ALGOL-GENIUS

An Early Success for High-level Languages

Bengt Asker

Formerly employee of Datasaab and Ericsson; Bengt.Asker@comhem.se

Abstract: Algol-Genius, an Algol 60 implementation with features from COBOL, was the brainchild of Börje Langefors. In 1964, assembler was the dominant programming language, but Algol-Genius broke that trend among Datasaab D21 customers. Algol-Genius programs were still in production in the late nineties.

Key words: Algol, Datasaab computers, programming languages

1. INTRODUCTION

The main topic of this paper is the programming language Algol-Genius. Since the history of the first Datasaab computers, D21 and D22 had little representation at this conference, I will place Algol-Genius in the context of Datasaab's early history.

2. HOW DATASAAB WAS BORN

Saab was one of the first organizations in Sweden to use computers on a large scale. Aircraft design required extensive computing. Initially, women did this on desk calculators. Börje Langefors pioneered the of use punch card machines for matrix calculations. When Besk became available, Saab was one of the heavy users and soon built a copy, called Sara. These efforts meant that Saab early on acquired competence in software development. On a parallel line, in other parts of Saab, Viggo Wentzel designed a transistorized digital computer, aiming at an airborne version. The result

was a prototype, one of the first transistorized computers in the world, called D2, which he demonstrated to the Swedish air force in 1960. It was a desktop computer; in fact, it covered the entire desktop. See Figure 1.

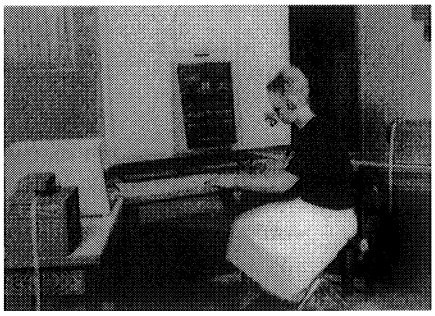

Figure 1. The D2 desktop computer

Saab had attempted to diversify from a total dependence on military products. Now it could use this unique combined expertise in hardware and software to launch a commercial computer, the D21. Gunnar Lindström was the typical entrepreneur heading the division. Quite naturally, Viggo Wentzel took care of the hardware and Börje Langefors the software. Saab's CEO Tryggve Holm, one the last patriarchs, said in a speech to prospective customers: "We woke up one day at Saab and found that we made computers"

The first customer was Skandinaviska Elverk, not surprising since Gunnar Lindström came to Saab from the power utility sector. The first order, shown in Figure 2, has become quite famous. It is a three-line letter, dated 14 December 1960, ordering a computer according to the offer so and so.

3. DATASAAB D21

3.1 Hardware

The first delivery of the D21 occurred in 1962. It was a very competitive computer for its time. Maximum primary storage was 32K words (96KB) with a cycle time of 4.8 μs. The addition of two 24-bit integers took 9.6 μs. Secondary storage was tape, an inventive construction that originally was developed for Sara with the appropriate name Saraband. Based on a suggestion from Börje Langefors, the designer Kurt Widin invented a coding, which he later found out was called a Hamming code, that corrected

one bit and detected two-bit errors. Searching for a block, reading, and writing occurred in parallel with processes in the CPU. This was a novelty in that price class and contributed strongly to the excellent performance of D21. Basic input and output was, in conformance with the Nordic tradition, punched tape, with a very fast tape reader from Regnesentralen. Since they designed the D2 for process control, it had an interrupt function that carried over to D21.

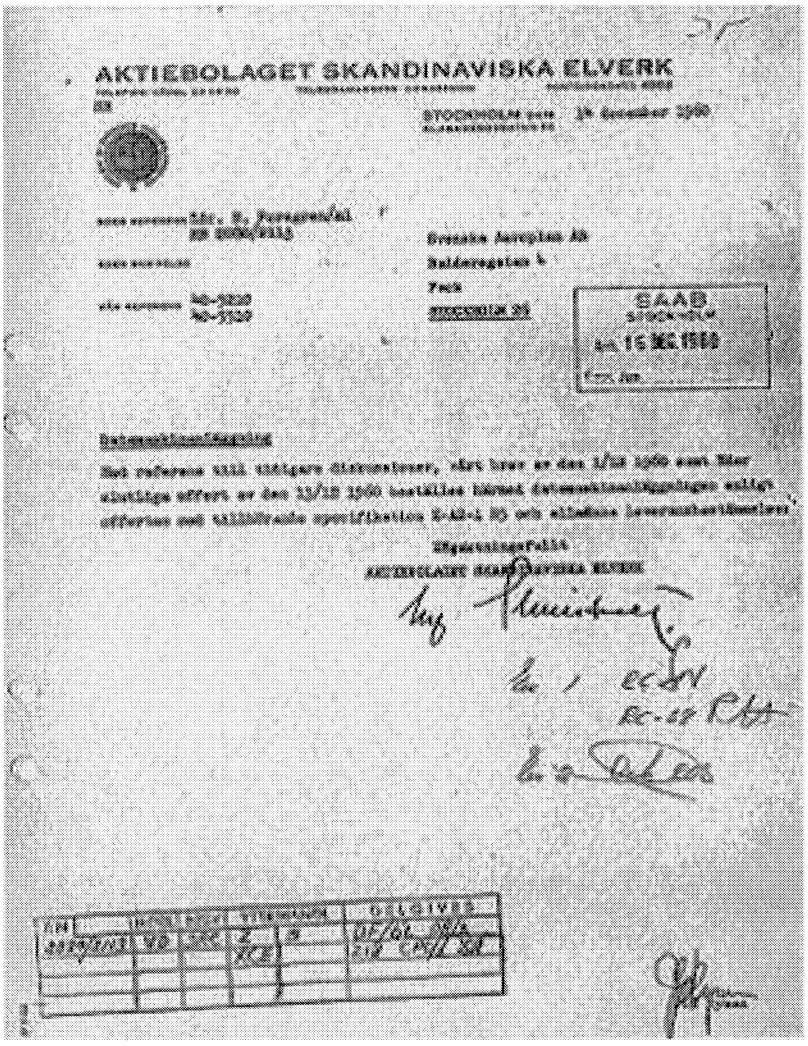

Figure 2. The first D21 contract

3.2 Market

The computer was a reasonable success. In all, they manufactured 32 computers, mostly sold in Scandinavia, a few also in Czechoslovakia and Hungary. Some early customers after Skandinaviska Elverk were Gothenburg University, Slakteriförbundet, Kockums shipyard, Volvo Flygmotor, Allmänna Brand (insurance), Saab and SMHI (weather forecasts). This was an interesting mixture and shows that the D21 was a truly general computer. With the exception of SMHI and Gothenburg University that had extremely demanding computations, they all used Algol-Genius extensively.

The biggest order came from the county governments (Länsstyrelserna). At first, orders for some twenty computers went to IBM and its 360/30. After heavy lobbying, they changed the decision so that IBM received only half of the orders and Datasaab the other half. Since the two computer brands were incompatible in every respect, this meant that two parallel software development efforts had taken place for the Länsstyrelserna. Today, this would be unthinkable; at that time, however, it was not a unique compromise. The companies delivered the computers between 1963 and 1966. However, in 1969 the parliament decided that the counties should replace the IBM computers with D21s, because of their superior performance, mainly in sorting. In the end, they delivered eight D21 and seven D220. The Länsstyrelserna did not use Algol-Genius for two reasons: Application development started before the compiler was available and the computers had a much-squeezed memory.

3.3 Other software than Algol-Genius

The interrupts in D21 were never used for true multiprocessing, only to run I/O in parallel with computing. Therefore, the "operating system" occupied a tiny 1024 words in primary memory. In addition, "Dirigenten" (the conductor) translated job control statements into a very compact representation and proved to be very efficient and flexible.

The name of the assembler was DAC. As a protest against the very stereotyped punched card tradition, it sported special features such as a free format, use of capital and small letters, the use of + instead of ADD, the use of C+ for LOAD.

The D21 excelled in sorting. The tape system with parallelism and the comparatively big memory contributed of course, but also the advanced sorting algorithms used. At this time, Donald Knuth and others had just published their pioneering research and Åke Fager did a great job in implementing the results for D21.

4. ALGOL-GENIUS

4.1 4.1 Accidental problems

Fred Brooks' first published *The Mythical Man-month* in 1975. He based the book on his experience as a project manager for IBM's OS/360. *The Mythical Man-month* is mandatory reading for anyone interested in computer history and is now available in a new edition, which also contains *No Silver Bullet.* In this essay, Brooks divides the difficulties in software design in essential and accidental. His point is that we will never find the silver bullet that kills the essential difficulties but that we have made great progress eliminating the accidental. One example of essential problems is the inherent complexity that "software entities are more complex for their size than perhaps any other human construct, because no two parts are alike." Another is the necessity to conform to and implement rules and regulations "forced without rhyme and reason by the many human institutions and systems to which his interfaces must conform." Accidental problems are those that are inherent in the "machine programs concerned with bits, registers, conditions...." The first step to eliminate the accidental problems was assemblers; then came the problem-oriented languages with FORTRAN leading the way. Algol and of course Algol-Genius came soon after. I would claim that Algol-Genius went as far as you could reasonably go to eliminate the accidental problems with the machine resources then available. Today's software world with so many complexities such as the web and distributed components requires much more powerful tools. That is another story to which I will return. The essential problems we still have to live with!

4.2 The first version

In 1962, the dominating language for business data processing was assembler, in particular in the IBM world. Börje Langefors was convinced that we could and should use high-level languages. Algol 60 was already well established at that time and they defined the first version of COBOL. Since D21 had no decimal arithmetic, it was judged that performance in a COBOL implementation would not be acceptable. The decision to choose Algol was because early COBOL was not a good programming language. It had a very primitive procedure concept with no parameters; all variables were global and it featured some dangerous constructs like COMPUTED GO TO and ALTER. A syntax that prescribed "ADD A TO B GIVING C" instead of "C := A + B;" did not make things better. (COBOL has evolved since then!) However, COBOL had a feature that was missing in Algol—it

could handle records. So, Langefors suggested that we add a record and input/output part to Algol, modeled after COBOL. The addition was called Genius (Generellt in ut system).

Ingemar Dahlstrand had then implemented Algol for the BESK and the Facit EDB in a pioneering work. We used the experience from that implementation and engaged one the designers Sture Laryd. He worked together with Gunnar Ehrling, a pioneer from Matematikmaskinnämnden and so we were able to get a good, efficient, and reliable implementation of the Algol part. The specification was compatible with Algol 60 with few exceptions. Consult Ingemar Dahlstrand's paper on the implementation of Algol 60 for further details on that subject.

Based on the ideas from Börje Langefors, Gunnar Hellström and the author made the detailed specification and supervised the implementation of the Genius part. Data declared in Genius were globally accessible. The syntax was rather faithful to COBOL except that it used underlining, like in Algol, instead of reserved words. Looking back, at that time, the fact that in Algol you could use any names for your variables was a great advantage in the view of many. However, all modern languages use reserved words and no one is complaining. The record concept was mostly used to model and format data on peripheral equipment; it was not as flexible as in later languages like Pascal.

Procedure calls replaced the verbs in COBOL since they were part of the Algol program. One could perform operations such as open and close files and read, search, and write records as one would expect.

4.3 Applications

The first version of Algol-Genius was released in 1964 and it soon became evident that the language, as well as its implementation, was well suited for a wide range of applications. Algol-Genius became the preferred programming language for most of our customers. This was true both for technical and administrative applications. It turned out that for specific cases, only assembler could substantially improve performance. Let me just mention one example. Kockums shipyard was then the leading builder of huge tankers in the world. Under the leadership of Kai Holmgren, it introduced CAD/CAM methods early on. With Algol-Genius as a base, the developed a language and a system called Styrbjörn/Stearbear for the entire design process. Kockums shipyard is long gone, but Stearbear still lives on, now under the name of Tribon. The software it uses had its beginnings from D21 and Algol-Genius way back, of course. Nevertheless, Tribon is still a very successful, although not a very well known Swedish software company.

4.4 Extensions

Datasaab had a very active user club and one favorite subject of that club was extensions to Algol-Genius. Since we owned the language, the formalities were minimal. If enough customers wanted an addition and we deemed it reasonable, we would implement it. One feature, which I still have bad feelings about, concerned the use of punched tape. Algol-Genius already had very powerful procedures for that. However, we got a request for further automation from Industridata, who used D21s and later D22s in their service bureaus. They had to accept and process tapes coming from machines such as cash registers with many different formats. Together with them, we designed a table driven method to handle this, but it was not very successful and surely not worth the effort. Nevertheless, it is a good example of the close cooperation we had with the customers.

5. DATASAAB D22

5.1 Hardware

1968 marked the launching of the D22. It was a much more advanced computer with protected mode for the operating system, ability to address 256K word (768KB) of memory with a cycle of 1.6μs. Floating-point arithmetic was part of the hardware. Professor Germund Dahlquist was a consultant on the project and did pioneering work in specifying a new arithmetic with a minimum of rounding errors. The D22 could manage character handling and decimal arithmetic to obtain efficient execution of COBOL programs. Quite an effort went into making the decimal hardware as COBOL-compatible as possible. We could have all of these improvements without breaking the compatibility with D21. Existing binary D21 programs could run on the D22 without problems.

The D22 could support modern and extended peripherals. The most important addition was the inclusion of disk memories with removable disk packs. The tape standard changed to be compatible with the market (read IBM) at that time, while still being program compatible with the D21. A front-end computer handled the D22's communication with the outside world; it was Datasaab's own 2100. That same computer also played a role the ATM described by Viggo Wentzel; consult his paper on the topic in these Proceedings.

5.2 Market

D22 was of course the natural choice for the D21 customers. But sales were not limited to those, all in all some 70 systems were delivered up and until 1975. However 70 was a way to small number to create any interest in the independent software vendors that begun to emerge at this time. They picked of course IBM machines as their target. For this reason, the first IBM-compatible machines, exemplified by Amdahl, made their entrance at this time. Datasaab D23 was an attempt to build a computer that could emulate D22 as well as IBM/360. But time run out for this project, it was in fact doomed from the beginning.

5.3 Other software than Algol-Genius

The major software effort for D22 was an operating system. It inherited the name Dirigenten, but was on a very different scale, with true multiprocessing facilities. The work was lead by Jan Nordling. Dirigenten was optimized for batch processing and was extended for time-sharing later on. Jan Nordling and his colleagues had taken part in the design of the hardware that consequently was well suited to multiprocessing, but there was still a lot of new ground to be covered.

As indicated above, D22 could handle a COBOL as well as a FORTRAN implementation. Good and efficient but not widely used, Algol-Genius continued to be the preferred language.

Datasaab developed applications in some targeted areas, mainly health care and logistics. They appropriately named the logistic system "Lagom." It was quite successful and survived the D22 by being converted to Univac 1100 and to Unix.

5.4 Algol-Genius

Since D22 was compatible with D21, conversion was no problem. The Algol-Genius language required a few additions, the major one being indexed-sequential files for the disk memories. Still, substantial work had taken place on the compiler to take advantage of the new features in the hardware.

6. UNIVAC 1100

6.1 Background

The small number of computers we managed to sell and the consequent lack of interest from software vendors killed Datasaab's mainframe line (tunga linjen in Swedish). A deal was made with Sperry Rand, who acquired the D21, D22 and D23 lines, resulting in a joint venture, called Saab-Univac, formed in 1975 The goal was to convert as much as possible of the customer base to Univac computers, specifically Univac 1100.

6.2 Algol-Genius

We soon decided that the best way to move the existing customers to the Univac 1100 was to implement Algol-Genius for that computer. Once more, we had an Algol implementation that we could use, this time already running on 1100. It came from the University of Trondheim (sic!) and required very few changes and additions. We had to build the Genius part from scratch. Since it was quite dependent on the operating system and the file system, we implemented it in close cooperation with Univac. For that purpose, Hans and Christel Ljungren spent a winter in Minneapolis/St Paul.

It was not possible to achieve 100% compatibility because of differences in the hardware as well as in the operating system, but it turned out that the conversion was painless, compared to other alternatives. Algol-Genius contributed no doubt to the fact that Saab Univac lost only few Datasaab customers to other vendors.

Saab-Univac did not commit itself to develop Algol-Genius and furthermore, it was after all a proprietary language. The customers thus gradually converted to other languages. Even so, Algol-Genius enjoyed a long life. Even at the end of the nineties, some companies were using Algol-Genius programs in their production efforts; the subject of necessary ongoing maintenance contributed to its final extinction. As far as the author knows, none has survived the millennium.

6.3 What can we learn?

Software is expensive to develop but cheap to produce, copy, and distribute. This means that volume is crucial for economic well being. With large volume, a vendor has a great freedom in setting prices on his software. Maybe even more important than the price advantage is the fact the large volume attracts other players. It generates interest to do other creative things such as developing add on products and specializing in consulting and

training for that software, which in turn makes it even more attractive to customers. Consequently, in each specific software area, there will always be a gorilla and a few chimpanzees [Moore]. Datasaab not only tried to build and sell software, it did so with software that could only run on its own, very proprietary, platform. Moreover, this was in a language area and a geographic market that was minimal. In hindsight, the result was inevitable.

If it were difficult then, today it would be almost impossible. Operating systems, middleware, and development environments run to millions and millions lines of code. The corresponding D22 software was tiny! So now, we have one gorilla (Microsoft) and a few chimpanzees (the Unix/Linux companies). By using one of these platforms and specializing in a specific software area, it is possible to succeed. In other words, let others solve the accidental problems and focus on the essential ones. I have already mentioned Tribon; another example is Opera from Norway and doubtless, there are many others, unknown to me. (Linux, by the way, solves the accidental problems. The invention is a fascinating and novel business idea. Can you, by the way, call "giving away for free" a business idea?)

The area of embedded systems is also very interesting. Whether it is mobile phones, process control, or car and engine control, it starts out as a very proprietary product. Only the hardware vendor has access to the interface specifications and is interested in developing the software. But gradually, as the applications grow, the need for other sources arises, as well as the interest from software vendors to contribute. Standards evolve that make this possible. So far, we in Scandinavia have been quite successful in embedded systems but the fight is not over yet. We hope that Symbian, backed by Nokia and Ericsson-Sony, will become a force in the mobile phone market.

REFERENCES

[Brooks] Fredrick Brooks, *The mythical man-month*, Addison Wesley 1995
[Mooore] Geoffrey Moore *Inside the tornado*, HarperBusiness 1995
[Wentzel, et al] Tema D21, *Datasaabs vänner*, 1994
[Yngvell, et al] Tema D22-D23, *Datasaabs vänner*, 1997