A Group Is Its Own Worst Enemy

By Clay Shirky

A speech at ETech, April, 2003

Published July 1, 2003 on the "Networks, Economics, and Culture" mailing list.

This is a lightly edited version of the keynote I gave on Social Software at the O'Reilly Emerging Technology conference in Santa Clara on April 24, 2003

Good morning, everybody. I want to talk this morning about social software ...there's a surprise. I want to talk about a pattern I've seen over and over again in social software that supports large and long-lived groups. And that pattern is the pattern described in the title of this talk: "A Group Is Its Own Worst Enemy."

In particular, I want to talk about what I now think is one of the core challenges for designing large-scale social software. Let me offer a definition of social software, because it's a term that's still fairly amorphous. My definition is fairly simple: It's software that supports group interaction. I also want to emphasize, although that's a fairly simple definition, how radical that pattern is. The Internet supports lots of communications patterns, principally point-to-point and two-way, one-to-many outbound, and many-to-many two-way.

Prior to the Internet, we had lots of patterns that supported point-to-point two-way. We had telephones, we had the telegraph. We were familiar with technological mediation of those kinds of conversations. Prior to the Internet, we had lots of patterns that supported one-way outbound. I could put something on television or the radio, I could publish a newspaper. We had the printing press. So although the Internet does good things for those patterns, they're patterns we knew from before.

Prior to the Internet, the last technology that had any real effect on the way people sat down and talked together was the table. There was no technological mediation for group conversations. The closest we got was the conference call, which never really worked right -- "Hello? Do I push this button now? Oh, shoot, I just hung up." It's not easy to set up a conference call, but it's very easy to email five of your friends and say "Hey, where are we going for pizza?" So ridiculously easy group forming is really news.

We've had social software for 40 years at most, dated from the Plato BBS system, and we've only had 10 years or so of widespread availability, so we're just finding out what works. We're still learning how to make these kinds of things.

Now, software that supports group interaction is a fundamentally unsatisfying definition in many ways, because it doesn't point to a specific class of technology. If you look at email, it obviously supports social patterns, but it can also support a broadcast pattern. If I'm a spammer, I'm going to mail things out to a million people, but they're not going to be talking to one another, and I'm not going to be

talking to them -- spam is email, but it isn't social. If I'm mailing you, and you're mailing me back, we're having point-to-point and two-way conversation, but not one that creates group dynamics.

So email doesn't necessarily support social patterns, group patterns, although it can. Ditto a weblog. If I'm Glenn Reynolds, and I'm publishing something with Comments Off and reaching a million users a month, that's really broadcast. It's interesting that I can do it as a single individual, but the pattern is closer to MSNBC than it is to a conversation. If it's a cluster of half a dozen LiveJournal users, on the other hand, talking about their lives with one another, that's social. So, again, weblogs are not necessarily social, although they can support social patterns.

Nevertheless, I think that definition is the right one, because it recognizes the fundamentally social nature of the problem. Groups are a run-time effect. You cannot specify in advance what the group will do, and so you can't substantiate in software everything you expect to have happen.

Now, there's a large body of literature saying "We built this software, a group came and used it, and they began to exhibit behaviors that surprised us enormously, so we've gone and documented these behaviors." Over and over and over again this pattern comes up. (I hear Stewart [Brand, of the WELL] laughing.) The WELL is one of those places where this pattern came up over and over again.

This talk is in three parts. The best explanation I have found for the kinds of things that happen when groups of humans interact is psychological research that predates the Internet, so the first part is going to be about W.R. Bion's research, which I will talk about in a moment, research that I believe explains how and why a group is its own worst enemy.

The second part is: Why now? What's going on now that makes this worth thinking about? I think we're seeing a revolution in social software in the current environment that's really interesting.

And third, I want to identify some things, about half a dozen things, in fact, that I think are core to any software that supports larger, long-lived groups.

Part One: How is a group its own worst enemy?

So, Part One. The best explanation I have found for the ways in which this pattern establishes itself, the group is its own worst enemy, comes from a book by W.R. Bion called "Experiences in Groups," written in the middle of the last century.

Bion was a psychologist who was doing group therapy with groups of neurotics. (Drawing parallels between that and the Internet is left as an exercise for the reader.) The thing that Bion discovered was that the neurotics in his care were, as a group, conspiring to defeat therapy.

There was no overt communication or coordination. But he could see that whenever he would try to do anything that was meant to have an effect, the group would somehow quash it. And he was driving himself crazy, in the colloquial sense of the term, trying to figure out whether or not he should be looking at the situation as: Are these individuals taking action on their own? Or is this a coordinated group?

He could never resolve the question, and so he decided that the unresolvability of the question was the answer. To the question: Do you view groups of people as aggregations of individuals or as a cohesive group, his answer was: "Hopelessly committed to both."

He said that humans are fundamentally individual, and also fundamentally social. Every one of us has a kind of rational decision-making mind where we can assess what's going on and make decisions and act on them. And we are all also able to enter viscerally into emotional bonds with other groups of people that transcend the intellectual aspects of the individual.

In fact, Bion was so convinced that this was the right answer that the image he put on the front cover of his book was a Necker cube, one of those cubes that you can look at and make resolve in one of two ways, but you can never see both views of the cube at the same time. So groups can be analyzed both as collections of individuals and having this kind of emotive group experience.

Now, it's pretty easy to see how groups of people who have formal memberships, groups that have been labeled and named like "I am a member of such-and-such a guild in a massively multi-player online role-playing game," it's easy to see how you would have some kind of group cohesion there. But Bion's thesis is that this effect is much, much deeper, and kicks in much, much sooner than many of us expect. So I want to illustrate this with a story, and to illustrate the illustration, I'll use a story from your life. Because even if I don't know you, I know what I'm about to describe has happened to you.

You are at a party, and you get bored. You say "This isn't doing it for me anymore. I'd rather be someplace else. I'd rather be home asleep. The people I wanted to talk to aren't here." Whatever. The party fails to meet some threshold of interest. And then a really remarkable thing happens: You don't leave. You make a decision "I don't like this." If you were in a bookstore and you said "I'm done," you'd walk out. If you were in a coffee shop and said "This is boring," you'd walk out.

You're sitting at a party, you decide "I don't like this; I don't want to be here." And then you don't leave. That kind of social stickiness is what Bion is talking about.

And then, another really remarkable thing happens. Twenty minutes later, one person stands up and gets their coat, and what happens? Suddenly everyone is getting their coats on, all at the same time. Which means that everyone had decided that the party was not for them, and no one had done anything about it, until finally this triggering event let the air out of the group, and everyone kind of felt okay about leaving.

This effect is so steady it's sometimes called the paradox of groups. It's obvious that there are no groups without members. But what's less obvious is that there are no members without a group. Because what would you be a member of?

So there's this very complicated moment of a group coming together, where enough individuals, for whatever reason, sort of agree that something worthwhile is happening, and the decision they make at that moment is: This is good and must be protected. And at that moment, even if it's subconscious, you start getting group effects. And the effects that we've seen come up over and over and over again in online communities.

Now, Bion decided that what he was watching with the neurotics was the group defending itself against his attempts to make the group do what they said they were supposed to do. The group was convened to get better, this group of people was in therapy to get better. But they were defeating that. And he said, there are some very specific patterns that they're entering into to defeat the ostensible purpose of the group meeting together. And he detailed three patterns.

The first is sex talk, what he called, in his mid-century prose, "A group met for pairing off." And what that means is, the group conceives of its purpose as the hosting of flirtatious or salacious talk or emotions passing between pairs of members.

You go on IRC and you scan the channel list, and you say "Oh, I know what that group is about, because I see the channel label." And you go into the group, you will also almost invariably find that it's about sex talk as well. Not necessarily overt. But that is always in scope in human conversations, according to Bion. That is one basic pattern that groups can always devolve into, away from the sophisticated purpose and towards one of these basic purposes.

The second basic pattern that Bion detailed: The identification and vilification of external enemies. This is a very common pattern. Anyone who was around the Open Source movement in the mid-Nineties could see this all the time. If you cared about Linux on the desktop, there was a big list of jobs to do. But you could always instead get a conversation going about Microsoft and Bill Gates. And people would start bleeding from their ears, they would get so mad.

If you want to make it better, there's a list of things to do. It's Open Source, right? Just fix it. "No, no, Microsoft and Bill Gates grrrrr ...", the froth would start coming out. The external enemy -- nothing causes a group to galvanize like an external enemy.

So even if someone isn't really your enemy, identifying them as an enemy can cause a pleasant sense of group cohesion. And groups often gravitate towards members who are the most paranoid and make them leaders, because those are the people who are best at identifying external enemies.

The third pattern Bion identified: Religious veneration. The nomination and worship of a religious icon or a set of religious tenets. The religious pattern is, essentially, we have nominated something that's beyond critique. You can see this pattern on the Internet any day you like. Go onto a Tolkein newsgroup or discussion forum, and try saying "You know, The Two Towers is a little dull. I mean loooong. We didn't need that much description about the forest, because it's pretty much the same forest all the way."

Try having that discussion. On the door of the group it will say: "This is for discussing the works of Tolkein." Go in and try and have that discussion.

Now, in some places people say "Yes, but it needed to, because it had to convey the sense of lassitude," or whatever. But in most places you'll simply be flamed to high heaven, because you're interfering with the religious text.

So these are human patterns that have shown up on the Internet, not because of the software, but because it's being used by humans. Bion has identified this possibility of groups sandbagging their sophisticated goals with these basic urges. And what he finally came to, in analyzing this tension, is

that group structure is necessary. Robert's Rules of Order are necessary. Constitutions are necessary. Norms, rituals, laws, the whole list of ways that we say, out of the universe of possible behaviors, we're going to draw a relatively small circle around the acceptable ones.

He said the group structure is necessary to defend the group from itself. Group structure exists to keep a group on target, on track, on message, on charter, whatever. To keep a group focused on its own sophisticated goals and to keep a group from sliding into these basic patterns. Group structure defends the group from the action of its own members.

In the Seventies -- this is a pattern that's shown up on the network over and over again -- in the Seventies, a BBS called Communitree launched, one of the very early dial-up BBSes. This was launched when people didn't own computers, institutions owned computers.

Communitree was founded on the principles of open access and free dialogue. "Communitree" -- the name just says "California in the Seventies." And the notion was, effectively, throw off structure and new and beautiful patterns will arise.

And, indeed, as anyone who has put discussion software into groups that were previously disconnected has seen, that does happen. Incredible things happen. The early days of Echo, the early days of usenet, the early days of Lucasfilms Habitat, over and over again, you see all this incredible upwelling of people who suddenly are connected in ways they weren't before.

And then, as time sets in, difficulties emerge. In this case, one of the difficulties was occasioned by the fact that one of the institutions that got hold of some modems was a high school. And who, in 1978, was hanging out in the room with the computer and the modems in it, but the boys of that high school. And the boys weren't terribly interested in sophisticated adult conversation. They were interested in fart jokes. They were interested in salacious talk. They were interested in running amok and posting four-letter words and nyah-nyah-nyah, all over the bulletin board.

And the adults who had set up Communitree were horrified, and overrun by these students. The place that was founded on open access had too much open access, too much openness. They couldn't defend themselves against their own users. The place that was founded on free speech had too much freedom. They had no way of saying "No, that's not the kind of free speech we meant."

But that was a requirement. In order to defend themselves against being overrun, that was something that they needed to have that they didn't have, and as a result, they simply shut the site down.

Now you could ask whether or not the founders' inability to defend themselves from this onslaught, from being overrun, was a technical or a social problem. Did the software not allow the problem to be solved? Or was it the social configuration of the group that founded it, where they simply couldn't stomach the idea of adding censorship to protect their system. But in a way, it doesn't matter, because technical and social issues are deeply intertwined. There's no way to completely separate them.

What matters is, a group designed this and then was unable, in the context they'd set up, partly a technical and partly a social context, to save it from this attack from within. And attack from within is what matters. Communitree wasn't shut down by people trying to crash or syn-flood the server. It was

shut down by people logging in and posting, which is what the system was designed to allow. The technological pattern of normal use and attack were identical at the machine level, so there was no way to specify technologically what should and shouldn't happen. Some of the users wanted the system to continue to exist and to provide a forum for discussion. And other of the users, the high school boys, either didn't care or were actively inimical. And the system provided no way for the former group to defend itself from the latter.

Now, this story has been written many times. It's actually frustrating to see how many times it's been written. You'd hope that at some point that someone would write it down, and they often do, but what then doesn't happen is other people don't read it.

The most charitable description of this repeated pattern is "learning from experience." But learning from experience is the worst possible way to learn something. Learning from experience is one up from remembering. That's not great. The best way to learn something is when someone else figures it out and tells you: "Don't go in that swamp. There are alligators in there."

Learning from experience about the alligators is lousy, compared to learning from reading, say. There hasn't been, unfortunately, in this arena, a lot of learning from reading. And so, lessons from Lucasfilms' Habitat, written in 1990, reads a lot like Rose Stone's description of Communitree from 1978.

This pattern has happened over and over and over again. Someone built the system, they assumed certain user behaviors. The users came on and exhibited different behaviors. And the people running the system discovered to their horror that the technological and social issues could not in fact be decoupled.

There's a great document called "LambdaMOO Takes a New Direction," which is about the wizards of LambdaMOO, Pavel Curtis's Xerox PARC experiment in building a MUD world. And one day the wizards of LambdaMOO announced "We've gotten this system up and running, and all these interesting social effects are happening. Henceforth we wizards will only be involved in technological issues. We're not going to get involved in any of that social stuff."

And then, I think about 18 months later -- I don't remember the exact gap of time -- they come back. The wizards come back, extremely cranky. And they say: "What we have learned from you whining users is that we can't do what we said we would do. We cannot separate the technological aspects from the social aspects of running a virtual world.

"So we're back, and we're taking wizardly fiat back, and we're going to do things to run the system. We are effectively setting ourselves up as a government, because this place needs a government, because without us, the place was falling apart."

People who work on social software are closer in spirit to economists and political scientists than they are to people making compilers. They both look like programming, but when you're dealing with groups of people as one of your run-time phenomena, that is an incredibly different practice. In the political realm, we would call these kinds of crises a constitutional crisis. It's what happens when the

tension between the individual and the group, and the rights and responsibilities of individuals and groups, gets so serious that something has to be done.

And the worst crisis is the first crisis, because it's not just "We need to have some rules." It's also "We need to have some rules for making some rules." And this is what we see over and over again in large and long-lived social software systems. Constitutions are a necessary component of large, long-lived, heterogenous groups.

Geoff Cohen has a great observation about this. He said "The likelihood that any unmoderated group will eventually get into a flame-war about whether or not to have a moderator approaches one as time increases." As a group commits to its existence as a group, and begins to think that the group is good or important, the chance that they will begin to call for additional structure, in order to defend themselves from themselves, gets very, very high.

Part Two: Why now?

If these things I'm saying have happened so often before, have been happening and been documented and we've got psychological literature that predates the Internet, what's going on now that makes this important?

I can't tell you precisely why, but observationally there is a revolution in social software going on. The number of people writing tools to support or enhance group collaboration or communication is astonishing.

The web turned us all into size queens for six or eight years there. It was loosely coupled, it was stateless, it scaled like crazy, and everything became about How big can you get? "How many users does Yahoo have? How many customers does Amazon have? How many readers does MSNBC have?" And the answer could be "Really a lot!" But it could only be really a lot if you didn't require MSNBC to be answering those readers, and you didn't require those readers to be talking to one another.

The downside of going for size and scale above all else is that the dense, interconnected pattern that drives group conversation and collaboration isn't supportable at any large scale. Less is different -- small groups of people can engage in kinds of interaction that large groups can't. And so we blew past that interesting scale of small groups. Larger than a dozen, smaller than a few hundred, where people can actually have these conversational forms that can't be supported when you're talking about tens of thousands or millions of users, at least in a single group.

We've had things like mailing lists and BBSes for a long time, and more recently we've had IM, we've had these various patterns. And now, all of a sudden, these things are popping up. We've gotten weblogs and wikis, and I think, even more importantly, we're getting platform stuff. We're getting RSS. We're getting shared Flash objects. We're getting ways to quickly build on top of some infrastructure we can take for granted, that lets us try new things very rapidly.

I was talking to Stewart Butterfield about the chat application they're trying here. I said "Hey, how's that going?" He said: "Well, we only had the idea for it two weeks ago. So this is the launch." When you can go from "Hey, I've got an idea" to "Let's launch this in front of a few hundred serious geeks

and see how it works," that suggests that there's a platform there that is letting people do some really interesting things really quickly. It's not that you couldn't have built a similar application a couple of years ago, but the cost would have been much higher. And when you lower costs, interesting new kinds of things happen.

So the first answer to Why Now? is simply "Because it's time." I can't tell you why it took as long for weblogs to happen as it did, except to say it had absolutely nothing to do with technology. We had every bit of technology we needed to do weblogs the day Mosaic launched the first forms-capable browser. Every single piece of it was right there. Instead, we got Geocities. Why did we get Geocities and not weblogs? We didn't know what we were doing.

One was a bad idea, the other turns out to be a really good idea. It took a long time to figure out that people talking to one another, instead of simply uploading badly-scanned photos of their cats, would be a useful pattern.

We got the weblog pattern in around '96 with Drudge. We got weblog platforms starting in '98. The thing really was taking off in 2000. By last year, everyone realized: Omigod, this thing is going mainstream, and it's going to change everything.

The vertigo moment for me was when Phil Gyford launched the Pepys weblog, Samuel Pepys' diaries of the 1660's turned into a weblog form, with a new post every day from Pepys' diary. What that said to me was: Phil was asserting, and I now believe, that weblogs will be around for at least 10 years, because that's how long Pepys kept a diary. And that was this moment of projecting into the future: This is now infrastructure we can take for granted.

Why was there an eight-year gap between a forms-capable browser and the Pepys diaries? I don't know. It just takes a while for people to get used to these ideas.

So, first of all, this is a revolution in part because it is a revolution. We've internalized the ideas and people are now working with them. Second, the things that people are now building are web-native.

When you got social software on the web in the mid-Nineties, a lot of it was: "This is the Giant Lotus Dreadnought, now with New Lightweight Web Interface!" It never felt like the web. It felt like this hulking thing with a little, you know, "Here's some icons. Don't look behind the curtain."

A weblog is web-native. It's the web all the way in. A wiki is a web-native way of hosting collaboration. It's lightweight, it's loosely coupled, it's easy to extend, it's easy to break down. And it's not just the surface, like oh, you can just do things in a form. It assumes http is transport. It assumes markup in the coding. RSS is a web-native way of doing syndication. So we're taking all of these tools and we're extending them in a way that lets us build new things really quickly.

Third, in David Weinberger's felicitous phrase, we can now start to have a Small Pieces Loosely Joined pattern. It's really worthwhile to look into what Joi Ito is doing with the Emergent Democracy movement, even if you're not interested in the themes of emerging democracy. This started because a conversation was going on, and Ito said "I am frustrated. I'm sitting here in Japan, and I know all of

these people are having these conversations in real-time with one another. I want to have a group conversation, too. I'll start a conference call.

"But since conference calls are so lousy on their own, I'm going to bring up a chat window at the same time." And then, in the first meeting, I think it was Pete Kaminski said "Well, I've also opened up a wiki, and here's the URL." And he posted it in the chat window. And people can start annotating things. People can start adding bookmarks; here are the lists.

So, suddenly you've got this meeting, which is going on in three separate modes at the same time, two in real-time and one annotated. So you can have the conference call going on, and you know how conference calls are. Either one or two people dominate it, or everyone's like "Oh, can I -- no, but --", everyone interrupting and cutting each other off.

It's very difficult to coordinate a conference call, because people can't see one another, which makes it hard to manage the interrupt logic. In Joi's conference call, the interrupt logic got moved to the chat room. People would type "Hand," and the moderator of the conference call will then type "You're speaking next," in the chat. So the conference call flowed incredibly smoothly.

Meanwhile, in the chat, people are annotating what people are saying. "Oh, that reminds me of So-and-so's work." Or "You should look at this URL...you should look at that ISBN number." In a conference call, to read out a URL, you have to spell it out -- "No, no, no, it's w w w dot net dash..." In a chat window, you get it and you can click on it right there. You can say, in the conference call or the chat: "Go over to the wiki and look at this."

This is a broadband conference call, but it isn't a giant thing. It's just three little pieces of software laid next to each other and held together with a little bit of social glue. This is an incredibly powerful pattern. It's different from: Let's take the Lotus juggernaut and add a web front-end.

And finally, and this is the thing that I think is the real freakout, is ubiquity. The web has been growing for a long, long time. And so some people had web access, and then lots of people had web access, and then most people had web access.

But something different is happening now. In many situations, all people have access to the network. And "all" is a different kind of amount than "most." "All" lets you start taking things for granted.

Now, the Internet isn't everywhere in the world. It isn't even everywhere in the developed world. But for some groups of people -- students, people in high-tech offices, knowledge workers -- everyone they work with is online. Everyone they're friends with is online. Everyone in their family is online.

And this pattern of ubiquity lets you start taking this for granted. Bill Joy once said "My method is to look at something that seems like a good idea and assume it's true." We're starting to see software that simply assumes that all offline groups will have an online component, no matter what.

It is now possible for every grouping, from a Girl Scout troop on up, to have an online component, and for it to be lightweight and easy to manage. And that's a different kind of thing than the old pattern of "online community." I have this image of two hula hoops, the old two-hula hoop world, where my real

life is over here, and my online life is over there, and there wasn't much overlap between them. If the hula hoops are swung together, and everyone who's offline is also online, at least from my point of view, that's a different kind of pattern.

There's a second kind of ubiquity, which is the kind we're enjoying here thanks to Wifi. If you assume whenever a group of people are gathered together, that they can be both face to face and online at the same time, you can start to do different kinds of things. I now don't run a meeting without either having a chat room or a wiki up and running. Three weeks ago I ran a meeting for the Library of Congress. We had a wiki, set up by Socialtext, to capture a large and very dense amount of technical information on long-term digital preservation.

The people who organized the meeting had never used a wiki before, and now the Library of Congress is talking as if they always had a wiki for their meetings, and are assuming it's going to be at the next meeting as well -- the wiki went from novel to normal in a couple of days.

It really quickly becomes an assumption that a group can do things like "Oh, I took my PowerPoint slides, I showed them, and then I dumped them into the wiki. So now you can get at them." It becomes a sort of shared repository for group memory. This is new. These kinds of ubiquity, both everyone is online, and everyone who's in a room can be online together at the same time, can lead to new patterns.

Part Three: What can we take for granted?

If these assumptions are right, one that a group is its own worst enemy, and two, we're seeing this explosion of social software, what should we do? Is there anything we can say with any certainty about building social software, at least for large and long-lived groups?

I think there is. A little over 10 years ago, I quit my day job, because Usenet was so interesting, I thought: This is really going to be big. And I actually wrote a book about net culture at the time: Usenet, the Well, Echo, IRC and so forth. It launched in April of '95, just as that world was being washed away by the web. But it was my original interest, so I've been looking at this problem in one way or another for 10 years, and I've been looking at it pretty hard for the a year and a half or so.

So there's this question "What is required to make a large, long-lived online group successful?" and I think I can now answer with some confidence: "It depends." I'm hoping to flesh that answer out a little bit in the next ten years.

But I can at least say some of the things it depends on. The Calvinists had a doctrine of natural grace and supernatural grace. Natural grace was "You have to do all the right things in the world to get to heaven..." and supernatural grace was "...and God has to anoint you." And you never knew if you had supernatural grace or not. This was their way of getting around the fact that the Book of Revelations put an upper limit on the number of people who were going to heaven.

Social software is like that. You can find the same piece of code running in many, many environments. And sometimes it works and sometimes it doesn't. So there is something supernatural about groups being a run-time experience.

The normal experience of social software is failure. If you go into Yahoo groups and you map out the subscriptions, it is, unsurprisingly, a power law. There's a small number of highly populated groups, a moderate number of moderately populated groups, and this long, flat tail of failure. And the failure is inevitably more than 50% of the total mailing lists in any category. So it's not like a cake recipe. There's nothing you can do to make it come out right every time.

There are, however, I think, about half a dozen things that are broadly true of all the groups I've looked at and all the online constitutions I've read for software that supports large and long-lived groups. And I'd break that list in half. I'd say, if you are going to create a piece of social software designed to support large groups, you have to accept three things, and design for four things.

Three Things to Accept

1.) Of the things you have to accept, the first is that you cannot completely separate technical and social issues. There are two attractive patterns. One says, we'll handle technology over `here, we'll do social issues there. We'll have separate mailing lists with separate discussion groups, or we'll have one track here and one track there. This doesn't work. It's never been stated more clearly than in the pair of documents called "LambdaMOO Takes a New Direction." I can do no better than to point you to those documents.

But recently we've had this experience where there was a social software discussion list, and someone said "I know, let's set up a second mailing list for technical issues." And no one moved from the first list, because no one could fork the conversation between social and technical issues, because the conversation can't be forked.

The other pattern that's very, very attractive -- anybody who looks at this stuff has the same epiphany, which is: "Omigod, this software is determining what people do!" And that is true, up to a point. But you cannot completely program social issues either. So you can't separate the two things, and you also can't specify all social issues in technology. The group is going to assert its rights somehow, and you're going to get this mix of social and technological effects.

So the group is real. It will exhibit emergent effects. It can't be ignored, and it can't be programmed, which means you have an ongoing issue. And the best pattern, or at least the pattern that's worked the most often, is to put into the hands of the group itself the responsibility for defining what value is, and defending that value, rather than trying to ascribe those things in the software upfront.

2.) The second thing you have to accept: Members are different than users. A pattern will arise in which there is some group of users that cares more than average about the integrity and success of the group as a whole. And that becomes your core group, Art Kleiner's phrase for "the group within the group that matters most."

The core group on Communitree was undifferentiated from the group of random users that came in. They were separate in their own minds, because they knew what they wanted to do, but they couldn't defend themselves against the other users. But in all successful online communities that I've looked at,

a core group arises that cares about and gardens effectively. Gardens the environment, to keep it growing, to keep it healthy.

Now, the software does not always allow the core group to express itself, which is why I say you have to accept this. Because if the software doesn't allow the core group to express itself, it will invent new ways of doing so.

On alt.folklore.urban , the discussion group about urban folklore on Usenet, there was a group of people who hung out there and got to be friends. And they came to care about the existence of AFU, to the point where, because Usenet made no distinction between members in good standing and drive-by users, they set up a mailing list called The Old Hats. The mailing list was for meta-discussion, discussion about AFU, so they could coordinate efforts formally if they were going to troll someone or flame someone or ignore someone, on the mailing list.

Addendum, July 2, 2003: A longtime a.f.u participant says that the Old Hat list was created to allow the Silicon Valley-dwelling members to plan a barbecue, so that they could add a face-to-face dimension to their virtual interaction. The use of the list as a backstage area for discussing the public newsgroup arose after the fact.

Then, as Usenet kept growing, many newcomers came along and seemed to like the environment, because it was well-run. In order to defend themselves from the scaling issues that come from of adding a lot of new members to the Old Hats list, they said "We're starting a second list, called the Young Hats."

So they created this three-tier system, not dissimilar to the tiers of anonymous cowards, logged-in users, and people with high karma on Slashdot. But because Usenet didn't let them do it in the software, they brought in other pieces of software, these mailing lists, that they needed to build the structure. So you don't get the program users, the members in good standing will find one another and be recognized to one another.

3.) The third thing you need to accept: The core group has rights that trump individual rights in some situations. This pulls against the libertarian view that's quite common on the network, and it absolutely pulls against the one person/one vote notion. But you can see examples of how bad an idea voting is when citizenship is the same as ability to log in.

In the early Nineties, a proposal went out to create a Usenet news group for discussing Tibetan culture, called soc.culture.tibet. And it was voted down, in large part because a number of Chinese students who had Internet access voted it down, on the logic that Tibet wasn't a country; it was a region of China. And in their view, since Tibet wasn't a country, there oughtn't be any place to discuss its culture, because that was oxymoronic.

Now, everyone could see that this was the wrong answer. The people who wanted a place to discuss Tibetan culture should have it. That was the core group. But because the one person/one vote model on Usenet said "Anyone who's on Usenet gets to vote on any group," sufficiently contentious groups could simply be voted away.

Imagine today if, in the United States, Internet users had to be polled before any anti-war group could be created. Or French users had to be polled before any pro-war group could be created. The people who want to have those discussions are the people who matter. And absolute citizenship, with the idea that if you can log in, you are a citizen, is a harmful pattern, because it is the tyranny of the majority.

So the core group needs ways to defend itself -- both in getting started and because of the effects I talked about earlier -- the core group needs to defend itself so that it can stay on its sophisticated goals and away from its basic instincts.

The Wikipedia has a similar system today, with a volunteer fire department, a group of people who care to an unusual degree about the success of the Wikipedia. And they have enough leverage, because of the way wikis work, they can always roll back graffiti and so forth, that that thing has stayed up despite repeated attacks. So leveraging the core group is a really powerful system.

Now, when I say these are three things you have to accept, I mean you *have to* accept them. Because if you don't accept them upfront, they'll happen to you anyway. And then you'll end up writing one of those documents that says "Oh, we launched this and we tried it, and then the users came along and did all these weird things. And now we're documenting it so future ages won't make this mistake." Even though you didn't read the thing that was written in 1978.

All groups of any integrity have a constitution. The constitution is always partly formal and partly informal. At the very least, the formal part is what's substantiated in code -- "the software works this way."

The informal part is the sense of "how we do it around here." And no matter how is substantiated in code or written in charter, whatever, there will always be an informal part as well. You can't separate the two.

Four Things to Design For

1.) If you were going to build a piece of social software to support large and long-lived groups, what would you design for? The first thing you would design for is handles the user can invest in.

Now, I say "handles," because I don't want to say "identity," because identity has suddenly become one of those ideas where, when you pull on the little thread you want, this big bag of stuff comes along with it. Identity is such a hot-button issue now, but for the lightweight stuff required for social software, its really just a handle that matters.

It's pretty widely understood that anonymity doesn't work well in group settings, because "who said what when" is the minimum requirement for having a conversation. What's less well understood is that weak pseudonymity doesn't work well, either. Because I need to associate who's saying something to me now with previous conversations.

The world's best reputation management system is right here, in the brain. And actually, it's right here, in the back, in the emotional part of the brain. Almost all the work being done on reputation systems today is either trivial or useless or both, because reputations aren't linearizable, and they're not portable.

There are people who cheat on their spouse but not at cards, and vice versa, and both and neither. Reputation is not necessarily portable from one situation to another, and it's not easily expressed.

eBay has done us all an enormous disservice, because eBay works in non-iterated atomic transactions, which are the opposite of social situations. eBay's reputation system works incredibly well, because it starts with a linearizable transaction -- "How much money for how many Smurfs?" -- and turns that into a metric that's equally linear.

That doesn't work well in social situations. If you want a good reputation system, just let me remember who you are. And if you do me a favor, I'll remember it. And I won't store it in the front of my brain, I'll store it here, in the back. I'll just get a good feeling next time I get email from you; I won't even remember why. And if you do me a disservice and I get email from you, my temples will start to throb, and I won't even remember why. If you give users a way of remembering one another, reputation will happen, and that requires nothing more than simple and somewhat persistent handles.

Users have to be able to identify themselves and there has to be a penalty for switching handles. The penalty for switching doesn't have to be total. But if I change my handle on the system, I have to lose some kind of reputation or some kind of context. This keeps the system functioning.

Now, this pulls against the sense that we've had since the early psychological writings about the Internet. "Oh, on the Internet we're all going to be changing identities and genders like we change our socks."

And you see things like the Kaycee Nicole story, where a woman in Kansas pretended to be a high school student, and then because the invented high school student's friends got so emotionally involved, she then tried to kill the Kaycee Nicole persona off. "Oh, she's got cancer and she's dying and it's all very tragic." And of course, everyone wanted to fly to meet her. So then she sort of panicked and vanished. And a bunch of places on the Internet, particularly the MetaFilter community, rose up to find out what was going on, and uncovered the hoax. It was sort of a distributed detective movement.

Now a number of people point to this and say "See, I told you about that identity thing!" But the Kaycee Nicole story is this: changing your identity is really weird. And when the community understands that you've been doing it and you're faking, that is seen as a huge and violent transgression. And they will expend an astonishing amount of energy to find you and punish you. So identity is much less slippery than the early literature would lead us to believe.

2.) Second, you have to design a way for there to be members in good standing. Have to design some way in which good works get recognized. The minimal way is, posts appear with identity. You can do more sophisticated things like having formal karma or "member since."

I'm on the fence about whether or not this is a design or accepting. Because in a way I think members in good standing will rise. But more and more of the systems I'm seeing launching these days are having some kind of additional accretion so you can tell how much involvement members have with the system.

There's an interesting pattern I'm seeing among the music-sharing group that operates between Tokyo and Hong Kong. They operate on a mailing list, which they set up for themselves. But when they're trading music, what they're doing is, they're FedExing one another 180-gig hard-drives. So you're getting .wav files and not MP3s, and you're getting them in bulk.

Now, you can imagine that such a system might be a target for organizations that would frown on this activity. So when you join that group, your user name is appended with the user name of the person who is your sponsor. You can't get in without your name being linked to someone else. You can see immediately the reputational effects going on there, just from linking two handles.

So in that system, you become a member in good standing when your sponsor link goes away and you're there on your own report. If, on the other hand, you defect, not only are you booted, but your sponsor is booted. There are lots and lots of lightweight ways to accept and work with the idea of member in good standing.

3.) Three, you need barriers to participation. This is one of the things that killed Usenet. You have to have some cost to either join or participate, if not at the lowest level, then at higher levels. There needs to be some kind of segmentation of capabilities.

Now, the segmentation can be total -- you're in or you're out, as with the music group I just listed. Or it can be partial -- anyone can read Slashdot, anonymous cowards can post, non-anonymous cowards can post with a higher rating. But to moderate, you really have to have been around for a while.

It has to be hard to do at least some things on the system for some users, or the core group will not have the tools that they need to defend themselves.

Now, this pulls against the cardinal virtue of ease of use. But ease of use is wrong. Ease of use is the wrong way to look at the situation, because you've got the Necker cube flipped in the wrong direction. The user of social software is the group, not the individual.

I think we've all been to meetings where everyone had a really good time, we're all talking to one another and telling jokes and laughing, and it was a great meeting, except we got nothing done. Everyone was amusing themselves so much that the group's goal was defeated by the individual interventions.

The user of social software is the group, and ease of use should be for the group. If the ease of use is only calculated from the user's point of view, it will be difficult to defend the group from the "group is its own worst enemy" style attacks from within.

4.) And, finally, you have to find a way to spare the group from scale. Scale alone kills conversations, because conversations require dense two-way conversations. In conversational contexts, Metcalfe's law is a drag. The fact that the amount of two-way connections you have to support goes up with the square of the users means that the density of conversation falls off very fast as the system scales even a little bit. You have to have some way to let users hang onto the less is more pattern, in order to keep associated with one another.

This is an inverse value to scale question. Think about your Rolodex. A thousand contacts, maybe 150 people you can call friends, 30 people you can call close friends, two or three people you'd donate a kidney to. The value is inverse to the size of the group. And you have to find some way to protect the group within the context of those effects.

Sometimes you can do soft forking. Live Journal does the best soft forking of any software I've ever seen, where the concepts of "you" and "your group" are pretty much intertwingled. The average size of a Live Journal group is about a dozen people. And the median size is around five.

But each user is a little bit connected to other such clusters, through their friends, and so while the clusters are real, they're not completely bounded -- there's a soft overlap which means that though most users participate in small groups, most of the half-million LiveJournal users are connected to one another through some short chain.

IRC channels and mailing lists are self-moderating with scale, because as the signal to noise ratio gets worse, people start to drop off, until it gets better, so people join, and so it gets worse. You get these sort of oscillating patterns. But it's self-correcting.

And then my favorite pattern is from MetaFilter, which is: When we start seeing effects of scale, we shut off the new user page. "Someone mentions us in the press and how great we are? Bye!" That's a way of raising the bar, that's creating a threshold of participation. And anyone who bookmarks that page and says "You know, I really want to be in there; maybe I'll go back later," that's the kind of user MeFi wants to have.

You have to find some way to protect your own users from scale. This doesn't mean the scale of the whole system can't grow. But you can't try to make the system large by taking individual conversations and blowing them up like a balloon; human interaction, many to many interaction, doesn't blow up like a balloon. It either dissipates, or turns into broadcast, or collapses. So plan for dealing with scale in advance, because it's going to happen anyway.

Conclusion

Now, those four things are of course necessary but not sufficient conditions. I propose them more as a platform for building the interesting differences off. There are lots and lots and lots of other effects that make different bits of software interesting enough that you would want to keep more than one kind of pattern around. But those are commonalities I'm seeing across a range of social software for large and long-lived groups.

In addition, you can do all sorts of things with explicit clustering, whether it's guilds in massively multi-player games, or communities on Live Journal or what have you. You can do things with conversational artifacts, where the group participation leaves behind some record. The Wikipedia right now, the group collaborated online encyclopedia is the most interesting conversational artifact I know of, where product is a result of process. Rather than "We're specifically going to get together and create this presentation" it's just "What's left is a record of what we said."

There are all these things, and of course they differ platform to platform. But there is this, I believe, common core of things that will happen whether you plan for them or not, and things you should plan for, that I think are invariant across large communal software.

Writing social software is hard. And, as I said, the act of writing social software is more like the work of an economist or a political scientist. And the act of hosting social software, the relationship of someone who hosts it is more like a relationship of landlords to tenants than owners to boxes in a warehouse.

The people using your software, even if you own it and pay for it, have rights and will behave as if they have rights. And if you abrogate those rights, you'll hear about it very quickly.

That's part of the problem that the John Hegel theory of community -- community leads to content, which leads to commerce -- never worked. Because lo and behold, no matter who came onto the Clairol chat boards, they sometimes wanted to talk about things that weren't Clairol products.

"But we paid for this! This is the Clairol site!" Doesn't matter. The users are there for one another. They may be there on hardware and software paid for by you, but the users are there for one another.

The patterns here, I am suggesting, both the things to accept and the things to design for, are givens. Assume these as a kind of social platform, and then you can start going out and building on top of that the interesting stuff that I think is going to be the real result of this period of experimentation with social software.

Thank you very much.