Moments after announcing the need to study the sociology of human-to-human, computer-mediated software, Clay Shirky brilliantly gets busy, drilling down into the specific phenomenon of flaming, reminiscent of the great ethnographies of yore, more like Margaret Mead’s studies of the sexual habits of Pacific Islanders than a typical software paper. – Ed.

When we hear the word “software,” most of us think of things like Word, PowerPoint, or Photoshop, tools for individual users. These tools treat the computer as a box, a self-contained environment in which the user does things. Much of the current literature and practice of software design—feature requirements, UI design, usability testing—targets the individual user, functioning in isolation.

And yet, when we poll users about what they actually do with their computers, some form of social interaction always tops the list—conversation, collaboration, playing games together, and so on. The practice of


2. “Flaming” refers to criticizing someone strongly online, for example through email or on discussion groups. – Ed.
software design is shot through with computer-as-box assumptions, while our actual behavior is closer to computer-as-door, treating the device as an entrance to a social space.

We have grown quite adept at designing interfaces and interactions between people and machines, but our social tools—the people-to-people software the users actually use most often—remain badly mis-fit to their task. Social interactions are far more complex and unpredictable than human/computer interaction, and that unpredictability defeats classic user-centric design. As a result, tools used daily by tens of millions are either ignored as design challenges, or treated as if the only possible site of improvement is the user-to-tool interface.

The design gap between computer-as-box and computer-as-door persists because of a diminished conception of the user. The user of a piece of social software is not just a collection of individuals but a group. Individual users take on roles that only make sense in groups: leader, follower, peacemaker, process enforcer, and so on. There are also behaviors that can only occur in groups, from consensus building to social climbing. And yet, despite these obvious differences between personal and social behaviors, we have very little design practice that treats the group as an entity to be designed for.

There is enormous value to be gotten in closing that gap, and it doesn’t require complicated new tools. It just requires new ways of looking at old problems. Indeed, much of the most important recent work in social software has been technically simple but socially complex.

≈

Learning from Flame Wars

Mailing lists were the first widely available piece of social software. (The PLATO system beat mailing lists by a decade, but had a limited user base.) Mailing lists were also the first widely analyzed virtual communities. And for roughly 30 years, almost any description of mailing lists of any length has mentioned flaming, the tendency of list members to forgo standards of public decorum when attempting to communicate with some ignorant moron whose to stupid to know how too spell and deserves to DIE, die a PAINFUL DEATH, you PINKO SCUMBAG!!!