

Translation between Software Designers and Users

Marian G. Williams
UNIVERSITY OF MASSACHUSETTS, LOWELL
Vivienne Begg
SUNSOFT, INC.
CHELMSFORD, MASS.

When asked how they create for an unfamiliar domain, software designers typically answer: "Either I have to learn enough about what the users do to be able to tell them what they want, or they have to learn enough about computers to tell me." Several assumptions underlie this response. One is that software design is concerned primarily with the user's task, in isolation from the workplace context in which it is performed. Another is that either the engineer or the user will have the opportunity, resources, knowledge, and skills to learn enough about the other's domain to engage in useful dialogue.

We have developed the notion of *translation* between designers and users to address this second assumption. PD requires effective communication between individuals with different kinds of training, different goals, different languages, and different workplace cultures. Users cannot be expected to describe their work and needs in the language and from the point of view of an engineer. Conversely, engineers seldom have an intuitive grasp of their users' working life, or of the environment in which a product will be used. Some kind of two-way translation is needed between the user's domain and the software designer's. Here we explore the role that can be played by a third party who serves as translator between the users and the engineers.

For dialogue between a user and an engineer to be meaningful, the discourse must include models that both parties can understand. These models are metaphors which act as a bridge between one domain and another [2]. A good translator can create metaphors that are meaningful both to users and to engineers. By enabling the user to participate in design, the translator may also be able to overcome the user's reluctance to confide in a developer and trust him or her to turn privileged information into a useful tool.

We have had the opportunity to work

both with and as translators. Our study of translation in PD focuses on our own experiences, as well as on the experiences of participants in a workshop we recently gave [5,6]. We are performing retrospective analyses of several design efforts that were somewhat extreme in their need for translation. It seems unlikely that these efforts would have succeeded without the translating services of a person with work experience both in the user's domain and in software design. The users in our case studies are as diverse as physicists, English teachers [3,4], psychiatric nurses, and CAD engineers [1].

Translation between high school teachers and software engineers. One of our case studies involves the adaptation of a commercial newspaper editorial system for use in the secondary classroom. An editorial system is the networked hardware and software used by reporters and editors at a newspaper. It is highly customized for writing and editing, and fosters cooperative work. A 22-seat editorial system, donated by Atex Publishing Systems in Massachusetts, was installed in the public high school in a middle-class suburb of Boston.

The project began with the premise that teachers and students could interact the way editors and writers do. However, the flow of information in the English classroom is substantially different from the flow of information in a newsroom. The editorial system required extensive customization. None of the teachers knew enough about hardware or software to participate effectively in the customization process, and they were not given release time to work on the customizations. The engineers' experience was in tailoring editorial systems for specific newspaper sites. They had general assumptions about the teachers' workplace, but were not aware of specific activities and conventions. Their time was donated to perform customizations, not to design them.

The customizations were designed by a former English teacher turned computer scientist, who enabled the high school teachers to participate in the customization process and who translated their needs into specifications for the software engineers. The translation process had these steps:

- (1) Elicit from the teachers a description of writing-related activities currently used in their paper-based classrooms.
- (2) Determine how these activities could

be carried out and extended in a computer-based classroom.

- (3) Describe the extensions to the teachers, and see if they wanted them.
- (4) Translate the descriptions of the activities from the language and workplace conventions of the teachers to the language and conventions of the engineers.
- (5) Work with the engineers to specify the customizations to the editorial system.

One of the customizations concerned versioning of files. In a newsroom, there is one current version of a story. Either the editor has it or the writer has it. When one of them sends the story to the other, the story disappears entirely from the sender's workspace. Earlier versions are archived, in a manner that is invisible to the editor and writer, in a *global workspace and are rarely retrieved*. By contrast, in an English classroom where *process writing*, is taught, students are expected to maintain a library of drafts of an essay. When a student submits a draft to the teacher, the student should retain all earlier versions, and a copy of the current version, in his or her workspace. No student's work should be kept in a global archive because of the temptation of plagiarism. The versioning capabilities of the editorial system were customized to reflect these differences. Whenever a student submits a paper to the teacher, a new version number is automatically assigned, and all drafts are archived in the student's own workspace.

The translator was vital to this design effort, since the users and the engineers had neither sufficient knowledge of one another's work and workplace, nor the resources to develop such knowledge. The translator was accepted by both users and engineers as a somebody who spoke their language and who knew how their work responsibilities were being affected by the project. For more detail, see [3,4].

Translation between psychiatric nurses and software developers. Another of our case studies involves the development of software for teaching psychiatric nurses how to deal with volatile situations involving potentially violent patients. The users were nurses in a mental hospital in England. The hospital had 1,300 patients of all ages and types, and a staff of 2,600 nurses who worked 12-hour shifts, on alternating weeks.

The nurses worked in difficult and sometimes dangerous conditions with

little training. Despite great pressure, both social and political, from outside the hospital to "return patients to the community," nurses were the heirs of a tradition many centuries old, of custodial rather than therapeutic care. This project was championed by the Chief Nursing Officer (CNO) and the Principal Psychologist, both young and enthusiastic about modern methods.

The software was intended to sensitize nurses to crisis situations in which they could easily fall back onto the old ways of restraining and tranquilizing patients. The program led them step-by-step through situations based on actual events. As the nurse determined what to do, a trainer was present to talk about different strategies. The CNO judged the software to be successful. The nurses reported that the quiet off-ward atmosphere and the novel challenge of computer interaction allowed them to think through events that would have been traumatic if they happened in real time.

Outsiders were not accepted at the hospital for two reasons: physical danger and the insularity of the nursing staff, who form a closed community. Only someone who had proved to be a competent and resourceful professional would be accepted by them. This competence included a full understanding of everyday situations on a ward, of mental illness and of institutions, of how to stay out of trouble and stay in a situation in order to observe it, and of workplace politics.

The translator was a clinical psychology intern who was known and trusted by the nurses, and who met their criteria for competence. The translator was also a programmer. Because the translator was an insider, the nurses participated willingly in the design of the software. The situations described in the program were real, because they were transcribed from videotapes taken in the wards. The nurses provided multiple choice responses to the situations. They believed that the training was not a test, and participated willingly both in the making of the tool and in its use.

Characteristics of translation

Whenever software is designed for particular end users, translation happens. If the user and designer cannot perform the translation themselves, a third party must be enlisted. In the extreme cases that we are studying, the translator has experience both in the user's domain and in software development. However,

in other circumstances, translation may be performed by a software designer, a user interface designer, an ethnographer, a systems analyst, or a recognized expert in the user's field. A translator is an active participant in the design process. Although an ethnographer can be a translator, translation involves intervention in the design process, which is more than the ethnographer's observer status usually allows.

Translation is not solely concerned with interpretation of tasks and terminology, but also with workplace culture and politics. Therefore, it may include negotiation or facilitation. In fact, the translator may perform translation between the design group and one or more outside entities (such as management), as well as within the design group. The role begins to resemble politician, lobbyist or diplomat, since adversarial relationships may exist between groups.

The translator role deserves recognition and reward. It is arduous and can be performed well only by a few people with exceptional qualities—qualities that are context-dependent, not constant across all design efforts. The translator role should be recognized and rewarded, but not institutionalized. Making "translator" a job description would foster rigidity in the translation process.

Translation happens whenever soft-

ware is written for a particular workplace. Sometimes the users and developers can perform this translation themselves. In other circumstances, they cannot because the users do not have the knowledge to participate in design and moreover do not have the resources (time, finances, education, confidence) to develop such knowledge, and the developers do not have the resources (time, expertise) to study the users' workplace. Our case studies lead us to believe that career-changers who become software developers after working in another field are a largely untapped resource for translation. The translator straddles two (or more) domains of discourse, and has to be attuned to the social, cultural and political issues to ensure product's successful introduction into the workplace.



References

1. Begg, V. *Developing Expert Systems for CAD*. (New York: N.Y.), 1985.
2. Lakoff, G. and Johnson, M. *Metaphors We Live By*. Chicago: University of Chicago Press, 1980.
3. Williams, M.G. A Case Study in Translation in Participatory Design: High School Teachers, University of Massachusetts Lowell, Center for Productivity Enhancement, Tech. Rep. CPE-92-001, 1992.
4. Williams, M.G. Adapting a Commercial Newspaper Editorial System for Teaching and Learning. *Proceedings of the Association for Media and Technology in Education Conference*, June 13-16, 1993. To be published.
5. Williams, M.G. and Begg, V. Translation in Participatory Design. *Proceedings of the Conference on Participatory Design (PDC '92)*, Nov. 6-7, 1992, pp. 113-14.
6. Williams, M.G. and Begg, V. Translation in Participatory Design: Lessons from a Workshop. *INTERCHI '93 Adjunct Proceedings*, April 24-29, 1993. To be published.

For more information and/or participation:

Computer Professionals for
Social Responsibility (CPSR):
P.O. Box 717, Palo Alto, CA 94301, US. (Proceedings of PDC'90 and PDC'92 are available from CPSR.)
PDC'94: *William Anderson (Conference chair), Xerox Corporation 817-03A, 295 Woodcliff Drive, Fairport, NY 14450*