GOLDBERG83

Goldberg,A., Robson,D.;
SMALLTALK-80 The Language and Its Implementation;
Addison-Wesley series in Computer Science (May 1983);
Part 4 (Ch. 26 - Ch. 30).

In this last part of the book, the underlying structure of the Smalltalk-80 system: the virtual machine which consists of an interpreter and an object memory is discussed.

The first chapter in this part described in detail how the system compiler creates an instance of the class CompileMethod to hold the bytecodes translated from user's source method. This instance also contains temporary/ shared variables used inside the method and something called literal frame (this may be considered as activation record of procedure in other programming languages). The bytecodes are eight-bit instructions for a stack-oriented interpreter. The actual mechanism that the interpreter used to handle message passing and context switching is also described in some detail.

The remaining chapters describe the virtual machine in terms of Smalltalk itself and gives a formal specification of the interpreter (the exact format of the bytecodes), the pimitive methods and the object memory.

Primitive methods are methods that the interpreter can respond directly without the need to pass it to another CompileMethod. Primitive methods fall into four categories: arithmetic, storage management, control and input/output.

The object memory provides the interpreter access to object in the system without worrying about the details concerning memory management. The task of the object memory among other things includes heap allocation/deallocation and garbage collection.

These are the aspects of the Smalltalk system which are hardware dependent: is it a 16 bits machine or 32 bit machine, how large is the address space ... etc. Thus the target readers of this part are people who want to know how the internals of Smalltalk work and its implementation, especially those who want to actually implement it.