

\* concrete experience with the new system (area 6).

It goes without saying that the developers must understand abstract descriptions of the new system (area 5) since they are major intermediate results. Relevant structures on users' present work (area 2) must be understood in order to identify and evaluate desirable changes.

The developers must have concrete experience with users' present work (area 1) in order to understand and produce descriptions of relevant structures on the users' present work. System developers who have developed this area of knowledge have a better background for communicating with the users, as they are able to refer to and understand references to concrete events in the users' organization.

Finally, the developers must have concrete experience with the new system (area 6) in order to be able to test and evaluate the products of their own work.

We conclude this section with the observation that our theory recognizes that all areas of knowledge must be dealt with in any normal system development process. The toolbox necessary for this work is discussed briefly in the following section.

### Tools and Techniques for Knowledge Development

Our model of user-developer communication can be used to define a toolbox for tools and techniques to facilitate this communication. The toolbox presented in Table 2 consists of 6 sections, one for each of the areas of knowledge discussed in the previous section. It accentuates the differences between the purposes of the tools and techniques, even though some fit into more sections.

The use of any tool or technique must be adapted to the particular conditions of each system development process. In particular, the users' prior experience with the tools and techniques must be considered. Some tools and techniques are almost always used successfully by developers and users working together. Others should rather be part of an extended process, in which developers

## STEPS—A Methodical Approach to PD

Christiane Floyd  
UNIVERSITY OF HAMBURG

**S**oftware Technology for Evolutionary Participatory System Design—STEPS—is a methodological framework for developing interactive application systems. It guides developers and users in carrying out their cooperation as well as supports the design of computer-based work. STEPS emphasizes the development process and intertwines development with use.

The method has been tried successfully in participatory development of an information system called the PETS project, which is designed for handling archives of union contracts in Germany. This participatory project resulted in a working system that is still in use today.

STEPS considers the anticipation of use an inherent part of the development. Whenever software systems—as tools or media—are to be fitted into work processes, development consists in unfolding the problem as well as in elaborating a solution. The technical concerns for providing high-quality products are inherently tied up with issues of communication, work, and social processes, which define the very nature of the problem.

Therefore, software development becomes a learning process for both developers and users. In STEPS, *evolution* refers to the emergence of insights into the functionality and the potential use of software. *Participation* refers to the strategy of mutual learning. Those participating in a software development project are creating a product, and at the same time, the development process itself. These two complementary dimensions are reflected by product-oriented and process-oriented activities. STEPS provides guidance to developers and users in both dimensions.

This approach relies on perspective for gaining insight—making perspectives explicit and allowing them to interact. The use perspective held by those who interact with software is distinct from the development perspective held by software developers. Furthermore, they both arise in a variety of views related to functional roles, collective interests, and individual tastes. Multiperspectivity is a fundamental prerequisite for cooperative work which rests on perspective-based

modeling and evaluation when using software. Software requirements are related to the context of user work processes as a whole. They cannot be completely fixed in advance as they evolve because of changes in the organization. Also, the software system itself gives rise to new requirements.

Thus, STEPS is an approach that accommodates various forms of prototyping and portraying system development in cycles of version production, application and revision. A system version consists of software and its defining documents, supplemented by guidelines for the organization of work to be supported. The interplay between each software version and the associated reorganization of user work is anticipated in the cooperative design and evaluated in the revision step. Thus, it supports mutual learning by developers and users by carefully establishing and coordinating processes of cooperation.

STEPS rests on a project model suitable for PD. This model is cyclical, all development steps and products being subject to revision. It combines software production and application, visualizing the tasks of both developers and users. It allows the choice of a situation-specific strategy in the actual project. It relies on a minimum of predefined intermediate products, thus allowing freedom for choosing them as needed. It provides for temporal flexibility in cooperation. It incorporates the dynamic coordination of the ongoing project through establishment and reference lines. STEPS embodies a human-centered notion of quality and creates a platform for discussing criteria on how to design computer-supported work and facilitates the emergence of quality through cooperative design. □

#### References

- Floyd, C., Reisin, F.-M., Schmidt, G. STEPS to Software Development with Users. C. Ghezzi, J.A. McDermid, Eds.: ESEC '88, Lecture Notes in Computer Science Nr. 387, Springer-Verlag, 1989, pp. 48–64.
- Floyd, C., Mehl, M., Reisin, F.-M., Schmidt, G., Wolf, G. Projekt PETS: Partizipative Entwicklung transparenter Software fuer EDV-gest. tztbe Arbeits-pl Arbeit, Gesundheit und Soziales des Landes Nordrhein-Westfalen, Technical University of Berlin, 1990.
- Floyd, C., Zilghoven, H., Budde, R., Kell-Slawik, R., Eds. Software Development and Reality Construction. Springer Verlag, 1992.