Background

The issue of how to develop a new programming language is a staple of computer science that has formalisms such as the T-Diagram (Aho 1986). Figure 10-1 shows a T-Diagram that depicts using a C compiler, which runs and creates PowerPC machine code, to compile a Pascal compiler written in C that creates PowerPC machine code, producing a compiler that runs with PowerPC machine code, creating PowerPC machine code.

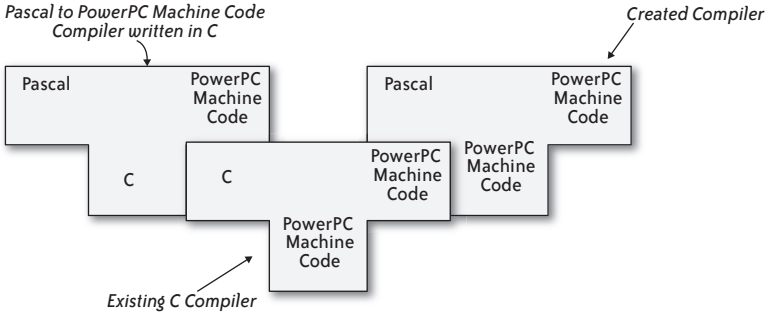

FIGURE 10-1. A T-Diagram showing the creation of a Pascal-to-PowerPC machine-code compiler

Unlike traditional programming languages, which compile down to the machine code for the computer upon which the program was intended to run, most modern languages can be compiled to an architecture-neutral machine code. In the case of Java, this is known as Java bytecode. A neutral machine code allows applications to be ported to any environment where the runtime is present. So, Java can be run anywhere a Java virtual machine is present.

Modern languages aim to help the programmer by designing out potential programming language pitfalls. The most prevalent feature is to have memory safety, by limiting what a programmer may do with data types and allowing only automatic garbage collection to release memory. Another feature is the ability to throw exceptions.

Self-hosting is seen as an important principle for programming languages. Self-hosting means that the programming language should allow enough expression that the programming language can be written in its own programming language. For example, a Pascal compiler written in Pascal is self-hosting, whereas a Pascal compiler written in C is not. Self-hosting allows the programming language developer to use the features of the programming language for which they are responsible. Critically, self-hosting creates a virtuous cycle in which language implementers desire to utilize advanced and/or expressive language features in performance-critical parts of the language implementation, and therefore often discover innovative ways to efficiently implement said language features.

Although making the compiler self-hosting is seen as important, many runtime environments are not written in the language in which they typically run. For example, a runtime written