Video Summary

History of Software: The Programmers

(Video published by Open University U.K. in 1997)

Preface

- According to a classical definition of a machine, it is something, a device, that somehow channels the forces of nature through a mechanism into a very specific purpose
- The computer does not fit that definition, the computers purpose is only specified when you load a piece of software into it and execute that software
- Software is at the heart of understanding these new general purpose machines
- The history of computing, until now, has been dominated by the milestone machines such as the ENIAC, the IBM 360, and the Apple II
- Software is also a cumulative process; today's software developers builds on the work of their predecessors
- How do you talk about a software milestone like that…it slips through your fingers when you try to grab it

Jean Sammet, Programming Language Consultant
"One day my boss came over to me and asked me 'Jean, do you want to be a programmer for this computer?' I said 'What's a Programmer?'. He, being a nice guy and a very competent engineer said, 'I dunno, but we need one!"

The Programmers

- What are the milestones of software?
- Who were the first programmers?
- How has their job changed?
- Will we need them in the future?
- …. In the beginning, there were computers, general-purpose machines.… but no software
- The birth of modern computing, according to the British, began with Colossus. It was used to crack the enemy codes of World War II.
- The war was also impetus to the development of ENIAC, according to the Americans, to be the first computer. It was initially designed to calculate the firing trajectories of field guns.

Kay Mauchly Antonelli, ENIAC Coder
"The ENIAC itself was 80 feet long and, because it had over 18,000 vacuum tubes in it, it generated a lot of heat"

Betty Holbertson, ENIAC Coder

- *" ENIAC was more like a Mechano set where you put pieces together to make it make a machine of your problems"*
- Colossus and ENIAC computers are said to be computers because they were general-purpose machines. Their purpose was changed by changing their wiring.
- Rewiring these computers was not software in the sense we know now, but was a step in the right direction.
- The limitations of both machines were that you had to rewire them for each new problem. Their only advancement was that they were patchable so you never had to cut and solder wires, merely re-patch using a modular connection panel.
- The stored program is the key to the modern computer age because it allows the program itself to be manipulated inside the machine as though it were just any piece of data. Ultimately, it allows one computer program to operate on another computer program.

The Next Generation of computers would need to have a stored program.…and the first practical machine was developed in Cambridge, England….

Prof. Maurice V. Wilkes, Cambridge University

- *We have developed a system of programming based on the use of a library of sub-routines.*

The EDSAC Film (1951)

- This is the first film showing the operation of the stored program computer to be made.
- It had a memory where you could store, temporarily, a sequence of machine instructions
- Programs were entered via paper-tape reader and stored for process
- *…. EDSAC marked the birth of modern computing. A feature of all subsequent computers would be the ability to store and process programs….*
- Programs would need programmers.
- Every single thing that the computer does would have to be specified. Usually it's a list of binary numbers or a symbolic code replacing 0s or 1s.
- That is an extremely tedious job.
- Most programmers up to now were math graduates, many of them women.

Joyce Currie Little, Early programmer

- *"The level of detail on machine language programming was so great that many people spent months & months & months writing code & deciding what code to be written…the amount of labor was intensive."*
- Nowadays there is no need to directly communicate with the computer's hardware. An English command like "Send Mail" might involve several translations but it is all done by the machine. The command may have been issued by a program written in a High-level language…the machine then might translate or compile this into another symbolic language. Now at a lower level, it includes precise hardware instructions. It will eventually translated to 1s & 0s of binary numbers.
- High-level languages avoids mistakes which are easily made by coding in lower level machine language.
- What a high level language did was to give you a productivity lever so that by writing a tenth as much code, you could actually generate the same number of machine instructions & consequently you would make a tenth the errors.
- High-level languages are more readable & understandable & less likely for error by human being.

Programming Language Evolution

			Machine Code		
			Assembly Language		
Lisp	APL	Algol 60	Fortran	Cobol	
Prolog		Logo	C	PL1	Basic
Concurrent Prolog		Simula 67	Pascal		Fortran 9X
Common Lisp	ADA	Smalltalk		Modula	
Loops	Flavours		C++		Hypertalk
CLOS	SELF	Object Pascal	Java	Object Cobol	Visual Basic
			Objective C		

Fortran is the most popular of the early programming languages. This was aimed at solving problems in engineering & science. Some features include the binary for adding 1 to a series of numbers is meaningless to the non-specialist, the assembler for the same task was shorter, but no clearer, and Fortran shows a major reduction in the number of instructions and was more readable. *More people could now be recruited...meeting an ever increasing demand…*

Tim Bergin, The American University
- *" Prior to Fortran, the number of programmers was relatively limited"*
- Now programmers are not just math graduates
- The minute Fortran was shown to be as efficient as native programming, suddenly it was open to a whole new class of programmers such as mathematicians, scientists, physicists, chemists, and engineers of all kinds.
- Instantly empowered tens of thousands of people
- The computer could now be programmed by people who understood the problem without having to understand computing
- The Business & Government sectors were now looking for a language suitable for their domain.
- By moving to COBOL, a single language for programming all their data processing applications, the US government maintained a coherent common language to transition from older equipment to new with limited disruption.
- Fortran & COBOL are still dominant languages today, but these are just 2 popular languages among thousands which have developed.

Stuart Shapiro, Brunnel University
- *"In the 1960's, as hardware prices were dropping, people started observing that software costs seem to be going ever upward"*

Joyce Currie Little, Towson State University
- *"Spaghetti Code is the colloquial name given to a picture of the structure of a computer program which has bits and pieces everywhere…it becomes just a mass of conglomeration & it looks like a bowl of spaghetti."*
- Some thought new approaches to code production were needed. They suggested that these approaches could only be found in the Math, Science & Engineering disciplines.
- Software Engineering is simply shorthand for "Write – Good – Efficient – Programs; *within budget & on time*."
- There are not yet the techniques to write programs very quickly that will be very efficient & won't have any errors. The techniques simply don't exist yet.
- It certainly has used & benefited from Mathematics, from formalized techniques.
- Today, a new piece of software may have mathematicians & engineers working on it. But the team will also include domain specialists, psychologists, and human-computer interaction experts.

Dennis Cunningham, Lotus Development Corporation
- *"The more diverse a team can be, the better…if you've got the right mix, that's the important thing".*

Jean Sammet, Programming Language Consultant
- *" At the very beginning, software was considered an art, not a science, not engineering and so forth…whether it's engineering or not is a point one could debate ad nausea and I'd just prefer not to debate it."*
- *Is this the end of programming…?*
- One of the notions is once you develop these languages; it would be the end of programming…people would simply walk up to a computer & program it themselves.
- You wouldn't call it programming. They would just use computer and in fact, they would be programming, but it would be so easy that no one would be aware of all this other stuff going on. Well it's a thread that runs through the history and it's never happened.
- There's always a new paradigm, which is going to be the salvation of a human being communicating with a computer to get a task done.
- If anything is learned from the history of computing, it's that some of the most exciting & important developments would come, not from changes to hardware, but from developments in software.
- Although we appear more developed than our predecessors, we are actually standing on the shoulders of giants that slogged through the mud & did things in prior years that make us look good