IV

Improving Environments

14. Methods
15. Maturity and CASE
16. Managing New Tools
17. Software Factories

While the first three parts of the book focus on how to understand and support professional practices the final two parts are primarily concerned with improvement of practices. Part V addresses professional development in general and part IV looks more specifically at the following question: How can we improve the environments that govern systems development practice? I introduce four of my own contributions that can help answer this question and summarize other contributions of my Danish colleagues aimed at improving systems development environments (see table 3 and reference list in chapter 1). Environments are webs of conditions and factors that shape systems development practices (Kling *et al.* 1982). They exert influence on the process, its organization, and the outcomes and they include labor, economy, technology, business application, organization, and legal institutions (Lyytinen 1987a). I agree with Brooks that technical innovations cannot solve the intrinsic problems related to systems development (1987). But the technological environment plays a distinct role in shaping and improving systems development practices. The contributions in part IV concentrate on this environment emphasizing the practical use of methods and tools.

Chapter 14 discusses the role of systems development methods in improving practices. Based on our experiences from developing an object oriented analysis and design method (Mathiassen *et al.* 1993, 1995, 1997) we suggest that the primary role of methods is to support learning. To be useful in practice, methods need to be

adapted, modified, and combined with other methods and experiences (cf. chapter 5).

The next two chapters address the use of software engineering tools. Chapter 15 reviews the Capability Maturity Model (CMM) (Humphrey 1989; Paulk *et al.* 1993) as a framework for CASE introduction. We explicate its strengths and weaknesses by exploring three issues: What is the role of organizational experiments in CASE introduction? How do the functional characteristics of CASE technology influence CASE introduction? How does the specific organizational context influence CASE introduction?

The resulting findings are developed further in chapter 16 into practical guidelines that can help managers design and combine initiatives to take new software engineering tools into use. We discuss the questions that need to be addressed and we outline possible decisions and rationales involved in answering these questions. The assumption underlying these two chapters is that it is a complex and uncertain process to make use of new software engineering tools. It requires, like systems development in general, a reflective approach in which experiments are combined with more rational and structured approaches.

Chapter 17 discusses a controversial idea related to the technological and organizational environments of systems development: the software factory. Factories are, in general, ambitious approaches to use technology to boost and improve production. Even though software applications cannot be mass-produced like other products, a number of factory initiatives have emerged over the past 30 years to improve the productivity and quality in systems development. We review and compare four well-known factory, or factory-like, initiatives and we discuss the possible contributions and illusions involved in these ambitious approaches to improve practice in our field.

In addition to the contributions contained in this book, (Nielsen 1990a) presents a reflective approach to learning and using systems development methods and parts of this research are published in (Nielsen 1989b, 1990b); (Bang *et al.* 1991) offers principles and guidelines for quality management systems in IT organizations; and quite a number of contributions address the adaptation and practical use of CASE tools (Aaen *et al.* 1991, 1992; Aaen 1992, 1994; Sørensen 1993).