The Development of the Software Testing in Finland 1950-2000

Pentti A. Pohjolainen

Department of Computer Science, University of Kuopio, P.O. Box 1627, FI-70211 Kuopio, Finland; papohjol@cs.uku.fi

Abstract: The paper presents the development of software testing in Finland. This topic has received little academic attention and it is frequently forgotten. The existing publications concentrate more on the history of machines and programming languages than on the history of the development of testing. The analysis made so far proves that the problems in the early times were very different from nowadays. For example, during the 1950s and 1960s, it was difficult to get computation time for testing. Meanwhile, during the 1990s, and after that, the greatest source of problems has been the complexity and the massiveness of programs. On the other hand, it seems that the education of testing has not been sufficient until the end of 1990s. Hence, the knowledge of diverse testing methods, test automation, and outsourcing are now better than in the past. In our research, we have interviewed over fifty persons. The interviewees vary from pioneers of Finnish computing, having tens of years career, to young professionals of testing. Their selection is from Finnish universities and over twenty companies in Finland.

Keywords: Development of testing, growth of testing, testing methods

1. Introduction

We have found during this research how important and interesting area software testing is. In the beginning of many meetings, the interviewee has said, "I think that I have nothing to tell about testing". After a couple of hours, we see surprisingly that we have discussed all the time just about testing.

We have also found that no one ever researched the development of the software testing in historical perspective in Finland. Outside of Finland, we have found only one article [32], which tries to classify the development of testing in some kind of the stages.

Gelperin and Hetzel [32] have named five stages of the growth of testing: The debugging-oriented period before year 1956, the demonstration-oriented period between years 1957-1978, the destruction-oriented period between years 1979-1982, the evaluation-oriented period between years 1983-1987, and finally the prevention-oriented period after year 1988. This partition received a contradictory reception among the interviewees. Some of the interviewees considered the partition artificial, while others considered it correct.

We know that Martin Campbell-Kelly [30] has written about the early days of debugging, especially about Maurice Wilkes and his works in1940s. In this research we have inspected the development of testing, not the development of debugging, because people consider debugging and testing different things in software development since 1956 [32]. Of course, the very early testing method in Finland was debugging.

We have found also some writings of the development of software in Finland, for example Olli Varho's writing [43]. There are also some discussions about testing, but not about the development of testing. Later, in the 1990s, there were some Master of Science theses published, and SYSTEEMITYÖ [42] - a Finnish journal on software engineering was also founded. We can also mention some writers from that time, for example, Mika Katara, Mitro Kivinen, Erkki Pöyhönen, and Maaret Pyhäjärvi.

We decided to do this research with interviews, because there is no written material on the subject matter, the development of testing, in Finland. We selected the interviewees with a so-called snowball-technique. The first interviewee named two, three possible candidates, who gave new interviewees in turn, and so on. We then chose from these candidates only those, who had an important role in companies and universities, developing and executing software testing. The interviews took place all around Finland. The interviews were based on a questionnaire, free discussions, and recordings and we carried them out in face-to-face fashion.

In the beginning of an interview, we inquired personal data, education, and work experience. Secondly, we inquired when the interviewees had heard first time about software testing, what kind of teaching, books, and articles of testing existed at that time, and how the development of software testing changed toward year 2000 and after. We also asked which programming languages they used, how the programming languages affected the testing, who tested the programs, and how they carried out the tests. We also inquired how they documented testing, and the kind of testing methods used. Furthermore, we asked the interviewees, what they thought about the coverage, outsourcing, and automation of testing. The interviewees mentioned also some books about testing.

In this article, we present based on the gathered material the development of testing in decades from 1950 to the present day. Every decade constitutes a chapter of its own. We should note, however, that the presentation is not exhaustive of all the gathered material.

2. The Time of the Very Early Pioneers 1950-1959

In 1950s only one of the interviewees had heard the word testing – Hans Andersin [2]. Therefore, we base this work on his memories. At first, he remembered working in Sweden on the BESK-computer, which according to his opinion was at that time the fastest one in the world. He remembered a phrase in Swedish "testa programmen". They wrote the programs in machine language or later in hexadecimal format, so there was no need to write long instructions.

When you needed time for testing, you had to wait for your turn. When it was your turn, the program usually ran very unsuccessfully in the beginning. The process could stop anywhere and it would list the contents of certain registers. By investigating these listings, you tried to find out what had happened. When they found the error, the same process started again. The cycle repeated several times before the program was working properly. Testing was primitive. Of course, they had found possible errors visually before testing by a computer. The most effective way to avoid errors was to use pieces of completely tested and properly working programs.

The second computer in his memories was the IBM 650. By then, some systematic ways to find errors were in use. One diagnostic method was a debugging program called DDT, where the computer told all found illogical instructions. The program ran slowly and at the same time, it listed the erroneous instructions.

In those days, the training arranged by IBM for their customers was of good quality. The data processing literature affected somehow to methods of thinking. Development of testing was strong, even exponential, just like in the following decade. Systems designers and programmers did testing. The used programming language did not affect to the willingness to test, but testing was easier using certain languages. They wrote very few documents on testing.

3. The Time of the Early Pioneers 1960-1969

Punched card technology mostly governed the 1960s. Computer time was not available very much and testing was rather difficult. If you did not replace the erroneous card with the corrected one, you had to "batch" the instruction either in octal or hexadecimal code.

Compilation times were long, usually from 30 to 60 minutes. Most of the program listings also produced a listing in machine language (octal, hexadecimal). It was very slow and difficult to find out where the errors were. Nostalgia of 1960s is high because nearly all older ADP-people remember that time. It was the time of data processing old pioneers [12].

3.1 Some Memories from the Infancy

The following are some excerpts from the interviews.

"I was studying in the university, when I took my first course in programming. Linked up with it, there was a rather large practical work we had to do. At that time, nobody cared about the clarity of programs and still less we knew about testing. The process ended up as a long series of trials and errors: trying various input data, using core dumps when looking for bugs, and trying again. To what extent the program became tested, remains a total mystery - and this state of affairs continued to be a prevailing feature for a long period to come.

It was also discovered that people tend to have a natural objection to find errors in their own programs." [1]

"It was in IBM, as I was a student in 1964, when a very unusual case happened. I worked as an operator and read at the same time the manuals of programming. Soon, I coded my own program. After the execution was correct, sometimes came up computational errors, sometimes all was correct. At last, I went to the present professor Markku Nurminen. We inspected the program together and agreed on that the computer operates incidentally wrong. After persuading the system analysts, we consulted the service. They tested the machine and found that there was one loose contact and therefore the result was sometimes right and sometimes wrong." [11]

"When I coded my first program I had no help. There was maybe one page of object code without any symbolic code in use. Of course, I thought the program works, so I threw all my documents into the wastepaper basket and went to the computer. Nearly nothing happened. I had to code the whole program again. One reason to all this was that the manuals were in Danish; I had mistaken some clauses." [15]

"The testing was familiar to me from the beginning of the decade 1960 in Kaapelitehdas in compiling statistical programs. Everybody believed, when coding his first program that it worked properly. There was no chance for a mistake. It was very humiliating to discover that the programs did not work as I had expected. Testing has always been to me going through trial and error. Sometimes when I had a very difficult problem, I had to go to get some advice from my colleagues. At the same time, as I explained my problem, it often happened that I myself solved it before my partner understood anything about it. It is useful to tell someone else about the difficulties; many times it helps you to get the idea of it." [19]

"Already in the basics of informatics was said that you must test - but not exactly how to do it. In practice you learned very soon that the basic things must be done carefully. There was no idea to go to the computer with poorly made program, because the available computation time was so restricted." [22]

"In the beginning, the programs were relative small so that there were no special problems in testing technique. Of course, everybody made errors except Ph.D in Mathematics Jussi Väisälä who coded correct code at the first time. It was very important to be careful in the programming because of the limited computation time." [25]

"First I remember the time, when there was no computer in our company. At that time, Hankkija did already have the kind of a computer that we had ordered. So we travelled 400 kilometres from Kuopio to Helsinki to test our programs in the night." [23]

3.2 The Development, Education, Methods and Literature of Testing

Only one of the interviewees, Hans Andersin [2], held the opinion that the development of testing in this decade was very strong. Probably the reason is that he can compare the decades of the 1950s and the 1960s. A few saw that testing

has been sufficient. Most of the interviewees thought that the development of the software testing at this time was insignificant or missing. However, there was some development. Many researchers talked about the formal verification of programs. On the other hand, if you prove your program correct, someone else can ask, are your proofs correct. No doubt, one step in the development was the creation of the first Nordic computer science professorship in Tampere 1965. From my interviewees, Seppo Mustonen [19] was one of the establishers and Reino Kurki-Suonio [15] the first professor.

There was very little training for testing in the 1960s. The interviewees named only the training of IBM and the internal trainings in some other firms. Reino Kurki-Suonio [15] thinks that there was a lack of test training, because people perceived data processing more of a science than a practice.

The literature consisted mostly of the manuals written by the manufacturers, some programming guides and course publications. The first Finnish book mentioned in these interviews was by Eero Kostamo [37], "Automaattisten tietojenkäsittelysysteemien suunnittelu". In the book, there are surprisingly good instructions to do basic testing. We can see that there are instructions to use test cases - although they are not called "test cases", but the philosophy is the same. In addition, there are instructions to do desk checking and so called automated testing which means here that we use test tape where the program and the test cases are. They had to use the tape for the whole time of testing. Kostamo presents also main rules to document the correction of the errors. Other Finnish books were "Johdatus ohjelmointiin" of Reino Kurki-Suonio [38] and in Finnish translated book "ATK: automaattinen tietojenkäsittely" of Sven R. Hed [34]. In Hed's book, many pages cover testing. It is amazing how little education of testing occurred even in the 1970s, although some material appeared in Finnish already in the 1960s.

3.3 The Effect of the Programming Language, Who Tested and How Much, the Coverage and Documentation of Testing

The selection of the programming languages was, as early as in the 1960s, very wide. The interviewees named, for example, ALGOL, Basic, COBOL, FORTRAN, PL/I, SIMULA, and the assembly languages of the various data machines. The program testability of different languages varied. The more high-level the language was, the easier the testing was. These languages had such features, which guaranteed definite functionality. On the other hand, the programs coded by primitive languages were smaller and therefore easier to control.

Nearly all of the interviewees thought that the programmer himself tested his own programs. According to the interviewees, the time used for testing varied between 10 to 70 per cent from the programming time. Mostly it was between 50 and 70 per cent.

The coverage of testing was nearly an unknown concept. It was enough that the program worked. The documentation, if such existed, contained the storage of the program lists and they wrote it afterwards. Many programmers thought that documentation was unnecessary. One of the interviewees, Pentti Kerola [14],

thought that documentation in this decade was sufficient. Most of the programmers thought that testing was a miserable thing.

4. The Time of the Early Professionals 1970-1979

In this period, testing was becoming easier and computers were becoming faster. They could make amendments to the programs using terminals. Compared with testing today, work was still very slow, but effectiveness was much higher than earlier [12].

4.1 Some Testing Experiences

The following are some excerpts from the interviews.

"I heard about testing when I entered to work in KELA at 1971. In my opinion, it was some kind of detective work. The test material was always too small. Therefore, errors were uncovered in the production run. That produced news in journals. Articles in the newspapers ensured that the same errors were not repeated." [18]

"My career began in Tampella, Tampere, but I heard more about testing in Softplan only after 1975. I remember the tight timetables and that the only chance to test was in the night. I tried to do my work so well that further testing was unnecessary." [21]

"One kind of test was when I started studying in 1972. I had never used a typewriter and I had to seek characters so long that the machine reached a timeout." [27]

"We have always had great hopes for testing to erase errors from programs. Sharers of the honour have always been around when everything was working, but when something went wrong the reason was in the faulty testing." [8]

In my first job in ADP department of a quite big company, practice was that the responsible person, who had changed or repaired a program, was on a standby at home also on weekends through the introduction of the program. I wondered that very much, because I thought that the programs had been tested before introduction so well that the project manager did not need to be on a standby." [3]

"Testing cycle was very slow, maybe two at most three times per day, so it was possible that you left your program for testing, but you got it back the next day. Therefore, desk checking was important. At that time, testing was batch processing, so you had no chance to debug. Tracing errors was very difficult." [4]

"It was not easy matter to estimate the time scale to make a program. Of course, testing was the last job and nearly all reserved time was already used. No small wonder was that we slept beside the computer many nights." [6]

"I remember when I coded my first programs, testing was to read dump and to debug. Overall, it was a boring job." [24]

4.2 Development, Methods, Literature, Coverage and Documentation of Testing; the Effect of the Programming Language, Who Tested It and How Much

Most of the interviewees agreed that the development of testing was slight or satisfying. Desk checking remained as a method. The new thing was that programmers began to think about "limits, loops, and whatnot", as Myers [41] presented in his book. Other writers in this period were for example Codd and Schaefer.[1] Some of the interviewees told that they had used prototypes and test beds.

Very few of the interviewees knew the names of the used methods. However, the methods used in Finland were nearly the same as outside Finland, although they came to use a bit later. Cited as an example, Moore [40] presented the method "State Test" as early as 1956 and Hirsch [35] presented the earliest known description of a software statement and branch coverage analyzer in 1967.

In this decade Systek, the State Computing Center (VTKK), and the Social Insurance Institution of Finland (KELA), developed actively testing in Finland. Education of programming and system design had started in many Finnish universities. Unfortunately, they nearly forgot the role of testing. One exception was ATK-Instituutti (The ADP Institute), where was a seminar of testing in 1970. Based on the seminar, there was a published report "Tietojenkäsittelysysteemin testaaminen" [29] in 1971.

Many of the interviewees saw that the proportion of testing in programming decreased. However, the volume was nearly 50 percent of the programming time. The mentioned languages were Algol 60, Mixal, and Lisp. Note that they published Lisp as early as in 1958. In this decade also designers tested, not only programmers. The programming language still affected testability. The coverage and documentation of testing was poor.

5. The Time of the Professionals 1980-1989

This is the preliminary time of PCs. They tested programs online. They delivered the preliminary tested version to a customer, who made the final testing. This decade started the customer oriented testing [12].

5.1 Beginnings of Testing

The following are some excerpts from the interviews.

"With my hobby, microcomputers, I came across the word testing. At first, I did not understand what it meant. When I coded my first little programs, I found

[1] These author names and those that follow may not be accurate but are based on the interviewees' memories only. Due to illness of the author of the article, we were unable to check the authors and to refer to their relevant publications.

that they were not always correct. Especially when I was young, I thought that testing was very difficult. At last, in the same decade, I made with my partners a test bed for testing programs automatically in C-language. It was quite a novel thing at that time." [5]

"My diploma work was a system for city of Oulu, to provide rented flats. I remember that we tested it very well. However, the end-users were very old people. We found that the usability testing was not sufficient. Generally, I have liked testing." [16]

"In the late 80's the testing of embedded telecommunication software was usually quite on ad-hoc type of work. A very essential improvement for the testing process and practises was proposed by Hannu Honka [6] for Nokia. I had a possibility to contribute to the testing of protocols for the first GSM base station in the world in the beginning of 1990's. Based on that experience, the systematic and automatic software testing approach proved to be an essential improvement for the testing practises." [17]

"I say that my testing was in the beginning quite much of trial and error. I knew no methods or operation models; they came later." [28]

"When I worked in Softplan in Tampere, Pasi Kantelinen wrote there a little guide for testing. In my opinion, these instructions are valid still today. Many times we tested together with the customers and that was productive." [10]

5.2 Development, Methods, Literature, Coverage and Documentation of Testing, the Effect of the Programming Language, Who Tested It and How Much

Most of the interviewees thought that the development in this decade was good or almost satisfying. Now, testers started to design test cases systematically. The V-model came to use as well as the black-box, white-box, top-down, and bottom-up techniques. For example, Harlan Mills [39] presented the top-down technique as early as 1970 in IBM.

They greatly discussed software engineering in the literature in this period. The interviewees remembered authors like Pressman, Sommerville, Gilb, Jackson, and Kaner. It should be noted, however, that for instance, Gilb wrote already in the 1970s.

One thing, which only a few of the interviewees mentioned, was test automation. Mark Fewster and many others worked hard with this problem in the 1980s [31]. We suppose that this was an unknown thing to many testers in Finland.

Very interesting thing was that one of my interviewees [26] got education in testing in the school (Linnanpellon lukio, in Kuopio) in the 1980s. That was exceptional.

As new languages, interviewees remembered the publication of ADA, C, C++, and Pascal. Note that C and Pascal appeared in the 1970s. In the 1980s, they mentioned testing groups for the first time in these interviews. Based on the interviews, it is easy to see that there was some development in testing. The

coverage and documentation of testing became a little better. The proportion of testing was between 30 and 50 per cent of programming time.

6. The Time of the Young Professionals 1990-1999

During this period, customers learned to use PCs. They tested the programs as before. Computers or terminals were on user's desk, and users knew what they really wanted. Computers were becoming more and more effective. Programming and testing was becoming easier, because of the new programming languages and macro-oriented languages. In fact, most testing was customer oriented. Customers received "almost a complete system" and started to use it. In guarantee time, the customer told about program errors that they had found. Programmers corrected the errors and they delivered the new version to the customer. The programs were larger than before and they were more complex, which was a problem [12].

6.1 Happenings and Memories between the Years 1990 and 1999

As an example, one of my interviewees, Juha Itkonen [9] became familiar with testing for the first time in Helsinki University of Technology, where courses on testing appeared. They had a testing course together with University of Helsinki already at the end of the decade.

"I got practical experience as an assistant researcher in VTT. Then, the eyes-opening-experience was that everything did not work in practice, although I had tested it well in my own opinion. It was enough that, when other people in VTT started to experiment with the program, errors appeared. I was aware of that testing is something else than just experimentation." [9]

"I remember that I was studying on a software engineering course and there was quite a lot work to be done. We had programs, which we had to test. There was CTC (Coverage Analyzer for Testing programs in C-language) indicator connected to the programs. Passing the task (obtaining a high enough coverage) was far from trivial." [13]

"When I began to test, I used to do it according to given instructions. In other words, I stared into the screen from day to day. The idea, what testing was, sprang up, when I myself picked up the baton what I tested and how. The point of view that the experienced tester could be better than a novice is completely wrong. The own capacity of the people is always higher than guidelines which somebody else has dictated. I suggest all to do instructions of their own and after that to ask comments from somebody else to them." [20]

Additionally, another interviewee [7] thought that too many rules, how to do testing, is not good.

6.2 Development, Methods, Literature, Coverage and Documentation of Testing, the Effect of the Programming Language, Who Tested It and How Much

The development of testing in this decade was good, some say even excellent. New methods were for instance the management of testing and quality management. Notable was that there were now some real attempts to automate testing in Finland. For instance, Edward Kit [36] said, "The time for test automation is now".

In the literature, the most notable Finnish book was Ilkka Haikala's and Jukka Märijärvi's "Ohjelmistotuotanto" [33] in 1995. In this book, there was an entire chapter dedicated for testing. Software engineering was discussed much in the literature in this period. The interviewees remembered authors like Kit, Beizer, Hetzell, Marick, Gilb, Whittaker, Fewster, Graham, Bach, Pettichord, Whittaker, Dustin, Rashka, Paul, and Kooman.

Testing groups began to work, particularly in big companies. Java, Visual Basic, and Python were some of the new languages introduced in this period. Testing was 50 per cent from the programming work. Documentation and the coverage of testing developed a little.

7. The Time from the Year 2000 until Today

This is the time of PC computers, used as stand-alones or as terminals to the central computer. There is no lack of computer time. They could test programs easily online. The customer's word was the final acceptance of the new computer system [12].

Many interviewees thought that the development of testing has been now very strong. Automation has developed furthermore. A quite new activity, outsourcing of testing, has increased exponentially. There are in Finland software houses that develop almost only outsourced testing. Also, Test Driven Development (TDD) and agile programming take root - testing is no longer only the last stage in programming. In TDD, the tests derived from the requirements, and only after they satisfied the requirements did the coding begin.

Although almost all think that development goes ahead just right, there are also different opinions. For example, Eerola [3] thinks that we tested before the millennium so much that after it, everybody thought that testing was complete and the correctness of programs was on a sufficient level. They forgot that testing is a continuous and evolving process and hence testing decreased catastrophically to Eerola's mind. The same thing has happened with other approved practical methods and ways of action, too. We do the good thing once and then we forget it. Another reason, which has decreased testing, is the downtrend of the software industry, which wants to economize on testing costs and to shorten time to market programs.

8. Conclusion

In this paper, we argued that the various testing methods had been available years before they came in use in Finland. Many methods had been in use, but the interviewees disagree about, when the methods came to use in Finland.

We know that there are some publications about testing published already in the 1950s outside of Finland. The quantity of literature has increased every decade.

We argue that we waked up a little late in Finland. In the year 1958, when we got our first computer, it is my opinion that we trailed maybe ten years behind the global state of development of the software testing. In addition, that was nine years after Wilkes [44] gained his historical insight in 1949 that "a good part of the remainder of my life was going to be spent in finding errors in my own programs". Now the situation is much better. We can say with a good reason that we have reached the global state of development.

Many of the interviewees think that TDD is the greatest revolution since object-oriented programming. Testing coverage increases little by little along with the documentation. Nowadays, testing and testers have respect for example in Qentinel Oy, which is one of the leading companies of outsourced testing in Finland.

Worth noticing is that the first testing experiences of almost all interviewed have happened through trial and error. Although training has increased remarkably in the decades 1990 and 2000, it does not reach beginner testers. Could we think that we should introduce testing more systematically already in the first programming course?

Acknowledgments

I thank warmly all interviewees for their time, interesting meetings, and their employers for great flexibility. Special thanks to the Department of Computer Science, University of Kuopio, which sponsored this research. I wish thank PhD Mauno Rönkkö for his reading and comments. I also wish thank PhD Petri Paju for his good literature references.

The interviewees in alphabetical order: Alakangas Tarja, Alanko Timo, Andersin Hans, Andersson Thorbjörn, Eerola Anne, Eriksson Trygve, Haglund Henry, Haikala Ilkka, Hannula Esko, Honka Hannu, Honkasaari Terttu, Hotti Virpi, Hyvönen Mervi, Itkonen Juha, Jantunen Pekka, Jokiharju Tuula, Järvi Timo, Karhula Tarmo, Kaseniemi Mauno, Katara Mika, Kerola Pentti, Kinnunen Pirkko, Kivinen Mitro, Kurki-Suonio Reino, Käckman Tarja, Laiho Matti, Lappalainen Vesa, Latvakoski Juhani, Lehtinen Heli, Liinanto Erkki, Mustonen Seppo, Poutanen Olavi, Pyhäjärvi Maaret, Pääkkönen Tuula, Pöyhönen Erkki, Roine Kirsti, Röyskö Kirsti, Sakkinen Markku, Stenius Mårten, Sulonen Reijo, Tarnanen Pentti, Tervonen Ilkka, Tienari Martti, Toroi Hannele, Toroi Tanja, Torvinen Seppo, Tukiainen Petteri, Törn Aimo, Vanhatalo Hilkka, Verkamo Inkeri, Yksjärvi Jorma

The employers in alphabetical order: Avain Technologies Oy, Bestsel Oy, City of Turku, Enfo Partner Oy, Financium Oy, Finish Tax Administration, F-Secure Oyj, Fujitsu Services Oy, GE Healthcare, Haaga-Helia University of applied sciences, Haglund Networks Ltd, Helsinki University of Technology, Nokia Mobile Phones, Nokia Multimedia Computers, Nokia Technology Platforms, Qentinel Oy, Social Insurance Institution, State Technical Research Centre, SYSOPENDIGIA Oyj, Tampere University of Technology, Testwell Oy, TietoEnator Oyj, TKP Tieto Oy, TS-Yhtymä Oy, University of Helsinki, University of Jyväskylä, University of Kuopio, University of Oulu, Åbo Academy University.

References

Interviews

[1] Alanko, Timo, University of Helsinki
[2] Andersin, Hans, Helsinki University of Technology
[3] Eerola, Anne, University of Kuopio
[4] Haikala, Ilkka, Tampere University of Technology
[5] Hannula, Esko, Qentinel Oy, Espoo
[6] Honka, Hannu, State Technical Research Centre, Oulu
[7] Hotti, Virpi, University of Kuopio
[8] Hyvönen, Mervi, Social Insurance Institution, Helsinki
[9] Itkonen, Juha, SoberIT, Helsinki University of Technology
[10] Jokiharju, Tuula, Nokia Multimedia Computers, Tampere
[11] Järvi, Timo, University of Turku
[12] Karhula, Tarmo, Pensioner, Kuopio
[13] Katara, Mika, Tampere University of Technology
[14] Kerola, Pentti, University of Oulu
[15] Kurki-Suonio, Reino, Tampere University of Technology
[16] Käckman, Tarja, Nokia Mobile Phones, Oulu
[17] Latvakoski, Juhani, State Technical Research Centre, Oulu
[18] Liinanto, Erkki, Social Insurance Institution, Helsinki
[19] Mustonen, Seppo, University of Helsinki
[20] Pyhäjärvi, Maaret, F-Secure Oyj, Helsinki
[21] Roine, Kirsti, Bestsel Oy, Tampere
[22] Sulonen, Reijo, Helsinki University of Technology
[23] Tarnanen, Pentti, Enfo Partner Oy, Kuopio
[24] Tervonen, Ilkka, University of Oulu
[25] Tienari, Martti, University of Helsinki
[26] Toroi, Hannele, GE Healthcare, Kuopio
[27] Vanhatalo, Hilkka, Fujitsu Services Oy, Helsinki
[28] Yksjärvi, Jorma, SYSOPENDIGIA Oyj, Helsinki

Literature

[29] ATK-Instituutti: Tietojenkäsittelysysteemin testaaminen (The Testing of Data Processing System). ATK-Instituutin kannatusyhdistys, Helsinki, 1971.
[30] Campbell-Kelly, Martin, The Airy Tape, An Early Chapter in the History of Debugging. *IEEE Annals of the History of Computing*. Vol.14, No. 4, 1992.

[31] Fewster, M., Graham, D.: *Software Test Automation. Effective use of test execution tools*. ACM Press, New York, 1999.
[32] Gelperin, David, Hetzel Bill, The Growth of Software Testing. *Communications of the ACM*, Volume 31, Number 6, June 1988, 687–695.
[33] Haikala, Ilkka, Märijärvi J.: Ohjelmistotuotanto (Software Engiueering). Suomen atk-kustannus, Espoo, 1995
[34] Hed, S. R.: ATK: automaattinen tietojenkäsittely (ADP-Automated Data Processing). Tammi, Helsinki, 1966
[35] Hirsch, I. N.: MEMMAP/360. Report TR P-1168, IBM Systems Development Division, Product Test Laboratories, Poughkeepsie, New York, 1967.
[36] Kit E.: Software Testing in The Real World, Addison Wesley, 1995.
[37] Kostamo, E. (ed.), Automaattisten tietojenkäsittelysysteemien suunnittelu (The Engineering of the Automated Data Processing Systems). Systemisuunnittelukurssin opettajakunta, Helsinki, 1963.
[38] Kurki-Suonio R.: Johdatus ohjelmointiin (An Introduction to Programming). Tampereen yliopiston tutkimuslaitos, Tampere, 1966.
[39] Mills, Harlan: Software Productivity, Dorset House, June 1988.
[40] Moore, E. F.: Gedanken-Experiments on Sequential Machines, in Automata Studies, Annals of Mathematics Studies, No. 34, pp. 129-153, Princeton University Press, Princeton, N. J., 1956.
[41] Myers, Glenford, The Art of Software Testing, John Wiley & Sons, New York, 1979.
[42] Systeemityöyhdistys SYTYKE: SYSTEEMITYÖ. Sytyke ry, jäsenlehti. Helsinki, 1993.
[43] Varho, O.: Automaattisten laskukoneiden yleispiirteet. Teknillinen aikakauslehti 2/1959, 25-29.
[44] Wilkes, M.: Memoirs of a Computer Pioneer. MIT Press, 1985.