The Science of Programming Languages & tools

Parsing of programming languages was based on the study of grammars, formal languages and automata. At ICALP’72, 30 out of 60 presentations dealt with formal languages and automata theory. In the 1970’s, the theory of programming languages turned to the description of their semantics with algebra, denotational semantics, and mathematical logic. Since then, new conferences have appeared about logic in computer science, principles of programming languages, compilers, functional programming, types, static analysis, concurrency, automatic verification.

**Programming Languages**

The first 50 programming languages predicted by Peter Landin in 1966 are now nearly extinct. Today languages are introduced with their semantics written in a more or less formal setting. Mathematical models have also influenced the design of new concepts: types, modules, algebra, etc.

From the 1950’s foundational computation models...

- Lambda-calculus
  - Church
  - Closures
  - Monads
  - Turing

**Programming Paradigms**

- Object-oriented
- Logic
- Scripting
- Descriptive
- Functional
- Procedural

**Programming Tools**

The first programming tools dealt with compiler construction or program profiling. Nowadays they include program verification, static analysis, and program testing. Those new tools have followed the mainstream progress in the semantics of programming languages: descriptiv types, interactive proof-checkers, automatic provers, and abstract interpreters.