The Java platform is becoming a vital tool for developing real-time and safety-critical systems. Design patterns and the availability of Java libraries, both provide solutions to many known problems. Furthermore, the object-oriented nature of Java simplifies modular development of real-time systems. However, limitations of Java as a programming language for real-time systems are a notable obstacle to producing efficient real-time systems. These limitations are found in the unpredictable execution model of the language, due to Java's garbage collector, and the lack of support for non-functional specification and verification tools. In this dissertation I introduce SafeJML, a specification language for support of functional and non-functional specifications, based on an implementation of a safety-critical Java platform and the Java Modeling Language (JML). This dissertation concentrates on techniques that enable specification and dynamic checking of timing constraints for some important Java features, including methods and subtyping. SafeJML and these dynamic checking techniques allow modular specification and checking of safety-critical systems, including those that use object-orientation and design patterns. Such coding techniques could have maintenance benefits for real-time and safety-critical software.