Traceability can be used to check that specified requirements have been satisfied in design and code, and help to assess and manage the impact of changing requirements, among many other things. The demands that are placed upon the traceability for end use, and thus the ability to use traceability effectively in practice, obviously varies dependent upon the context in which software systems are engineered and used.

This part of the book describes some typical development contexts and pinpoints their particular concerns. The chapter by Cleland-Huang, "Traceability in Agile Projects", reminds us that while successful traceability is always needs driven, there are sometimes leaner ways to achieve traceability goals in certain contexts. The chapter by De Borger et al. examines aspect-oriented software development, a context within which "Traceability Between Run-Time and Development Time Abstractions" becomes the all-important focus. The chapter by Mirakhorli and Cleland-Huang considers the benefits that can arise from "Tracing Non-Functional Requirements" and explains the associated complexity. Architectural
centric traceability is proposed as an approach for those contexts in which the quality of service is the foremost concern. Finally, the chapter by Mc Caffery et al. on “Medical Device Software Traceability” explains the special demands of tracing regulatory requirements in the medical device industry and illustrates a medical device traceability software process assessment method.