In Chapter 8, native PROLOG was used to develop a complete knowledge-based system. Some of the implementation problems generated by the decision to limit ourselves to native PROLOG were discussed. In this chapter, we will look at how to use meta-PROLOG programming methods to build forest management KBS using a toolkit called DSSTOOLS. We will continue to use the Red Pine Forest Management Advisory System (RP-FMAS) introduced earlier as our implementation example, however, the focus is really on the toolkit. We have rewritten the original RP-FMAS using the PROLOG toolkit presented in this chapter. Readers unfamiliar with forest management theory and practice should review the background for RP-FMAS in section 4.3 and the knowledge model presented in section 6.1.1. Rauscher et al. (1990) and the

1 We want to acknowledge Donald E. Nute, Guojun Zhu, and Yousong Chang, members of the Artificial Intelligence Laboratory at the University of Georgia, for their contributions to the DSSTOOLS development project described in this chapter.
RP-FMAS hypertext document (see the Frontmatter for file location) may be consulted for additional detail.

The PROLOG toolkit approach, described in this chapter, makes the most sense for developers who expect to be involved in the development of a large and evolving KBS or who expect to be involved in the development of a series of systems in a variety of domains. On the other hand, for more casual experimentation in support of learning activities or for rapid prototyping exercises, the native Prolog programming approach described in Chapter 8 is probably more practical.

9.1 THE KBS TOOLKIT

Many authors have called for toolkit packages that can be tailored to specific applications with a minimum of new programming (Chandrasekaran 1986, Cherubini et al. 1989, Geneserth and Ginsberg 1985, Suzuki 1988). A few PROLOG toolkits have been reported in the literature (Oliveira 1984, Lee 1986). Unfortunately, none of these previously developed PROLOG KBS toolkits are readily available or in wide-spread use at the present time.

The Red Pine FMAS described in this chapter is a decision support system written in PROLOG using the DSSTOOLS toolkit. DSSTOOLS lets developers concentrate on solving the domain problem rather than on writing computer code by making available a library of reusable software code. The toolkit modules can be invoked with small amounts of additional code. They also encapsulate some of the more difficult components of an KBS.

Toolkits are typically associated with a programming language and require at least a moderate level of familiarity with that computer language. For example, the C Language Integrated Production System (CLIPS) toolkit, developed by NASA, is written in the C language and meant to be used within a C language program (Giarratano and Riley 1994). CLIPS is a forward-chaining, rule-based KBS development toolkit. It does not directly support backward chaining, frames, or object-oriented programming (Giarratano and Riley 1994). An important difference between CLIPS and DSSTOOLS is that it is easier to develop alternative inference engines for KBS applications in PROLOG than in C.

The toolkit approach is actually a variation of the native Prolog programming approach, used in Chapter 8, together with a particular design philosophy based on taking the long view about system development. The basic idea of the toolkit is that developers collect pieces of generic code that can be used repeatedly in the same or several systems. When two very similar tasks have to be performed, the toolkit design philosophy dictates that a generic routine should be developed that can do both tasks. The resulting routine is likely to be reusable. As more