On the Difficulty of Describing Difficult Things

Troy K. Astarte
troy.astarte@ncl.ac.uk

April 25, 2019

Abstract

In the 1960s, a full formal description was seen as a crucial and unavoidable part of creating a new programming language. A key part of that was a thorough and rigorous description of the semantics. However, in the decades since, the focus on providing this has somewhat diminished. Why was formal semantics once seen as so critical? Why did it not succeed in the ways hoped?

My PhD was spent researching the early history of programming language semantics, with a particular focus on the IBM Laboratory Vienna under Heinz Zemanek, and the Programming Research Group at Oxford University under Christopher Strachey. It could also be seen as an history of *model-based* (rather than algebraic or axiomatic) semantics. In this talk, I will present the key findings of my research, as a way to whet my audience's appetite for my thesis, and argue that formal description was a crucial part of the formation of theoretical and formal computer science in the European tradition.

(*162 words*)

Extended Abstract

In 1970, formal description of programming languages was regarded as the state of the art in computing; the golden standard for academic research. When Friedrich Bauer, gave a talk at the tenth anniversary celebration of IFIP on the history of computation [Bau72], the story went back as far as ancient methods of counting with pebbles and coins and came right up to date by referring to the ALGOL 68 description as a kind of pinnacle of achievement in computing: how far it had come since its humble beginnings! Formal description was also considered as something of a standard in academic circles: when Working Group 2.1 were developing a successor to ALGOL 60, it was agreed unanimously that the new language must have fully formalised syntax and semantics. But such full formal descriptions were rare even in the 1960s, and those which were presented (such as ALGOL 68 [WMPK69] and PL/I [ULD66]) were not at all well-received. In the decades since the 1970s, full language descriptions have been even less common. So why is this? Why was formal semantics once seen as

a great hope for computing? Why did it not receive a warm reception? How was formal semantics worked on? What kind of impact did it have on computing despite its lack of mainstream success? These are the questions considered the author's dissertation [Ast19]: the key findings are summarised in this talk.

Getting to grips with programming was and is hard. The core association of variables and their values could be done with a 'state', a key component of both operational and denotational semantics, but this became more complicated as new challenges were added, such as jumps, procedures, and concurrency.

The main motivations for approaching semantics can be divided into two categories: 'theoretical' and 'practical'. For the first, there was a desire amongst many to formalise the foundations of computing, combatting the "vague feeling of unease" felt when designing a programming language due to the purely intuitive understanding most people had of programming. Mahoney wrote about the struggle to find a theory for computing and placed formal semantics as central to that [Mah11]. On the practical side, there were very real concerns about finding the correctness of compilers, determining appropriate ways to standardise and define languages for a broad audience, and easing the very difficult task of designing a good programming language.

These different challenges led to different kinds of semantics. Despite fundamental similarities in model-based semantics (see [JA18]), notational variances had a strong impact on the usability of definitions. A major part of the reason for differences in style was due to the varying backgrounds of the researchers involved. That said, most workers in formal description took to it as a result of wrestling with programming language design. The influence of one language in particular, ALGOL 60, was very important: at least six formal definitions of that language were made.

Organisations played an important role in this academic period. The need for an industrial product was a heavy motivator for the IBM Vienna Lab, contrasting with Strachey's research group, "a highly critical and thoughtful atmosphere in which ad hoc or superficial ideas [were] given very short shrift" [Str71].

Collaboration was a key element in successfully developing approaches for formal semantics. For example, denotational semantics grew first from Landin and Strachey working together, took off only thanks to contributions from Scott, and really flourished with the involvement of graduate students and RAs at the Programming Research Group. However, groups attempting to work together across description styles often struggled: IFIP's Working Group 2.2, on formal language description, frequently suffered crises of identity as they tried to determine their agenda. Some exceptions exist, however; notably the later Vienna work on VDM, which took denotational semantics and added in ideas from the earlier VDL period.

Throughout its early period, formal semantics faced heavy criticism. Many working programmers preferred intuitive notions of understanding, and argued that rewriting a language in a metalanguage was not really giving the meaning. The largest criticism against formal semantics was that the results were unusable due to their size and complexity—but, as

Hoare said, "difficult things are difficult to describe" [in Wal69].

Although formal semantics didn't receive the success once expected, the work was impactful in other ways through its influence on other parts of computing. Some areas of theoretical computing, such as program verification and type theory, grew out of work on formal semantics and have become rich fields of study in their own right.

(744 words)

References

[Ast19] Troy K. Astarte. "Formalising Meaning: a History of Programming Language Semantics". PhD thesis. Newcastle University, 2019.

[Bau72] Friedrich L. Bauer. "From Scientific Computation to Computer Science". In: *The Skyline of Information Processing: Proceedings of the tenth anniversary celebration of the IFIP*. Ed. by Heinz Zemanek. IFIP. North-Holland, 1972.

[JA18] Cliff B. Jones and Troy K. Astarte. "Challenges for semantic description: comparing responses from the main approaches". In: *Proceedings of the 3rd School on Engineering Trustworthy Software Systems*. Ed. by Jonathan P. Bowen and Zhiming Liu. Lecture Notes in Computer Science 11174. 2018, pp. 176–217.

[Mah11] Michael S Mahoney. "Computer Science: The Search for a Mathematical Theory". In: *Histories of Computing*. Ed. by Thomas Haigh. Harvard University Press, 2011. Chap. 10, pp. 128–46.

[Ste66] T. B. Steel. *Formal Language Description Languages for Computer Programming*. North-Holland, 1966.

[Str71] Christopher Strachey. *Curriculum Vitae*. Christopher Strachey Collection, Bodleian Library, Oxford. Box 248, A.3. Written by Strachey to send to the Times newspaper in case of the need for obitual information. 1971.

[Wal69] Kurt Walk. *Minutes of the 3rd meeting of IFIP WG 2.2 on Formal Language Description Languages*. Held in Vienna, Austria. Chaired by T. B. Steel. 1969.

[WMPK69] A. van Wijngaarden, B. J. Mailloux, J. E. L. Peck, and C. H. A. Koster. *Report on the Algorithmic Language ALGOL 68*. Second printing , MR 101. Mathematisch Centrum, Amsterdam, 1969.

[ULD66] ULD-III. *Formal Definition of PL/I (Universal Language Document No. 3)*. Tech. rep. 25.071. Author given as 'PL/I – Definition Group of the Vienna Laboratory'. IBM Laboratory Vienna, 1966.