

# Lecture 11. Software industry

Informal and unedited notes, not for distribution. (c) Z. Stachniak, 2011-2015.

*Note: in cases I were unable to find the primary source of an image used in these notes or determine whether or not an image is copyrighted, I have specified the source as "unknown". I will provide full information about images, obtain reproduction rights, or remove any such image when copyright information is available to me.*

## Introduction

Computer software—the "soul" of a computer—is rarely a subject of a passionate discussion of new trends in information technologies. When a new smartphone is demonstrated, its amazing functionality is instinctively but unfairly linked with pressing of keys or touching the screen— a physical sensation connecting our experiences more with hardware than with ... yes, with what?

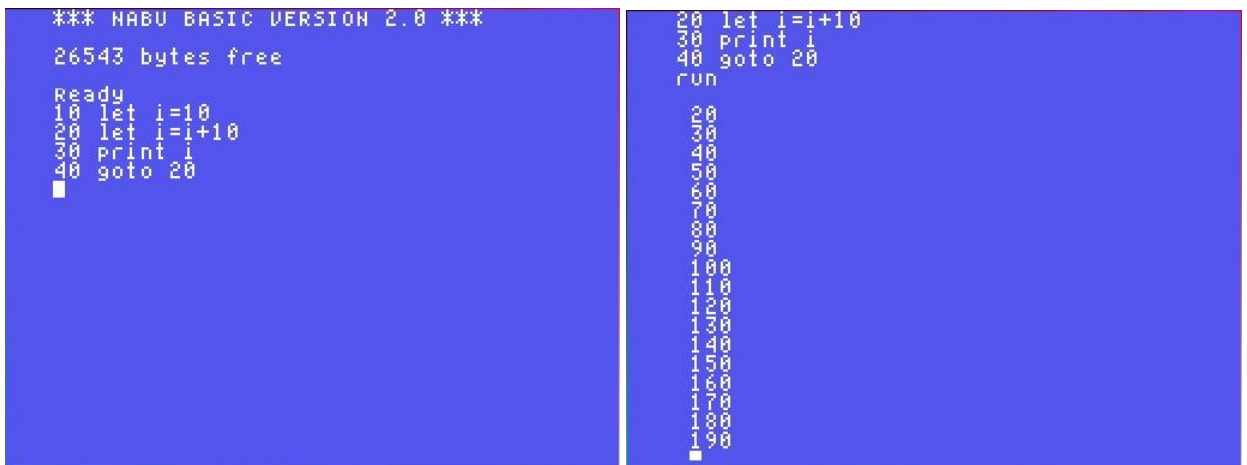
What is exactly happening when a smartphone is turned on? Well, several things are happening: the screen lights on, the radio is turned on, several icons appear, information appears about received e-mails, text messages, updates, and other information. The device begins to "listen" to our requests: screen touches, key presses, spoken commands. How is all of that accomplished?

When a new game is downloaded for, say the Microsoft's Xbox360/Kinect console, what exactly is being downloaded? how does the downloadable stuff that constitutes a game look like? and how a click on the "download game now" link results in such a game finding its way onto a screen connected to an Xbox?



**Computer software:** computer programs and associated data that determine a computer's operating environment and provide users with applications by commanding the hardware to perform specific tasks.

Computer software is delivered to a computer for execution in the form of a sequence of instructions written in a specific programming language and translated into machine instructions. A computer is following these instructions in the order given in the program. The following example illustrates this idea with a little program written in the language called BASIC (see Figure 2). Can you guess what this program instructs a computer to do?



```
*** NABU BASIC VERSION 2.0 ***
26543 bytes free
Ready
10 let i=10
20 let i=i+10
30 print i
40 goto 20
█

20 let i=i+10
30 print i
40 goto 20
run
20
30
40
50
60
70
80
90
100
110
120
130
140
150
160
170
180
190
█
```

Fig. 2. Sample BASIC program (left) and its execution (right).

## Types of computer software

Though there are many distinct categories of computer software (e.g. computer games, operating systems, spreadsheets, databases, text editors, etc.), most of them fall into two main categories: systems software and applications software.

**Systems software** takes care of a computer's functionality. When a computer or a smartphone is turned on, a certain kind of software is executed to perform many functions (e.g. initialization of devices such as screen, keyboard, mouse, and audio card). Another kind of systems software, called an operating system, enables a user to communicate with a computer and its peripherals, and to request the execution of other programs such as launch-

ing a Web browser, a text editor, or a computer game. An operating system is "watching" a user, reacting to keyboard presses, mouse position, screen touches, and other input data. Other types of systems programs are those used by software engineers to design and debug software.

**Applications software** is a large class of programs that are used (executed) by users to directly accomplish some tasks such as the creation and editing of text documents, playing multimedia content and games, or surfing the Web. This class also includes the apps developed for mobile devices.

## Who invented software?

As discussed earlier, a computer program is a list of commands instructing a computer and its peripherals about the operations that they should perform.

It is not difficult to see that similar programs can be designed to instruct not only computers but other machines such as simple robotic equipment at a manufacturing plant that is to drill a few holes in specified places of what would be a chair or a table, or to command a set of mechanical instruments in a mechanical orchestra box (more about this below).

Since Renaissance (and culminating with the Industrial Revolution) mechanical devices with built-in "software" flourished. We have already discussed one of such "programmed" marvel – the mechanical monk or the Clockwork Prayer (see Fig. 3).

The main challenge faced by the designers of these complicated machines was how to provide an automaton with instructions that it would follow to take appropriate actions such as, in the case of the mechanical monk: walking in a square, striking his chest with his right arm, kissing the cross, raising and lowering a wooden cross and rosary in his left hand, turning and nodding his head, rolling his eyes, and mouthing silent obsequies.



Fig. 3. Mechanical monk, "undressed" to reveal its design. National Museum of American History, Smithsonian Institution, Washington, DC. Source: [4].

But looking inside the mechanical monk it is difficult to point out to "software". What we see is an assembly of iron cams and levers of various shapes, powered by a spiral spring (see [4]). The monk could not be re-programmed to exhibit a different behaviour.

## Early "reprogrammable" automata

We begin our search for early "re-programmable" mechanical devices in rather unusual places – art galleries. Let us take a look at the following two paintings, one by Jean Simeon Chardin and the other by William Hogarth. Let us look at the common themes in the paintings: a wooden "instrument", a bird in a cage, a person operating the instrument while looking at the bird.



Fig. 4. J.B.S. Chardin, Lady with a ?? c. 1753, The Frick Collection, New York.



Fig. 5. William Hogarth, *The Graham Children*, 1754, National Gallery, London.





Fig. 6. Bird Organs as depicted in the 18th century paintings.

These mysterious-looking boxes are Bird Organs – mechanical devices capable of reproducing birds’ songs. In both pictures, a woman and a boy “play” an instrument hoping to teach the bird in a cage a new “bird song”.

To see what these Bird Organs have to do with the history of software, let us look inside one of them.

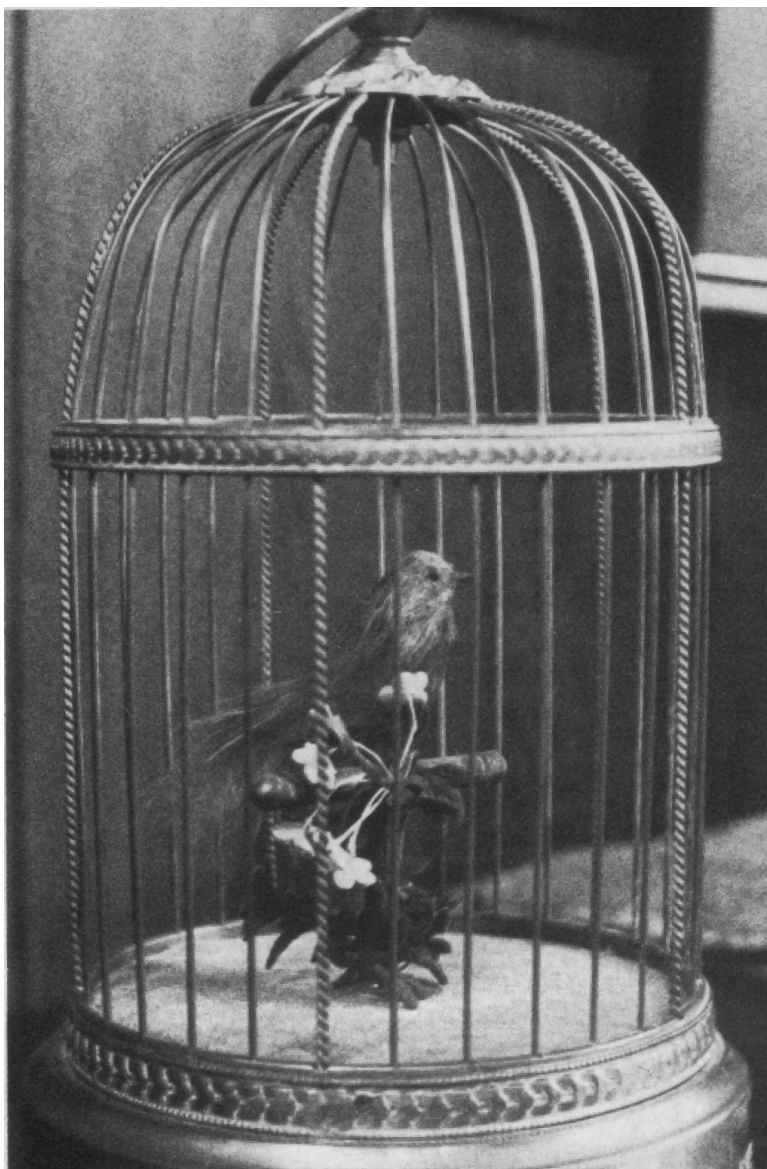


Fig. 7. This "bird organ, a popular novelty of the 18th and 19th centuries ... [was] a sophisticated automaton capable of imitating the sound and movements of a real bird: the wing flaps, the head turns, and the beak moves to the accompaniment of assorted bird whistles." Photo and quote from [3].

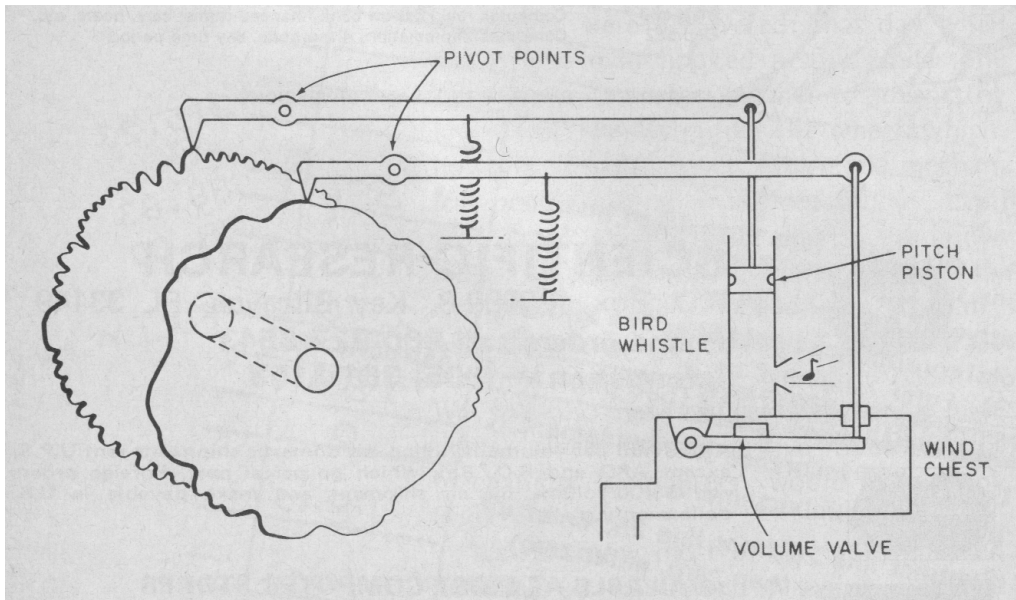


Fig. 8. A design of a typical bird organ mechanism. "Two metal cams control the bird's voice: the far cam controls the pitch piston (located in the body of the head whistle), and the near cam controls the volume valve (located inside the wind chest)." Diagram and quote from [3].

The cams used in the above design store data and "instructions" to raise or lower the pitch, or increase or reduce the volume. When the cams were replaced by another pair, the bird would be "re-programmed" and sing a different song.

## Jacquard loom

Early computers and computer designs used another media and technique for storing data and instructions: **punch cards** and **paper tapes**. At that time, they provided the most convenient way of entering programs and data. Their use explicitly separated data and programs from the device they supposed to control.

To search for the first uses of punch cards one has to go to the early 19th century France and look at the development of the textile industry. In 1801, a French man Joseph Marie Jacquard invented a new type of loom (i.e. a machine used to weave cloth) which, in a fully mechanical way, could manufacture textiles with complex patterns. Without getting into details concerning loom's designs and weaving principles, let us only mention that weaving is done row by row along the so-called warp threads and that each row is "completed" by passing weaving material between selected warp threads.



Fig. 9. Weaving is done row by row along the so-called warp threads...

In Jacquard's loom, the design pattern of each row was stored as a row of punched holes in a wooden "card". The many cards that composed the entire design of the textile were strung together in order. In short, Jacquard loom's operations were under the control of "software" stored on punched cards. (Jacquard adopted the use of punch cards to control looms from Frenchmen Basile Bouchon, 1725, Jean Baptiste Falcon, 1728, and Jacques Vaucanson, 1740).

Simple, repeating designs could be encoded on a single or a few cards.

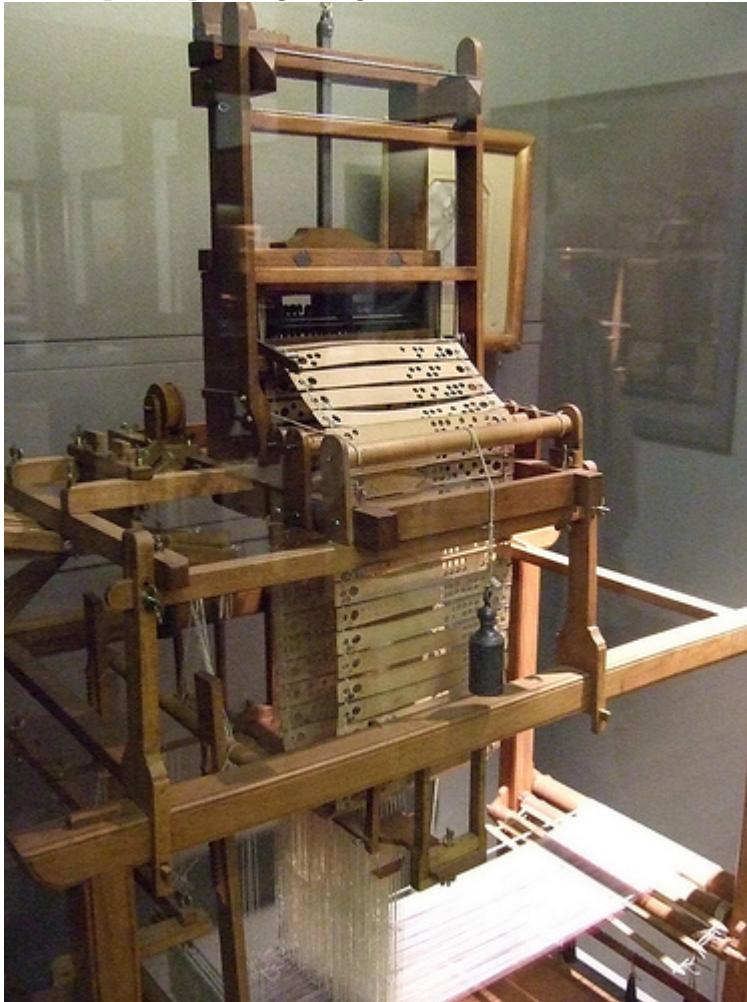


Fig. 10. Jacquard loom. Source: unknown





Apparently, a silk print of Jacquard produced on one of his looms (and depicted in Figure 12) used 10,000 punch cards (see [1], p. 64).



Fig. 12. Portrait of J.M. Jacquard woven in silk, c. 1839. Source: unknown.

The enormous success and impact of the fully automated looms reminds us that "there is a high technology in every age, not just in our own. The technology of electronics is merely the latest link in a continuous chain of technological developments spanning 20,000 years." [3]



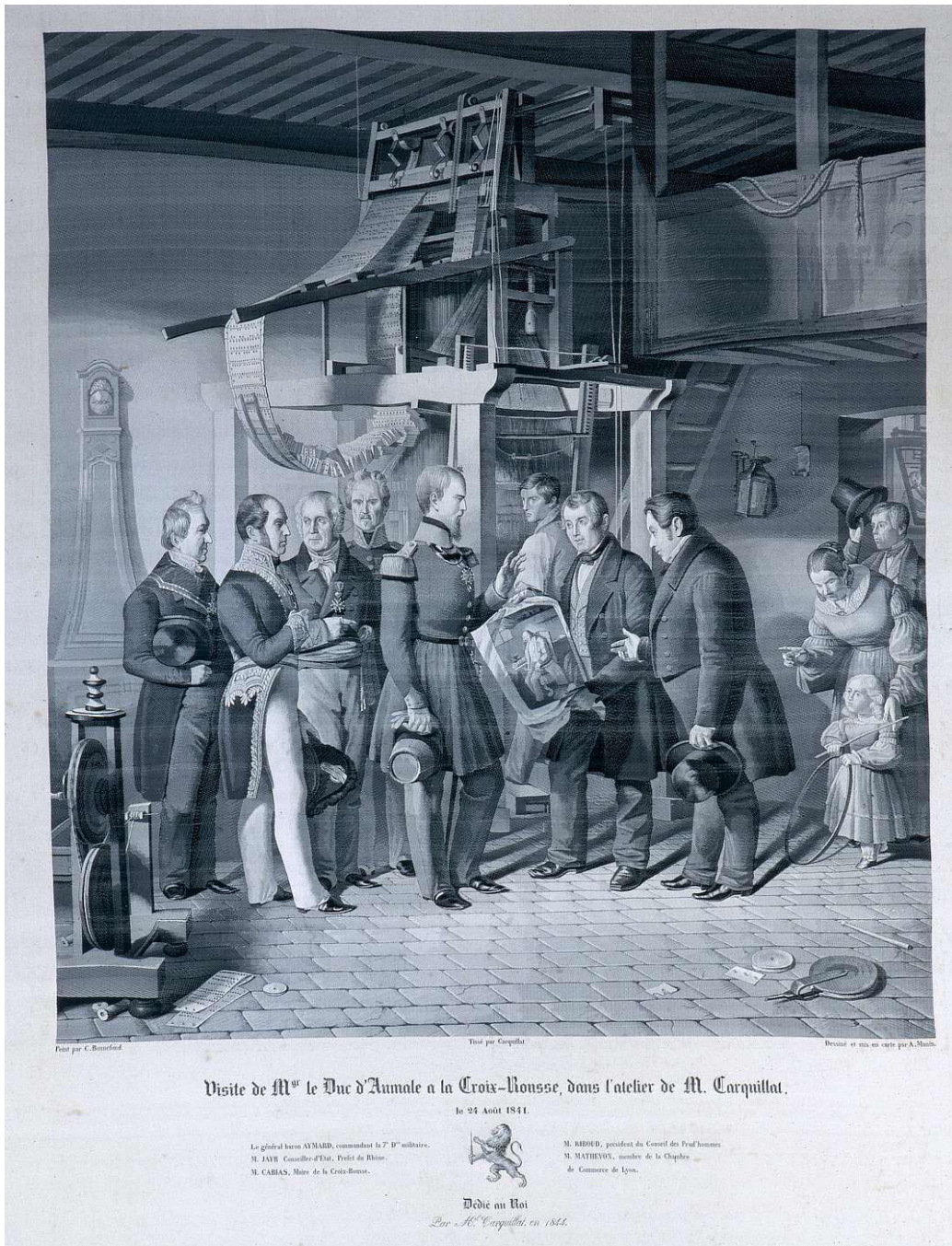


Fig. 13. A visit to a Jacquard loom woven in silk, by M. Carquillat 1844.  
 Source: <http://www.duke.edu/web/isis/gessler/collections/jacquard.htm>.



Since the 18th century, punch cards were also used to store data and instructions other than those used by textile industry. Another significant application area for these cards were mechanical musical instruments such as this impressive French orchestra box depicted in Fig. 14, made in the 19th century and still in perfect operational order when photographed in 2008.

As in the Bird Organ, the hand cranked punch card reader of the mechanical orchestra transfers information about pitch and volume values for various wind instruments placed inside the box.



Fig. 14. *El Bucanero* band playing traditional Cuban music to the accompaniment of a mechanical orchestra. Photographs by Z. Stachniak

## Software after Babbage

Charles Babbage was the first inventor of computing devices to use punch cards for storage of data and program instructions. His Analytical Engine (1834, see lecture 3) was to be operated under the control of a program supplied to the machine on punched cards.

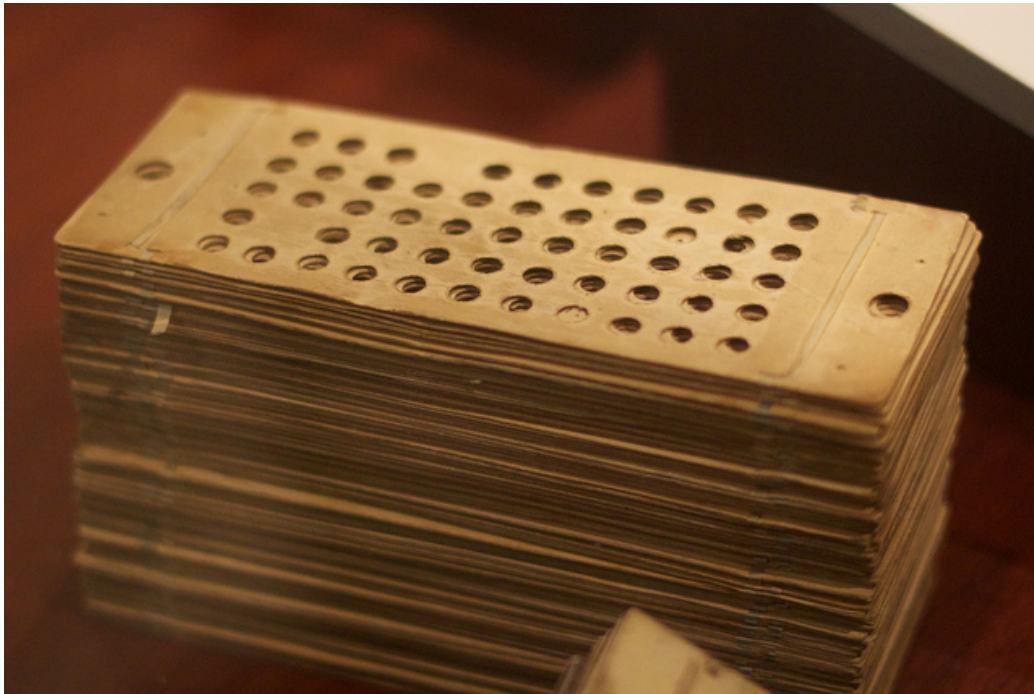


Fig. 15. A stack of punched cards for Babbage Analytical Engine. Source: photo by cogdogblog, <http://www.flickr.com/photos/cogdog/4905713747>

Punch cards reappeared in the late 19th century again when Herman Hollerith used them to store data destined for his Card Tabulating Machine (see lecture 6). Punch cards would continue to be used by Hollerith's Tabulating Machine Company and, after the formation of IBM, by the new company, until 1970s.

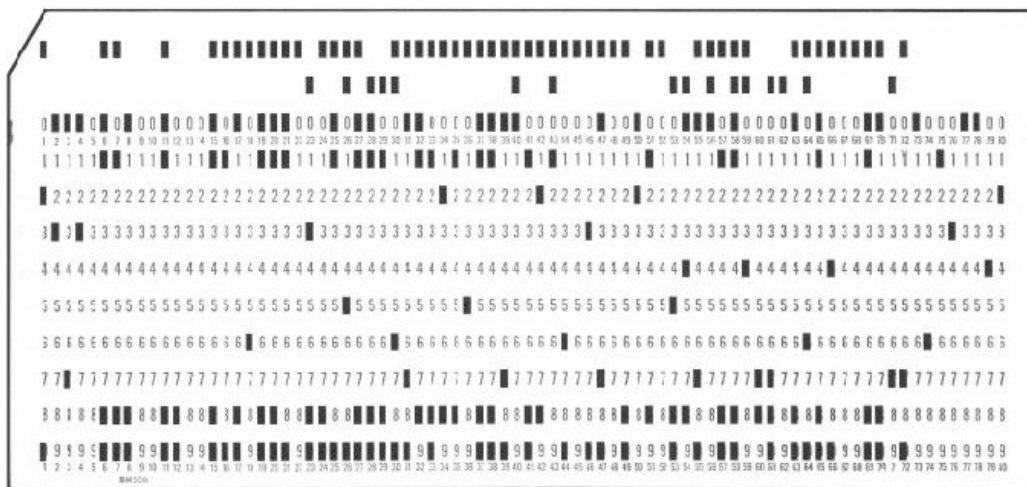
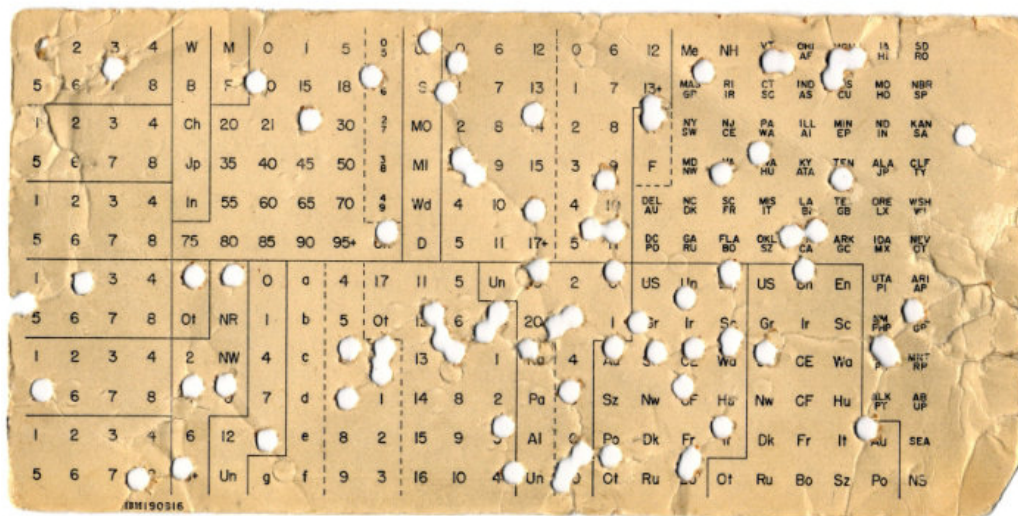


Fig. 16. An original Tabulating Machine Company's punched card (top) and an IBM 360 card (bottom). Sources: York University Computer Museum and Tao Shi, <http://taoshistat.wordpress.com/2011/10/21/computers-and-statistics>

## Software in the age of mainframes

In the early era of mainframe computers there was no software industry. A computer owner had only two sources to acquire software. The first was the computer manufacturer which provided its computer with some general purpose software and "bundled" its price with the price of hardware. The manufacturer was offering no additional software.

A mainframe computer user in need of additional software dealing, say, with specific business applications (e.g. payroll, inventory management, banking, file management, or report generation utilities) had no choice but either to develop its own software in-house from scratch or by hiring the services of a programming firm that would develop required software under contract, typically a single copy for each customer. In both cases, obtaining such software was very expensive.

Since the 1960s, the production and distribution of software has changed dramatically (see [5,6]). Just a few at first, then thousands of software firms began to sell multiple copies of software to multiple customers.

These firms increasingly found opportunities to package the software they had already written and deliver it to multiple customers, a situation that promised potentially high profits given the low cost to reproduce already developed software. [6]

John Cullinane, who founded his own software company in 1968, described this process in the following way

The idea of software products as a business came to me when I was working with Philip Hankis Inc., a traditional contract programming firm in the 1960s. We designed and implemented a payroll system for a bank in New Orleans. ... Then we designed and implemented a similar payroll system for a bank in Connecticut... Then we received a contract from Marine Midland Bank in Buffalo... for another payroll system... About the same time that we finished that payroll system, a bank in New Jersey called and wanted a similar payroll system for the same computer configuration. Why not sell them a system we had just finished rather than reinventing the wheel? [6]

By the mid-1960s, the software industry selling application packages was already established. An application package contained the software itself, some manuals, and a licence agreement allowing, among other things, some customer support. All of that placed in a box and shrink wrapped. The software itself was delivered on a variety of media: punched cards and paper tape, or on magnetic tapes.



Fig. 17. Magnetic tape equipment attached to a Bendix G-15 computer. Several of such computers were installed at the University of Manitoba. Source: Keith Smillie, <http://webdocs.cs.ualberta.ca/~smillie/ComputerAndMe/Part14.html>

Software was stored for execution in a computer's internal memory implemented as magnetic drums, core memory, and (since 1956) on hard drives.





Fig. 18. The first hard drive developed by IBM in 1956 for its Ramac 305 computer had just 5 Mega bytes (Mb) of storage capacity (or 5 million characters). Source: Kyle VanHemert, <http://www.gruponeva.es/blog/noticia/7331/el-disco-duro-de-5-mb-lanzado-por-ibm-en-1956-que-pesaba-mas-de-una-tonelada-fotos-retro.html>

Today, little smartphone SDM cards and flash drives have storage capacity equivalent to thousands of Ramac 305's hard drives!! For instance, the little SDM card in Figure 19 can store 6,400 times more data (its capacity is 32GB) while 128GB flash drives are storage-equivalent to 25,600 of Ramac 305's hard drives!!



Fig. 19. 32GM SDM card from Samsung. Source: Samsung.

By the end of the 1960s, the software industry was established as an independent economic entity.

## The microcomputer software explosion

Since the emergence of the computer industry in the early 1950s, every decade or so, a new computing paradigm is introduced which is immediately reflected in the way software is designed, distributed, and used.

What these new computing paradigms bring to the world of software is not necessarily the elimination of the old forms of software design and delivery but their novel forms. The bundled and packaged software introduced in the 1950s and 60s are offered for this day (e.g. smartphone software is bundled with hardware, and new operating systems, such as the coming Microsoft Windows 8, will be also available in a packaged form).

The appearance of the first microcomputers and, soon after, personal and home computers in the second half of the 1970s, resulted in the rapid expansion of the software industry that had to supply millions of new type of computers with all kinds of software from operating systems and programming languages to applications. The formation of the microcomputer software industry introduced:

- new forms of distribution: computer stores, electronic distribution, free software,
- new software categories (e.g. computer games, home economics, educational software, electronic spreadsheet programs), with bestsellers such as the CP/M operating system (1974, Digital Research), Microsoft BASIC (1975, Microsoft), Microchess (1976, P. Jenings), VisiCalc (1979, Personal Software), or Flight Simulator game (1976, Bruce Artwick, subLOGIC Corporation);
- new forms of software advertising: in dedicated microcomputer magazines, catalogs, and newsletters published by computer stores and computer clubs, during computer events (such as exhibits, shows, and fests), even in popular press, on television, and radio (in general-interest programs dedicated to computing).



## Microcomputing and electronic delivery of software

Today, a large number of software products are purchased and delivered not in a physical packaged form but electronically, on-line. With a click of the computer mouse or a touch of an icon on a smartphone, a software is downloaded and installed.

The electronic delivery of software dates back to the rapid development of home and personal computer industries in the late 1970s and early 1980s. It was based on the idea of the delivery of software directly to home microcomputers, game consoles, and other dedicated microprocessor-based hardware (I will refer to all that hardware as software players) via a standard common communications network such as phone and cable TV networks or using FM radio waves.

For a small monthly fee, the owner of a software player could subscribe to a service that would offer an electronic access to software and data. This was a cost-effective alternative to microcomputer packaged software offered on ROM cartridges, audio cassettes, and floppy diskettes and distributed through variety computer stores, large departmental stores, and hardware manufacturers' outlets (e.g. Radio Shack). Some of the early electronic microcomputer software delivery services are listed in Table 1.

System	Company	Country	software player	delivery method	In operation
Telesoftware	Ceefax	UK	BBC micro	TV, VBI	1983-89
Telesoftware	Prestel	UK	multi platform	telephone	1981-?
PlayCable	Jerrold Mattel	USA	Mattel Intellivision	CATV	1980-83
GameLine	Control Video Corp	USA	Atari 2600	CATV	1983-83
The Games Network	The Games Network	USA	Window game console	CATV	1983-84
NABU Network	Nabu Mfc.	Canada	Nabu PC	CATV	1983-86

Figure 1: Early commercial microprocessor software delivery systems.

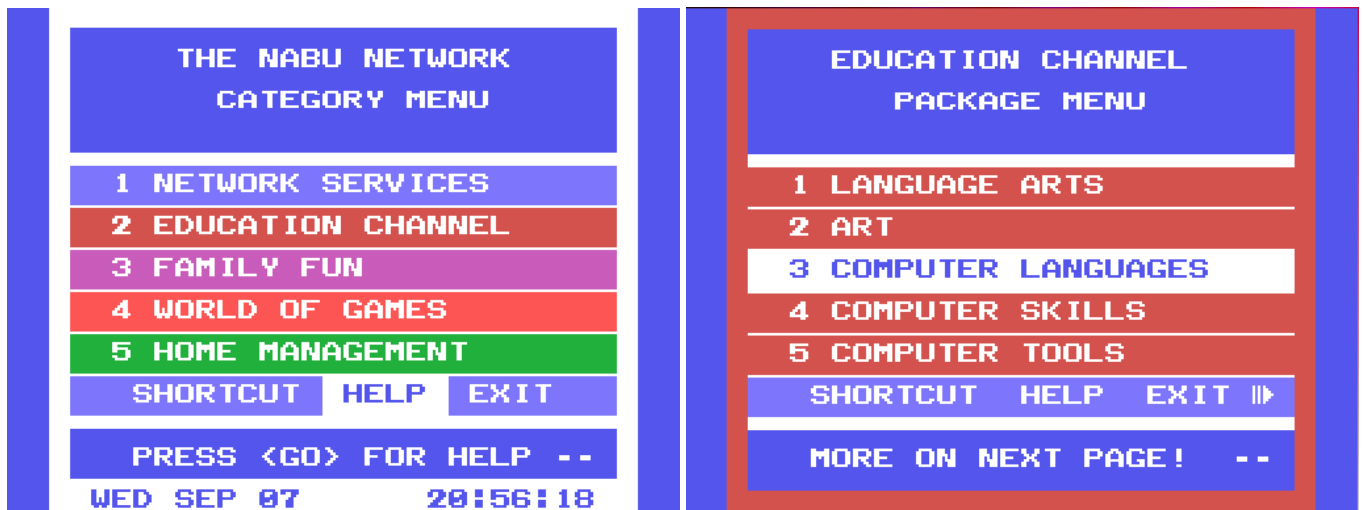


Fig. 20. NABU Network software selection menus (1984). Source: York University Computer Museum.

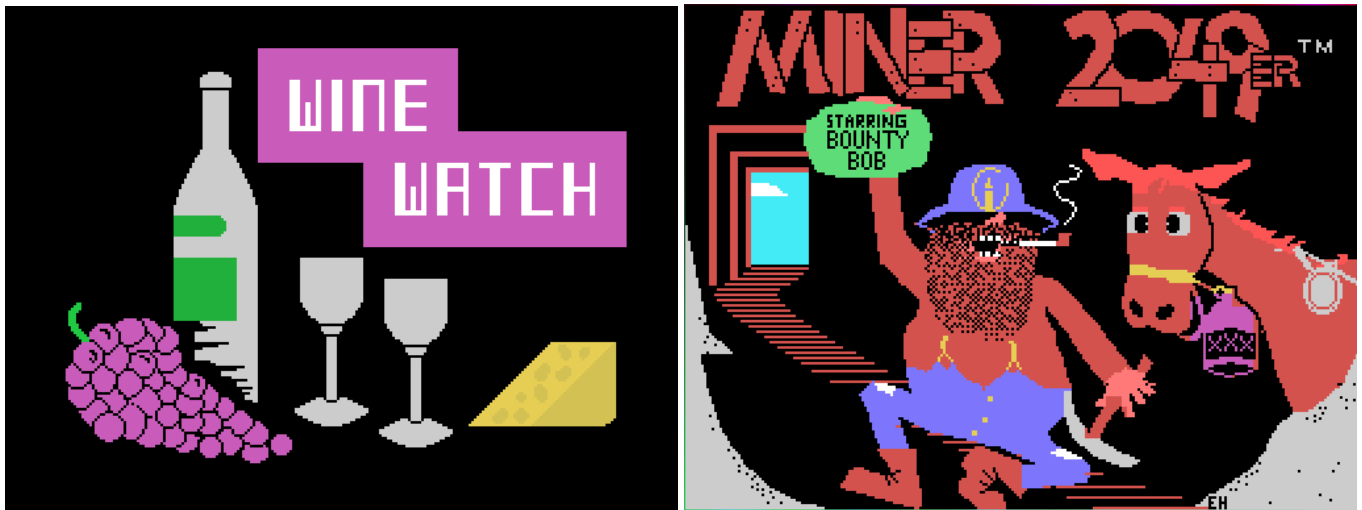


Fig. 21. Sample NABU Network application software Source: York University Computer Museum.



Fig. 22. Two Ceefax Telesoftware screen shots.

## **BBS systems and downloadable software**

Thanks to inexpensive modems, computer bulletin board systems (BBS) grew in popularity and began offering an electronic exchange (downloading and uploading) of non-commercial software.

Computer enthusiasts flocking around BBS systems created the first virtual communities. A BBS user could download programs written in popular languages such as BASIC, PASCAL, or C. These programs could then be executed on a user's hardware either directly (when the program was written in an interpreted language) or after compilation. In either case, the downloaded software could also be stored on an external media owned by the user.

The following pages contain screenshots of two BBS introduction pages.

```

/-----\
/ [-----]
|   TRS-80   | ----
|           |
| MODEL III | ----
| [-----] |
\-----/

The Searchlight BBS
(516) 724-0971
Smithtown, NY

-----
|           | Volksmodem
|           | 1200
-----

-----
|           |
|           | Disk Drive
|           |
-----

---> TRS-80 Model III Computer - LDOS Disk Operating System - One
      Megabyte Floppy Storage - 5 Mhz CPU Speedup - Volksmodem 1200
---> Original Software - Fast, High Performance BBS System for the
      Serious User - Over One Year and 11,000 Calls in the Making
---> Public Message Boards - Electronic Mail - Original Downloads For
      TRS-80 & IBM - Xmodem Transfer Supported - Online Games - User
      Polls - Dating Questionnaires w/Computer Matching - Text Files
---> Free Access - Easy Validation - GUEST Account Always Available -
      Plus Many More Suprises - Call (516) 724-0971 - 300/1200 Baud

TRY THE INCREDIBLE SEARCHLIGHT BBS!
The Hardware-
A truly advanced system, featuring Radio Shack's state of the art
Model III Z80 based microcomputer, boosed to a blinding 5.1 mhz!
48,000 bytes of internal memory, and *3* big disk drives with nearly
ONE MEG of total storage..! Over 50 programs and text files
available for downloading, over 200 active messages!
The Software-
Custom programmed in the powerful, leading-edge TRS-80 Level III
BASIC language! Many powerful and convenient features: Log in with
your own personal password! Check your mail! Read and post messages!
Download files! Log off!

```

Fig. 23. TRS 80 BBS welcome page.



Fig. 24. Cracked Ice BBS welcome page.

## Computing in the clouds

Presently, a number of corporations, companies, and individual users no longer buy application software or store corporate data on their own computers. Instead, they subscribe to software and data storage resources, they access and use application software on-line and store data, over a network (typically the Internet), on remote computers. In this computing paradigm, software is a service provided remotely by "cloud computing" companies. This, in essence, is what is now called **cloud computing**.

Is cloud computing a new concept? Not exactly. As with electronic delivery of software, such remote software services were tried before and were particularly popular forms of the so-called time-sharing services in the 1970s and 80s. A customer could login into a time-sharing service (i.e. a remote computer providing software, databases, and storage resources) using a dumb terminal connected, typically, to a phone line, and could execute a range of software tools and application programs.

A Toronto-based software and communications firm I.P. Sharp Associates Ltd. (IPSA) was one of such companies. Since mid 1970s, IPSA offered its communications capability, its on-line SHARP APL software and application program environment as well as private and public databases and libraries via its world-wide communications network – IPSANET.

By the end of the 1980s, IPSANET was linking over 800 cities in 80 countries and territories. The company maintained one of the world's largest collections of on-line business-oriented data.

For a number of customers, electronic access to software represented the easiest and most effective way to satisfy their computing needs, being free from problems associated with software installations, updates, and patches. In this model of software use and access, a user was neither owning software nor retaining it after terminating the time-sharing session.

After twenty years, cloud computing re-emerged again on the strength and world-wide reach of the Internet. The example of cloud computing illustrates a profound rule of inventing and advancement according to which they are just steps in a complex chain of preceding findings and inventions leading us

into even more profound discoveries.

## Conclusions

Computer and Information technologies involve both hardware and software. It is hardware that determines what kind of applications, as defined by software, are possible. It is software that determines how useful and user-friendly a device (hardware) is.

The future we are building is based more and more on software. The influence of software in all areas of industry, economy, and in ordinary life will likely persist over the next few decades. The demand for software of ever-increasing degree of complex will continue and so will debates concerning privacy, security, and sharing of software.

## References

1. S. Augarten, *Bit by Bit, An Illustrated History of Computing*, Ticknor & Fields, 1984.
2. JP. Ganapati, Jacquard's Loom Will Weave a Durable Web, *Wired Magazine*, July 7, 2009. [http://www.wired.com/thisdayintech/2009/07/dayintech\\_0707/](http://www.wired.com/thisdayintech/2009/07/dayintech_0707/)
3. J.M. Williams, Antique Mechanical Computers. *Byte*, July 1978, pp. 48–58.
4. E. King, *Clockwork Prayer*  
[http://www.blackbird.vcu.edu/v1n1/nonfiction/king\\_e/prayer\\_introduction.htm](http://www.blackbird.vcu.edu/v1n1/nonfiction/king_e/prayer_introduction.htm)  
(follow the links on the bottom).
5. E.W. Pugh, Origins of Software Bundling, *IEEE Annals of the History of Computing*, January-March 2002, pp. 57–58.
6. L. Johnson, Creating the Software Industry, *IEEE Annals of the History of Computing*, January-March 2002, pp. 14–42.