Unraveling Algol: US, Europe, and the Creation of a Programming Language

David Nofre
University of Amsterdam

Current views on the programming language Algol assume its European origins. However, the inability to exchange information between computers affected both sides of the Atlantic. Whereas Algol promoters sought to create one universal programming language, other approaches sought to preserve a variety of languages and create a general translation system. Therefore, the polarity was not between programming languages, but uniformity versus diversity.

Computer programs are often regarded as elusive objects, surrounded by an aura of abstractness, immateriality, and complexity. This perception affects programming languages as well. Regarded as the skeleton of software, such human-made languages were first developed in the late 1950s to facilitate the process of writing down programs.[1] Since then, numerous programming languages have been developed. Their proliferation and subsequent evolution into sophisticated technological artifacts, however, might mask that they are the outcome of complex processes of definition and negotiation carried out within and between different kinds of institutions and computing communities. Programming languages, therefore, are cultural and social endeavors, with a social and even a material existence.[2]

So far, the history of programming languages has focused on the technical characteristics and development of the different languages.[3] These accounts often mirror an evolutionary approach with the distinctive genealogical tree as an iconic image.[4] In the specific case of the programming language Algol (for algorithmic language), this evolutionary image is often accompanied by references to a situation of Babel-like confusion with languages as a result of the inability of the different computing communities to agree on a single universal programming language.[5]

In this article, I discuss the early history of Algol. This programming language was the product of a collaboration between representatives of American computing organizations and industry and West European computing centers. In the late 1950s, the fast proliferation of computers in the US and the construction of the first electronic computers in Europe increased the need to exchange experiences and techniques between computing centers. They felt this need on both sides of the Atlantic, and several meetings took place in 1958 and 1959 between representatives of American computing organizations led by the Association for Computing Machinery (ACM) and a West German-Swiss committee established with the support of the German Society for Applied Mathematics and Mechanics (GAMM). Their goal was to define a ''universal programming language''—that is, a programming language to be used in all kinds of electronic computers and for all purposes.[6] The outcome of these meetings was a first definition of the International Algebraic Language, soon renamed Algol.

For some of these parties, Algol was meant to be the universal programming language, at least for scientific computations. This perception was especially strong in Europe. Indeed, the new language spread quickly all over Europe, especially in Central Europe, the Low Countries, and Scandinavia after the publication of the first official report with the language's specifications in May 1960.[7] However, Algol did not arouse the same expectations in the US. Although the ACM, IBM, and Share (IBM's users group) had initially supported Algol's development, at

 Published by the IEEE Computer Society 1058-6180/10/$26.00

some stage in 1960 these American organizations gradually withdrew their support of the language as the standard for all computing purposes.[8] Their decision was perceived as a major obstacle to Algol's success and part of a strategy to favor the more commercially oriented IBM language Fortran. As a result, bitter debates followed about the supposed qualities or shortcomings of both programming languages.

Ruthlessly summarized, this is the established narrative of Algol. This narrative is based on a handful of participants' accounts and anchored in the American–European and academic–commercial polarities.[9] Contrary to this narrative, I argue that conflicts around Algol were rooted in an earlier, deeper dilemma—namely, to preserve the diversity of programming languages or to adopt a single universal programming language. Moreover, this dilemma was strongly related to the need to find an efficient method to advance the transportability of programs.

In the late 1950s, two competing solutions appeared to offer an answer to this problem. The first solution was to create a single universal programming language such as Algol (or indeed any standardized Fortran-like language). The second option was to keep the multiplicity of programming languages and develop an effective general system of translation. In essence, this was Share's project to create the Universal Computer-Oriented Language (or UNCOL). Therefore, the opposition was not between Fortran and Algol—or any other universal programming language candidate—but between diversity or uniformity, between preserving the variety of programming languages or adopting a universal programming language.

This article is not a comprehensive history of Algol. Instead, I have focused on the early period in its development (1955–1960). It was in these years that the dilemma between diversity and uniformity was most clearly expressed. And although later developments might require a different narrative, this dilemma was still at the core of many debates around Algol and the convenience (or not) of a universal programming language. Additionally, this choice to focus on the early period lets me show how the image of abstractness and Europeanness that surrounds Algol is in fact a post-hoc construct.

The images of Algol

In November 1969, Robert Bemer published an article in the journal *Annual Review in Automatic Programming* with the explicit and provocative title ''A Politico-Social History of Algol.''[10] In his article Bemer, a former researcher at IBM and active participant in Algol's development in the US, reported the changing attitudes of the American actors involved in the process of defining Algol—namely, IBM, Share, and the ACM. The article is basically a collection of fragments of private letters, articles, reports, and minutes of conferences that chronicle the making of Algol. The article also contains a short introduction with veiled references to IBM's role, unmentioned political factors, and the inevitable comparison with Fortran. In this sense, Bemer's article is paradigmatic of the way the history of Algol has been written so far: a pioneer's account avoiding, unlike the title suggests, any clear political and social factor.

At the same time, when Algol was spreading quickly through Europe, bitter debates aroused about its qualities. These debates revolved around two sets of features that supposedly characterized the language. On the one hand, those who were in favor of the new language stressed its elegance, beauty, simplicity, completeness, grace, and power. On the other hand, those who disliked it pointed out its academic origins, which supposedly made it unsuitable for business users, and stressed its abstractness, complexity, inflexibility, and inefficiency.[11] All together, they contributed to create the image of elegance and mathematical abstractness attributed to Algol, whether considered a positive or negative feature.[12]

Indeed, this image has followed the language until the present. In 2005, the ACM awarded Danish computer scientist Peter Naur with the prestigious Turing Award for his life contributions to computer science. During the ceremony of the prize, Intel senior fellow Justin Rattner declared that Naur's award was meant to encourage future language designers to achieve the ''elegance and simplicity'' of Algol:

> Over the years, programming languages have become bloated with features and functions that have made them more difficult to learn and less effective. This award should encourage future language designers who are addressing today's biggest programming challenges, such as general-purpose, multi-threaded computation, to achieve that same level of elegance and simplicity that was the hallmark of ALGOL 60.[13]

This image of Algol has also been reinforced in the retrospective accounts of the actors who took part in the development of this programming language. In particular, Alan Perlis and Peter Naur's articles on Algol in 1978 significantly consolidated this image.[14] In his paper, American Alan Perlis considered Algol to be ''an object of stunning beauty,''[15] which he attributed to the European ''general passion for orderliness.''[16] For Perlis, however, these ''European'' features were far away from the concerns of the American computing centers, which thought of Algol as a ''welcome abstraction'' but suitable only for publishing technical issues in the specialized literature.[17] Algol thus came to be seen in comparison with IBM's Fortran, which supposedly was a programming language that ''fitted well into the needs of the customers.''[18]

The Naur and Perlis papers reinforced the American-European dualism present in most literature on Algol. The panel session on Algol of the ACM conference included the American Perlis and Danish Naur to reflect on ''the European and the American side of the development of Algol.''[19] The Algol session was the only one scheduled in such a way during the conference. This intended dualism contributed to a great extent to the general acceptance to the idea that Algol had been a European endeavor eventually rejected by the Americans.

This American-European polarity is further reinforced in the accounts of the European participants,[20] who often mention the work of the West German and Swiss computing groups in the early 1950s as the starting point of Algol. They also tend to emphasize the international character of Algol. Hence, these accounts give much attention the international character of the meetings; the support received from UNESCO and from national computing associations in the UK, France, and the Netherlands; and the spread of the implementation of the new language. Finally, the European pioneers' accounts also contain more or less explicit references to the reluctance of the American organizations to embrace and sustain Algol's implementation in the US.

This image of Algol as a European, elegant, abstract, and therefore inefficient programming language is a historical construct that emerged following the modest spread of Algol in the US in the early 1960s. I argue, however, that the transatlantic collaboration that would lead to the making of Algol was born from a shared concern within American and European computing centers: the need to foster information exchange between computing centers. They also agreed on the use of mathematical notation to write computer programs.

''As close as possible to ordinary mathematical notation''

The first attempts to automate programming in the US date from the early 1950s. These efforts were driven by a labor and economic rationale: the need to improve the efficiency of the whole computing process. The number of computers had increased significantly since 1950 as a consequence of the demand for more powerful computing centers to serve the military effort.[21] In a context of spreading automation, mechanizing part of the programmers' work seemed an obvious solution to improving efficiency and reducing programming costs.[22]

Many programming aids had already been introduced in the late 1940s to ease the task of programming.[23] These techniques included creating libraries of subroutines, introducing new procedures to make the storage of data in the computer more flexible, inventing mnemonic codes to label the most basic instructions, and designing programs to implement all these programming aids at the same time (compilers and interpreters). Despite all their differences, such technical developments shared the same goal: minimizing the time and work necessary to program a computer.[24]

In the early 1950s, however, a new idea started to gain momentum: simplify the programming task by expressing mathematical formulae in a straightforward way into the computer. In the US, this idea found a fertile ground within the Massachusetts Institute of Technology's (MIT's) Project Whirlwind.[25] In the context of this large-scale computing project, a team of programmers led by mathematicians Charles W. Adams and John W. Carr worked out new programming techniques to improve the Whirlwind computer's efficiency.[26] Their final goal was, in Adams' words, to create a procedure in which a mathematical problem ''can simply be set down in words and symbols and then solved directly by a computer without further programming.''[27]

Soon two young programmers working with Adams, J.H. Laning and N. Zierler, created a program capable of translating mathematical notation into computer instructions.[28]

Following the ideas of Adams and Carr, they created a notation ''as close as possible to ordinary mathematical notation'' and, more importantly, easy and fast to learn.[29] In the US, Laning and Zierler's work supposed a boost for the construction of programs capable of translating mathematical notation. Before long, similar developments took place elsewhere, including Remington Rand (producing Math-Matic and Unicode), IBM (Fortran), and the Purdue University Computing Laboratory (IT).[30]

In continental Europe, the advantage of expressing procedures in mathematical notation had also been considered since the late 1940s. More specifically, Eduard Stiefel and Heinz Rutishauser at the Institute of Applied Mathematics of the ETH Zurich had developed some ideas along these lines between 1949 and 1952.[31] However, these developments did not gather the same momentum as those in the US for several reasons. Primarily, there was a difference of scale. The small number of computers in continental Europe at that time, compared to the US and Britain, made the need for collaboration less critical.[32] In addition, the European countries' smaller-scale military research programs might have put less pressure on the programming task. There was also a difference in the institutional context of the European computing groups, which were often embedded in mathematical institutes,[33] likely making them less keen to implement management techniques typical in corporate and military settings. Finally, the first priority of European computing centers was to establish fluent relationships with their American and British counterparts rather than formal collaboration among themselves—not even with groups in their own countries.

Nevertheless, there were a few important examples of European collaboration in computing in these years.[34] The most relevant was the collaboration developed in the mid-1950s between Stiefel and Rutishauser at the ETH Zurich, and the West German groups of Alwin Walther at the TH Darmstadt and Robert Sauer, Hans Piloty, Friederich L. Bauer, and Klaus Samelson at the University of Munich. This collaboration became stronger after the now famous International Symposium on Electronic Digital Computers and Information Processing organized by Walther in October 1955 in Darmstadt with the support of GAMM.[35]

Darmstadt was a crucial event for the West German computing community.[36] Celebrated a few months after the admission of the country to NATO (May 1955) and just when the first West German electronic computers started running, the symposium helped break the relative isolation of the West German computing community.[37] In particular, Darmstadt strengthened the links between the computing groups in West Germany, Switzerland, and the US. In this regard, the attendance of Alston S. Householder, at that time ACM president, would turn out to be decisive to start the exchanges between the communities that would eventually lead to Algol.[38]

Darmstadt also had immediate consequences for the work of the West German computing groups. In Darmstadt, Heinz Rutishauser explained the need to come in the future to a ''unified algorithmic notation.''[39,40] Hence, Walther suggested starting to unify the design and notation of the programs written in their respective computing centers.[40] As a result, in May 1956, GAMM established a committee on programming to study the possibility of unification (*GAMM-Fachausschuss für Programmieren*).[41]

There is hardly any evidence about this committee's activities. According to retrospective accounts, the committee focused initially on terminology, translation techniques, and algebraic notation.[42] These accounts also suggest the possibility that the representatives from the Zurich, Munich, and Darmstadt groups (or ZMD group) at the GAMM committee did not share all the information with their Eastern colleagues, particularly in relation to translation techniques and common algebraic notation.[43] Therefore, it is difficult to know the level of development reached by the ZMD group when they proposed developing a common formula language with the ACM in October 1957. Nevertheless, clearly the ZMD group and the ACM leadership shared concerns and agreed on the need to move forward in the field of programming techniques in the direction of unification and the use of mathematical language.

Computing groups in America and Europe shared similar concerns about the need to foster information exchange between computers. They also agreed that the use of mathematical language offered good prospects. Eventually, however, two different approaches to this problem emerged: the ACM-GAMM project that would eventually lead to Algol and Share's UNCOL project. These two projects represented competing views on which level of

universality offered the most efficient way to exchange information.

The many levels of universality

In the mid-1950s, both corporate and state-sponsored computing centers established themselves as customer organizations. Such user groups as Share, USE (for users of Remington Rand's Univac), and DUO (for Datatron's customers) sought to foster cooperation between computing centers with the same type of computer.[44] Their final aim was to cut down programming costs—which caused much loss of manpower and machine availability—to reduce their dependence on the manufacturers. Their means was the exchange and dissemination of computer programs and the standardization of programming techniques.[45]

In the late 1950s, Share became the best known of these user groups and an influential actor in the American computing world. The ascendancy of Share was the result of IBM's increasing hegemony and the eclipse of its competitors, especially Remington Rand.[46] Share was founded in August 1955 as an association of computing centers from defense contractors, national security institutions, and corporations that all used IBM 704 computers. As historian Atsushi Akera has shown, Share had its roots in the tradition of collaboration within Southern Californian aviation firms that grew up on military contracts.[47] Share's founders were all managers at defense contractor computing centers such as Lockheed or of Cold War think tanks such as Rand. In this sense, Share embodied the Cold War characteristic alliance between industry, academia, and the military.[48]

Share played an important role in promoting information exchange between computing centers, thanks to IBM's active support. Share computing centers soon agreed on a set of standards, ranging from card decks to a common mnemonic code for programs, and quickly created an extensive catalog of programs.[49] IBM's department of publication and distribution supplied these programs.[50] Following the request of a computing installation, the department distributed a standardized set of card decks and program write-ups in 24 hours.[51] Together, IBM and Share developed a full customer service that provided computing centers with an extensive catalog of programs ready to be implemented.[52]

These close links between user groups and manufacturers contrasted with the goals and character of the other important computing association in the US, the ACM.[53] The ACM was more internationally oriented than the user groups, which were essentially national organizations. ACM members had a predilection for the study of programming techniques and applied mathematics and were less focused on hardware, which was the focus of the computing sections of the national engineering associations. Yet these differences did not always turn out to be an obstacle for collaboration between these organizations. Indeed, a significant example of collaboration between the ACM and the user groups took place in the spring of 1957 when they met to consider the possibility of creating a universal programming language in order to foster information exchange.

In May 1957, representatives from the ACM, Share, USE, and DUO convened in Los Angeles to consider the problems of information exchange between different computers.[54] The user group activities had contributed much to improve the flow of information between computing groups, but their efforts were naturally constrained to each group's scope. In addition, in the late 1950s, large-scale computing centers were starting to operate several different machines at the same time. Thus, communication problems between different machines had become an internal concern.

During this May meeting, the successes of Share and USE were presented as an example of the possibilities of a ''single universal computer language'' or at least a reduced number of ''universal languages.''[55] Thus, the user group representatives approved a resolution to invite the ACM National Council to take action and appoint three committees to study the creation of a universal programming language, the identification of possible areas of standardization, and the development of programming research.[56] It is important to stress that in this context the adjective ''universal'' pointed to the capability of working with any machine and for any computational purpose. Thus, the expression did not convey any internationalist message; this was an American national endeavor.

The ACM National Council decided to support the user groups and requested that the incoming president John W. Carr appoint a committee to investigate their recommendations.[57] During the next few months, the discussion with the user groups did not

bring any specific result and no committee was established.[56]

Eventually, the ACM established an Ad-Hoc Committee on Languages in February 1958, but the stimulus came from Europe. On 19 October 1957, representatives of the ZMD group wrote a letter to ACM President John W. Carr.[58] In this letter, they proposed a close conference between the ACM and GAMM to agree on a ''common formula language'' and stop the proliferation of programming languages both in the US and Europe.[59] In particular, they suggested creating a ''uniform formula language'' for scientific computations, based on simple mathematical notation, and capable of being implemented on any computer.[60] Indeed, this was in essence the ZMD group's ''formula-translation project,'' which was supposed to be extended to other European groups. At this point, however, they were still working on a formula translator and the project was far from completion.[61]

The eagerness of Carr and the ZMD group to join forces and overcome their weaknesses calls for a two-fold interpretation. Although the ZMD group had a clear ambition to expand the support of its views, to Carr a European initiative was most welcome in that it might offer the much-needed cement to lay the bricks of consensus among the user groups. Representatives of the ZMD group had been in contact with Carr since at least August of 1957 when Friederich L. Bauer (Munich) and Walther's pupil Hermann Bottenbruch (Darmstadt) had visited several American computing centers.[62] According to Bauer, their tour had been sponsored by the Office of Naval Research, thanks to the contacts of Bauer's boss Robert Sauer with NATO.[62] During their tour, Bauer and Bottenbruch learned about Perlis' IT language, Remington Rand's Math-Matic, and the different Share initiatives in programming. In exchange, they told their American colleagues, including Carr, about their formula-translation project.[62]

Bauer and Bottenbruch's trip was part of a broader initiative of Walther to strengthen links with the American computing world. From the early months of 1957, Walther had been in close contact with Householder, Samuel N. Alexander, and Isaac L. Auerbach from the US National Joint Computer Committee (NJCC), an umbrella for all computing organizations in the US.[63] The purpose of these contacts was a proposal submitted by the NJCC to UNESCO in November 1956 to celebrate an international conference on information processing.[64] This NJCC project mirrored other prominent initiatives of the Eisenhower administration such as the Atoms for Peace program (1955) and the International Geophysical Year (1957–1958).[65] These internationalist initiatives were aimed to serve American science and foreign policy, while reinforcing Western scientific cooperation.[66] Likewise, Walther activities, including the ACM-GAMM contacts of 1957, were also part of this attempt to build an Atlantic community of science.[67]

After Bauer and Bottenbruch returned from their American tour, the ZMD group met in Lugano, Switzerland.[68] There they decided to write down their proposal to the ACM. This meeting took place just two weeks after the Sputnik launch on 4 October 1957. The Sputnik launches further fostered transatlantic cooperation in science and technology, as historian John Krige has shown.[69] Thus, it is likely that this news acted as a catalyst for the ACM-GAMM contacts.

From this moment, events unfolded quickly. After receiving the ZMD group's proposal, Carr sent a memo to the ACM National Council and the user groups with a translation of the original letter, a first proposal for the dates of the conference, and a proposal to invite French and Soviet representatives.[70] Carr's memo also included a proposal for the composition of the American delegation as well. This delegation was stipulated to include six people, three representatives of the user groups (Share, USE, and DUO) and three from the universities (including Carr himself). All representatives were supposed to be familiar with the most advanced theoretical research in programming.[71]

Carr's memo provoked an internal debate within Share's Executive Board concerning the position that the organization should adopt in relation to the ACM-GAMM planned conference. Chairman Francis V. Wagner and Secretary Herbert S. Bright stood for the two main positions within the board.[72] Wagner was clearly against the idea of a universal programming language. A leading figure of the computing groups of the Southern Californian aviation industry, Wagner did not believe that such a language would be possible or even beneficial. He was a supporter of the UNCOL initiative and was convinced of the need to keep a variety of programming languages. Bright, however, was more sympathetic to the idea. Working

at the Westinghouse-Bettis Atomic Power Laboratory, Bright was a prominent figure in Share, but he also kept close links with the ACM. Still, Bright was not thinking of a full programming language, either. Bright thought the effort should focus on reaching ''general agreement on basic attributes of source languages,'' which would facilitate compatibility between machines.[73]

The discussion within Share's Executive Board was part of a broader debate over the strategy to follow in order to simplify the programming task. The increasing variety of new machines and the proliferation of programming languages demanded more and more compilers, which were expensive and time consuming to produce.[74] At that time, two levels of universality were discussed within Share to deal with this problem. A first solution was to implement universality at the highest level—that is, to adopt a new programming language as the standard for all machines and purposes (or at least for scientific computing). Preferably, this language would be a standardized Fortran-like language or a completely new language as suggested by the ZMD project. The second option, which Wagner strongly supported within Share, was to preserve the variety of programming languages and achieve universality at an intermediate level by developing a universal language between programming languages and machine codes—that is, UNCOL.[75]

UNCOL was meant to be a system based on the ''three-level of language'' concept,[76] which consisted of program-oriented languages (third level), an intermediate standard language or UNCOL (second level), machine languages (first level), and generators and translators. *Generators* were supposed to be compilers to translate from the program-oriented languages to UNCOL. *Translators* were compilers that would perform the same transformation, but from UNCOL to machine languages. The Share UNCOL committee saw many advantages in this scheme of double translation. Because UNCOL was supposed to share many characteristics of machine languages, translators and generators would be faster and easier to write than full compilers to translate from the highest to the lowest level. Furthermore, UNCOL offered a high level of versatility because each new machine or new program-oriented language only needed a new translator or generator. In addition, customers could shift to a new language without big breakthroughs in programming techniques. Finally, it was believed that such an intermediate language would be easy to standardize and be ready in three years' time.

Although Share had started to consider the development of UNCOL as early as the spring of 1957, the ACM-GAMM contacts acted as ''a catalytic agent.''[77] By the early months of 1958, Share had produced a first draft of the UNCOL concept, and there was a growing consensus that UNCOL offered the best solution to further transportability of programs.[76] Not everybody agreed on this strategy, however. There were prominent people within Share—such as Bright and Joseph Wegstein—who clearly advocated a standardized Fortran-like language. Others were equally skeptical of both solutions.[78] Therefore, Share's Executive Board reached a delicate consensus: pursue a prudent strategy in relation to the ACM-GAMM initiative to avoid both the impression of full commitment and of rejection.

Eventually, the ACM-GAMM joint meeting took place at the ETH Zurich from 27 May to 2 June 1958.[79] Before this international meeting, however, the ACM delegation—the ACM Ad Hoc Committee on Languages—held three internal meetings to work out a proposal.[80] The ACM committee's proposal was oriented toward a universal programming language rather than an UNCOL-like solution.[80] Not surprisingly, the outcome of the Zurich meeting clearly aimed at a universal programming language, which was called the International Algebraic Language (IAL) and soon renamed Algol. The new language's first specifications were published in December 1958 in a preliminary report by Perlis (ACM) and Klaus Samelson (GAMM).[79] Just a few months before, Share had welcomed IAL in a resolution approved during its 11th meeting.[81]

This resolution has often been interpreted as a full support of Algol as the universal programming language. Yet this view is based on an imprecise reading of the content and the context of the resolution. The language of the resolution was deliberately ambiguous. Share had just reached a fragile consensus about the convenience of keeping a variety of programming languages. The resolution stated the willingness of the users group ''to study the proposed International Algebraic Language and to implement its adoption as a Share standard.''[81] The resolution clearly stated that Share endorsed Algol as a Share standard, not the standard programming language. Share therefore welcomed the new

language as one of several problem-oriented languages.

Conclusion

In August 1959, Share president Frank Verzuh described how he was surprised that during a visit to Europe ''people would talk to ... [him] ... in Algol language.''[82] Algol was indeed warmly welcomed in many continental computing centers. The reasons behind this positive reception fall outside this article's scope, but it is likely that Algol might have been perceived as a huge step forward to advanced cooperation between centers. In the US, however, the new language faced a less uniform reception. Afterward, many discussions took place in the early 1960s about the qualities of the new language, the need and extent of its standardization, and even the possibility of a Fortran-Algol merge. Much criticism was also directed to Share and IBM for their lack of support to turn Algol into a truly universal programming language. All these disputes played an important role in the construction of the Algol-Fortran polarity.

The Algol-Fortran polarity, however, was a post-hoc construct. Algol was a product of the reinforcement of transatlantic cooperation in science of the late 1950s. Therefore, conflicts around Algol must be framed in this context. These conflicts were rooted in an earlier and deeper problem shared by computing groups on both sides of the Atlantic: the problem of information exchange between computers. In the late 1950s, the increasing production of new machines and programming languages made it urgent. Confronted with this acute problem, IBM's user group Share chose to develop a sort of universal translator (UNCOL). It was expected that UNCOL would eventually simplify the translation process and, above all, preserve the diversity of programming languages. The alliance between the ACM and continental European computing groups, however, sought to reduce the variety of programming languages and adopt a single universal programming language. Therefore, if the history of Algol calls for a dichotomous interpretation, it was a matter of uniformity versus diversity.

Acknowledgments

This article is part of the Algol Compilers: University-Industry Co-entrepreneurship project, funded by the Netherlands Organization for Scientific Research (NWO) and the European Science Foundation as part of the Software for Europe collaborative research project. Research for this article was carried out thanks to a 2009 Arthur L. Norberg Travel Award from the Charles Babbage Institute. I thank Jeffrey R. Yost, Gerard Alberts, and the two anonymous referees for critical comments on this article.

References and notes

1. For the emergence of the concept of programming language, see P.E. Ceruzzi, *A History of Modern Computing,* MIT Press, 2003, pp. 81–88.
2. M. Fuller, ed., *Software Studies: A Lexicon,* MIT Press, 2008, pp. 3–4.
3. J. Sammet, *Programming Languages: History and Fundamentals,* Prentice-Hall, 1969; D. Knuth and L.T. Pardo, ''The Early Development of Programming Languages,'' *Encyclopedia of Computer Science and Technology,* vol. 7, J. Belzer, A.G. Holzman, and A. Kent, eds., Marcel Dekker, 1977, pp. 419–493; R.L. Wexelblat, ed., *History of Programming Languages,* ACM Press, 1981; T.M. Bergin and R.G. Gibson, eds., *History of Programming Languages II,* ACM Press, 1996. For an exception to this approach, see J.R. Holmevik, ''Inside Innovation: The Simula Research Laboratory and the History of the Simula Programming Language,'' Simula Research Lab, 2005.
4. A language history chart probably appeared first in C.J. Saw, ''Milestones in Computer Programming,'' *SIGPLAN Notices,* Feb. 1965.
5. A Babel-like illustration appeared for the first time in January 1961 as a cover of *Comm. ACM.*
6. An early mention to a ''universal programming language'' took place during the meeting on ''Information Exchange'' between the user groups Share, USE, and DUO and the ACM in Los Angeles, 9–10 May 1957. UNCOL Committee, ''Report Universal Language Committee, UNCOL (Universal Computer Oriented Language),'' 8 Apr. 1958, Share records, CBI 21, box 1, folder 3.
7. In 1961 there were 25 institutions in 10 European countries involved in the implementation of Algol. P. Naur, ''Progress of Algol in Europe,'' *Algol Bulletin,* suppl. 18, 1961, pp. 1–6.
8. I use ''standard'' here to mean one single universal programming language for all purposes, all users, and all machines. For a discussion of the multiple meanings of standard, see J. Sumner and G.J.N. Gooday, ''Introduction: Does Standardization Make Things Standard?'' *By Whose Standards? Standardization, Stability and Uniformity in the History of Information and Electrical Technologies,* Continuum, 2008, pp. 1–13.

9. For example, P. Naur, ''Progress of Algol in Europe;'' H. Rutishauser, *Description of ALGOL 60: Handbook for Automatic Computing,* vol. I, Springer Verlag, 1967, pp. 4–8; R.W. Bemer, ''A Politico-Social History of Algol,'' *Ann. Rev. Automatic Programming,* vol. 5, 1969, pp. 151–237; A.J. Perlis, ''The American Side of the Development of Algol,'' *History of Programming Languages,* R.L. Wexelblat, ed., ACM Press, 1981, pp. 75–91; P. Naur, ''The European Side of the Last Phase of the Development of Algol 60,'' *History of Programming Languages,* R.L. Wexelblat, ed., ACM Press, 1981, pp. 92–138; C.H. Lindsey, ''A History of Algol 68,'' *History of Programming Languages II,* T.M. Bergin and R.G. Gibson, eds., ACM Press, 1996, pp. 97–132; F.L. Bauer, ''Die Algol-Verschwörung'' [The Algol Conspiracy], *Geschichten der Informatik: Visionen, Paradigmen, Leitmotive,* H.D. Hellige, ed., Springer, 2004, pp. 237–254.
10. Bemer, ''A Politico-Social History of Algol.''
11. See A.J. Perlis and R. Iturriaga, ''An Extension to Algol for Manipulating Formulae,'' *Comm. ACM,* vol. 7, no. 2, 1964, pp. 127–130; T.E. Cheatham, Jr. ''Editor's Note: A Successor to Algol?'' *Comm. ACM,* vol. 7, no. 7, 1964, p. 422; J. Sammet, ''Programming Languages: History and Future,'' *Comm. ACM,* vol. 15, no. 7, 1972, pp. 601–610; S. Rosen, ''Programming Systems and Languages, 1965–1975,'' *Comm. ACM,* vol. 15, no. 7, 1972, pp. 591–600; C.H. Hoare, ''The Emperor's Old Clothes,'' *Comm. ACM,* vol. 24, no. 2, 1981, pp. 75–83.
12. These attributed features of Algol fit neatly with a broader discourse that tend to highlight the supposed differences between European and American technological styles. I thank the reviewer who brought this my attention.
13. ''Software Pioneer Peter Naur Wins ACM's Turing Award,'' ACM press release, 1 Mar. 2006; http://campus.acm.org/public/pressroom/press_releases/3_2006/turing_3_01_2006.cfm.
14. Perlis, ''The American Side of the Development of Algol;'' and Naur, ''The European Side of the Last Phase of the Development of Algol 60.''
15. Perlis, ''The American Side of the Development of Algol,'' p. 88.
16. Perlis, ''The American Side of the Development of Algol,'' p. 78.
17. Perlis, ''The American Side of the Development of Algol,'' p. 87.
18. Comment from A.J. Perlis. See T. Cheatham, ''Transcription of Question and Answer Session,'' *History of Programming Languages,* R.L. Wexelblat, ed., ACM Press, 1981, p. 164.
19. R.L. Wexelblat, ed., *History of Programming Languages,* ACM Press, 1981.
20. For this paragraph, see especially Bauer, ''Die Algol-Verschwörung;'' Naur, ''The European Side of the Last Phase of the Development of Algol 60;'' and N.J. Lehmann, ''Algol im Ostblock und der Weg zu Systemen von Programmiersprachen'' [Algol in the Eastern Bloc and the Way to Systems of Programming Languages], *Geschichten der Informatik: Visionen, Paradigmen, Leitmotive,* H.D. Hellige, ed., Springer, 2004, pp. 256–259.
21. For the link between research on computers and the military in the US, see P.N. Edwards, *The Closed World: Computers and the Politics of Discourse in Cold War America,* MIT Press, 1996.
22. D.F. Noble, *Forces of Production: A Social History of Industrial Automation,* Knopf, 1984.
23. H.H. Goldstine, *John von Neumann: Planning and Coding of Problems for an Electronic Computing Instrument,* Inst. for Advanced Study, 1947; M.V. Wilkes, D.J. Wheeler, and S. Gill, *The Preparation of Programs for an Electronic Digital Computer: With the Special Reference to the EDSAC and the Use of Subroutines,* Addison-Wesley, 1951.
24. J.W. Backus and H. Herrick, ''IBM 701 Speedcoding and other Automatic-Programming Systems,'' *Proc. Symp. Automatic Programming for Digital Computers,* Navy Advisory Math. Panel, Office of Naval Research, 1954, p. 106.
25. For the Whirlwind project, see K.C. Redmond and T.M. Smith, *Project Whirlwind: The History of a Pioneer Computer,* Digital Press, 1980.
26. C.W. Adams, ''Small Problems on Large Computers,'' *Proc. 1952 ACM Nat'l Meeting,* ACM Press, 1952, pp. 99–102; J.W. Carr, ''Progress of the Whirlwind Computer Towards an Automatic Programming Procedure,'' *Proc. 1952 ACM Nat'l Meeting,* ACM Press, 1952, p. 237.
27. Adams, ''Small Problems on Large Computers,'' p. 99.
28. J.H. Laning Jr. and N. Zierler, ''A Program for Translation of Mathematical Equations for Whirlwind I,'' eng. memo E-364, MIT Instrumentation Lab, Jan. 1954.
29. Laning Jr. and Zierler, eng. memo E-364, pp. 1–2.
30. Sammet, *Programming Languanges,* pp. 132–172. See also Knuth and Pardo, ''The Early Development of Programming Languages,'' pp. 66–73.
31. On Rutishauser's ideas, see Knuth and Pardo, ''The Early Development of Programming Languages,'' pp. 24–29; H. Petzold, *Rechnende Maschinen. Eine historische Untersuchung ihrer Herstellung und Anwendung vom Kaiserreich bis zur Bundesrepublik* [Computing Machines: A Historical Study of their Production and Application from the Kaiserreich to the Federal Republic], VDI Verlag, 1985, pp. 484–487.
32. In 1955, no country on the European continent had 10 working computers. In comparison, the

US had more than 80 and the UK more than 25. W. Aspray, ''International Diffusion of Computer Technology, 1945–1955,'' *Annals of the History of Computing,* vol. 8, no. 4, 1986, p. 352.

33. M. Rees, ''Applied Mathematics in Western Europe,'' ONR report, 1948, p. 1, Int'l Computing collection, Charles Babbage Inst., CBI 62, box 3, folder 11.
34. The earliest initiatives in Western Europe of cooperation in computing date back to 1947. The Mathematisch Centrum in Amsterdam attempted first cooperation with Belgium and France in the summer of 1947. Then the Dutch center established a European Committee of Computing Institutes, ''Samenwerking met Frankrijk en België (. . .) op het gebied van grote rekenmachines in Juli 1947'' [Collaboration with France and Belgium in the Field of Large Computing Machines by July 1947], Noord-Hollands archive, Mathematisch Centrum archive, dossier 76; A. Van Wijngaarden, European Committee of Computing Institutes, Noord-Hollands archive, Mathematisch Centrum archive, dossier 71.
35. Proceedings published as J. Wosnik, ed., *Elektronische Rechenmaschinen und Informationsverarbeitung. Nachrichtentechnische Fachberichte* [Electronic Digital Computers and Information Processing], vol. 4, Vieweg & Sohn, 1956.
36. H. Petzold, ''Eine Informatiktagung vor der Gründung der Informatik. Die Darmstädter Konferenz von 1955'' [An Informatics Conference before the Foundation of Informatics: The Darmstadt Conference of 1955], *Zahl, Ordnung. Studien zur Wissenschafts- und Technikgeschichte,* R. Seising, M. Folkerts, and U. Hashagen, eds., Franz Steiner Verlag, 2004, pp. 759–782.
37. On postwar international isolation of German scientists, see U. Deichmann, ''Emigration, Isolation and the Slow Start of Molecular Biology in Germany,'' *Studies in History and Philosophy of Biological and Biomedical Sciences,* vol. 33, no. 3, 2002, pp. 460–464. A. Walther compared the importance of the symposium by using Tennessee Williams' expression ''the violets in the mountains have broken the rocks.'' Wosnik, *Elektronische Rechenmaschinen und Informationsverarbeitung,* p. viii.
38. On the friendship between F.L. Bauer and A.S. Householder, see Petzold, *Rechnende Maschinen,* p. 488; F.L. Bauer, ''Memories of Alston Householder (1904–1993);'' http://www3.math.tu-berlin.de/householder_2008/Cleve.html.
39. K. Samelson expressed similar views in a symposium in Dresden. See K. Samelson, ''Probleme der Programmierungstechnik'' [Problems of Programming Technique], *Bericht über das Internationale Mathematiker-Kolloquium. Aktuelle Probleme der Rechentechnik,* VEB Deutscher Verlag der Wissenschaften, 1957, pp. 61–68.
40. Wosnik, *Elektronische Rechenmaschinen und Informationsverarbeitung,* p. 143.
41. Lehmann, ''Algol im Ostblock und der Weg zu Systemen von Programmiersprachen,'' p. 257.
42. Participants F.L. Bauer and N.J. Lehman have briefly explained the activities of this committee. Bauer, ''Die Algol-Verschwörung;'' Lehmann, ''Algol im Ostblock und der Weg zu Systemen von Programmiersprachen.''
43. Lehmann, ''Algol im Ostblock und der Weg zu Systemen von Programmiersprachen,'' p. 258.
44. For the early days of Share, see A. Akera, *Calculating a Natural World: Scientists, Engineers, and Computers during the Rise of the US Cold War Research,* MIT Press, 2007, chapt. 7.
45. ''F. Jones to Share members,'' 9 Aug. 1955, Share records, CBI 21, box 1, folder 1, p. 1.
46. Ceruzzi, *A History of Modern Computing,* chapt. 2.
47. Akera, *Calculating a Natural World,* pp. 251–255.
48. Akera, *Calculating a Natural World,* p. 272.
49. ''A Proposal Relative to Cataloguing 704 Programs Distributed by Share,'' 24 Feb. 1956, Share records, CBI 21, box 1, folder 2.
50. IBM was in charge of the distribution at least from 1957. ''Verbatim Transcript of the 9th Meeting,'' 1 Oct. 1957, Share records, CBI 21, box 3, folder 13, pp. 32–36.
51. The expression is from Franz Ross, at that time head of the IBM Dept. of Publication and Distribution. ''Verbatim Transcript of the 9th Meeting,'' p. 36.
52. A. Akera (*Calculating a Natural World*) has corrected former views on Share as a pure annex of IBM by stressing Share's role as intermediary between IBM and its customers. Yet Akera fails to appreciate the extent to which the benefits of collaboration created a reciprocal dependence between both organizations. Whereas Share's members needed IBM to produce their library of previously tested programs, the IBM sales force could use the possibility of Share membership as a plus in their marketing of IBM computers.
53. The ACM was founded in 1947 as an association of individual scientists and engineers working in the new field of computing. ''Notice on Organization of an Eastern Association for Computing Machinery,'' 25 Jun. 1947, ACM records, CBI 205, box 11, folder 15. The other two American computing organizations were the computing sections of the American Inst. of Electrical Engineers (AIEE) and the Inst. of Radio Engineers (IRE).
54. ''Recapitulation of the May, 1957, Los Angeles Meetings on Information Exchange.'' A copy of this document is included as appendix A in the following document: UNCOL Committee,

''Report Universal Language Committee,'' 8 Apr. 1958, Share records, CBI 21, box 1, folder 3.
55. ''Recapitulation of the May, 1957, Los Angeles Meetings on Information Exchange,'' UNCOL Committee Report, p. 3.
56. ''Recapitulatioin of the May, 1957, Los Angeles Meetings on Information Exchange,'' UNCOL Committee Report, p. 4.
57. ''Minutes of the ACM Council,'' 20 Jun. 1957, appendix A, ACM records, CBI 205, box 11, folder 16.
58. ''H. Rutishauser et al. to J.W. Carr,'' 19 Oct. 1957, appendix A, UNCOL Committee Report.
59. ''H. Rutishauser et al. to J.W. Carr,'' UNCOL Committee Report, pp. 2, 3.
60. ''H. Rutishauser et al. to J.W. Carr,'' UNCOL Committee Report, p. 1.
61. ''H. Rutishauser et al. to J.W. Carr,'' UNCOL Committee Report, p. 2.
62. F.L. Bauer, ''Memories of Alston Householder (1904–1993).''
63. ''I.L. Auerbach to P. Auger,'' 22 Apr. 1957, UNESCO archives, Box Int'l Conf. Information Processing 1959, France, folder part I. In this letter, Auerbach informs the head of UNESCO's Science Department (P. Auger) of his contacts with A. Walther, who had just attended the last meeting of the NJJC in Chicago.
64. Joint Computer Committee, ''Proposal for an International Conference on Information Processing Systems,'' 1 Nov. 1956, UNESCO archives, Box Int'l Conf. Information Processing 1959, France, folder part I.
65. Joint Computer Committee Proposal, pp. 9, 10.
66. J. Krige, *American Hegemony and the Postwar Reconstruction of Science in Europe,* MIT Press, 2006. For US foreign policy and scientific internationalism during this period, see especially J. Manzione, '''Amusing and Amazing and Practical and Military': The Legacy of Scientific Internationalism in American Foreign Policy, 1945–1963,'' *Diplomatic History,* vol. 24, no. 1, 2000, pp. 21–56.
67. I take this idea of Atlantic community building in science from Krige, *American Hegemony,* p. 3.
68. Petzold, *Rechnende Maschinen,* p. 498.
69. Krige, *American Hegemony,* chapt. 7.
70. ''J.W. Carr to ACM Council,'' 26 Oct. 1957, UNCOL Committee Report, p. 2.
71. ''J.W. Carr to ACM Council,'' 26 Oct. 1957, UNCOL Committee Report, p. 3.
72. For their positions, see the UNCOL Committee Report: ''F.V. Wagner to J.W. Carr,'' 22 Nov. 1957; ''H.S. Bright to J.W. Carr,'' 26 Nov. 1957. About F.V. Wagner, see Akera, *Calculating a Natural World,* p. 253. About H.S. Bright, see E.A. Weiss, ''Biographies,'' *Annals of the History of Computing,* vol. 10, no. 3, 1988, pp. 217–218.
73. ''H.S. Bright to J.W. Carr,'' 26 Nov. 1957, UNCOL Committee Report, p. 4.
74. T.B. Steel Jr., ''A First Version of UNCOL,'' *Proc. Western Joint IRE-AIEE-ACM Computer Conf.,* ACM Press, 1961, p. 371.
75. For an overview of UNCOL, see T.B. Steel, ''UNCOL: The Myth and the Fact,'' *Ann. Rev. Automatic Programming,* vol. 2, 1961, pp. 325–344; Sammet, *Programming Languages,* pp. 708–709.
76. For a detailed description of UNCOL, see the UNCOL Committee Report, especially the following documents: ''Universal Language Committee to Share Membership,'' 8 Apr. 1958; ''The 3-Level Concept,'' 28 Feb. 1958, appendix B; ''UNCOL System Notation,'' appendix C; ''Use of the UNCOL System,'' appendix D; and ''The Problem of Programming Communication with Changing Machines: A Proposed Solution,'' appendix E.
77. The expression is from T.B. Steel (System Development Corp.), member of the Share UNCOL Committee. T.B. Steel, UNCOL Committee Report, 17 Mar. 1961, p. 2, F.V. Wagner papers, CBI 6, box 1, folder 32.
78. For the different positions within Share, see the panel discussion on universal languages (one set of notes), ''Verbatim Transcript of the Share 10th Meeting,'' 26–28 Feb. 1958, appendix E, Share records, CBI 21, box 3, folder 16.
79. A.J. Perlis and K. Samelson, ''Preliminary Report: International Algebraic Language,'' *Comm. ACM,* vol. 1, no. 12, 1958, pp. 8–22.
80. Perlis and Samelson, ''Preliminary Report,'' p. 9.
81. ''Verbatim Transcript of the Share 11th Meeting,'' 9–12 Sept. 1958, appendix C-17, p. 1, Share records, CBI 21, box 3, folder 17.
82. Bemer, ''A Politico-Social History of Algol,'' p. 168.

David Nofre is a postdoctoral research fellow at the University of Amsterdam. His Algol Compilers: University–Industry Co-entrepreneurship project is funded by the Netherlands Organization for Scientific Research as part of the European Science Foundation's Software for Europe project. Nofre has a PhD in the history of science from Universitat Autonòma de Barcelona. Contact him at d.nofremateo@uva.nl.