο γλωσσικό μας πρόβλημα». *Computer for All*, p.p. 72-74.

Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem in the *Proceedings of the London Mathematical Society*, 42(2), 230-265.

Turing, A. M. (1950). Computing Machinery and Intelligence, *Mind*, 49, pp. 433-460.

Υπολογιστές και Νεοέλληνας, *Computer for All*, 26, p.p. 80-90.