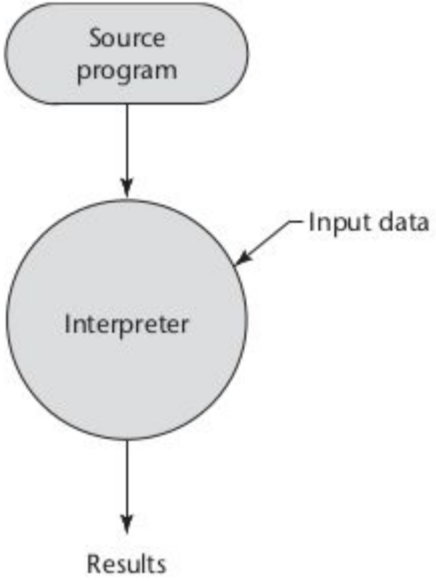
Interpreted Languages and Hybrid Languages

CARLOS EDUARDO SANCHEZ TORRES

Pure interpretation

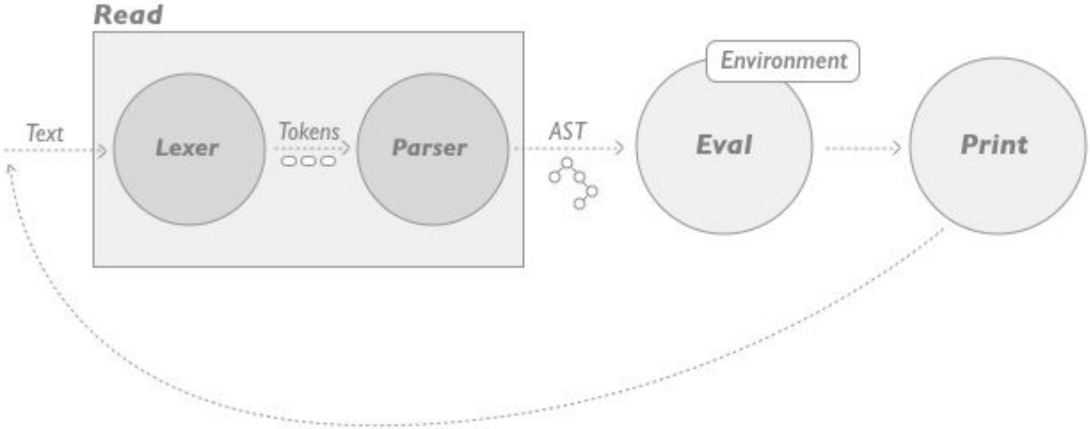
- Type-checker run-time
- Portable
- Interpreter



LISP

- Symbolic expression
- Atoms
- Lists
- LISP interpreter
- Number table

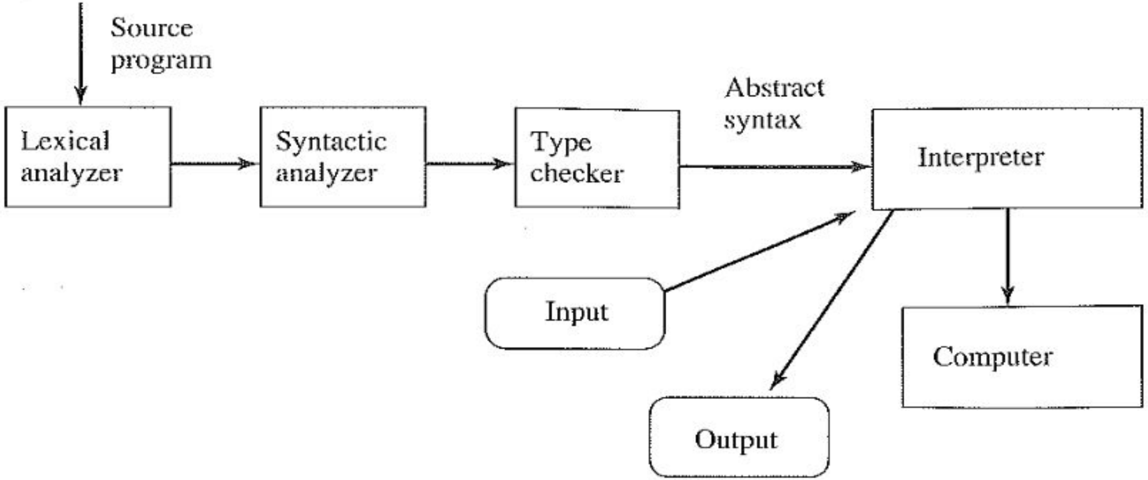
Read-Eval-Print Loop



https://www.uraimo.com/2017/02/05/building-a-lisp-from-scratch-with-swift/

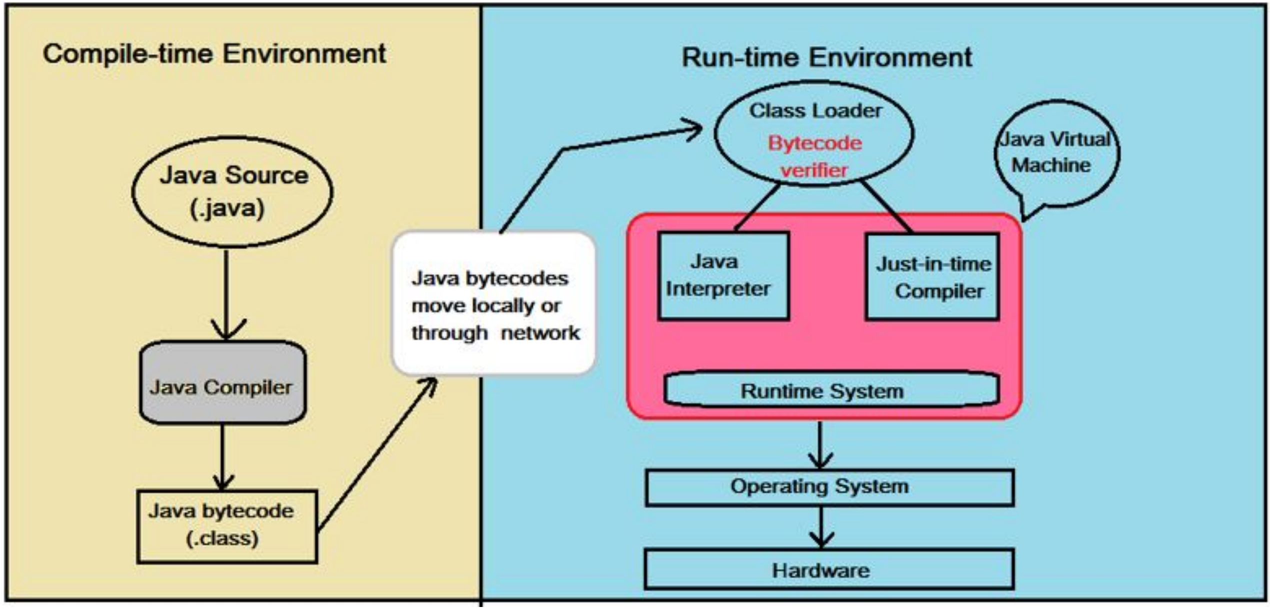
Hybrid Implementation Systems

- Type-checker compile time
- Lexical analyzer
- Portable
- Virtual machine
- Intermediate code



JAVA

- Bytecode
- Portable
- Class loader



REFERENCES

[1] Tucker, A. B., &; Noonan, R. (2007). Programming languages: Principles and paradigms. In Programming languages: Principles and paradigms. New York: McGraw-Hill Higher Education.

[2] Sebesta, R. W. (2016). Concepts of programming languages. In Concepts of programming languages. Upper Saddle River: Pearson.

[3] ”Write once, run anywhere?”. Computer Weekly. May 2, 2002. Retrieved July 27, 2009.

[4] Jones, Robin; Maynard, Clive; Stewart, Ian (December 6, 2012). The Art of Lisp Programming. Springer Science & Business Media. p. 2. ISBN 9781447117193.

[5] John McCarthy. ”Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I”. Archived from the original on 2013-10-04. Retrieved 2006-10-13.

[6] 2021. ANSI INCITS 226-1994 (R1999) - Information Technology - Programming Language - Common Lisp (formerly ANSI X3.226-1994 (R1999)). Ansi.org. Retrieved March 10, 2021 from https://webstore.ansi.org/standards/incits/ansiincits2261994r1999

[7] Gary Knott. 2008. Interpreting LISP. Retrieved March 10, 2021 from http://www.civilized.com/files/lispbook.pdf