Prehistory of Programming Languages
Matthias Felleisen

The Report on Algol 60

Content The report specifies the Algol 60 programming language. Specifically, it spells out the alphabet and the grammatical rules—dubbed syntax—for the language and thus determines the legal Algol 60 programs. Informal language explains what the various languages constructs mean.

Ideas The report introduce a technique for formally specifying the alphabet and grammar of an industrial-strength programming language. The so-called Backus-Naur Form (BNF) has been in use ever since. In addition, the report uses the novel technique of using meaningful names for grammatical pieces of Algol 60 phrases to suggest meaning. Finally, the introduction of “call by name” and its substitution-based reasoning alludes to a connection to the lambda calculus.

Abstract Machines

Abstract Machines
Landin’s paper introduces the lambda calculus as a basis for defining a programming language; he dubs the language *Iswim*. The paper(s) presents the syntax of the language without regard to details and a mechanical evaluator, a so-called machine, that in a step-wise fashion produces an answer for the programs in the language. The goal is to use this novel language to explain other languages.

**Ideas** The paper contains many important ideas. First, it introduces the notion of abstract syntax, i.e., that only certain elements of a phrase matter and the rest can be ignored. Second, it is one of the first papers to spell out the meaning of a language in a concise mathematical manner. (Bauer, McCarthy, and Böhm have alternative proposals, though none of them survived.) Third, the paper presents the idea that one can use this novel language as a meta-language to explain other languages. Fourth, Landin adds imperative constructs to the lambda calculus, anticipating mostly functional programming à la Scheme and ML. (Burge (IBM) later implemented the machine in the 1970s but few noticed.)

**Explaining Algol's Meaning**

Landin’s paper proposes to map new industrial languages such as Algol 60 to “canonical” mathematical models such as his Iswim language. His use of Iswim demonstrates why he added specific imperative features to the language.
Ideas  The paper gives birth to the idea of a meta-language, specifically the lambda calculus as a meta-language. It is thus the cause for a search for a “proper” (compositional) meaning of the language and, as a reaction, to the search for an explanation of meaning without the clumsy-looking imperative constructs.

@inproceedings{ jmc:algol,
  author="John McCarthy",
  title="A formal description of a subset of {ALGOL}",
  booktitle="Formal Language Description Languages for Computer Programming",
  year=1966,
  pages="1--12"
}

Content  McCarthy uses recursion equations to assign meaning to fragments of Algol 60. The recursion is reminiscent of his Lisp interpreter for Lisp and is probably the precursor for all interpreter-oriented work.