Behaviour & Information Technology, Vol. 24, No. 5, September 2005, 353–363

Taylor & Francis
Taylor & Francis Group

Polite Computing

BRIAN WHITWORTH*

New Jersey Institute of Technology, New Jersey

This paper presents politeness as a key social requirement for computer human interaction (CHI). Politeness is defined, in information terms, as offering the locus of control of a social interaction to another party. Software that creates pop-up windows is not illegal, but is impolite, as it preempts user choice. It is proposed that impolite software drives users away, while polite software attracts them. Specifying politeness suggests four requirements: (1) Respect user choice (2) Disclose yourself (3) Offer useful choices (4) Remember past choices. Software that ignores these rules may fail not by logic error but by social error. ''Mr. Clippy'' is an example of software that users often disable because it is impolite. Operating systems could support application politeness by providing an application source registry and a general meta-choice console. A future is envisaged where software politeness is a critical software success requirement.

1. Introduction

As my Windows computer boots, the taskbar fills with little icons. Each application that loads into memory increases my startup time. Many of them I never use and don't want, but each new application installation seems to add another. Some, like the virus checker, must be there, but others I could call up as needed. They need not always consume memory. Windows programs commandeer the start up process in so many ways that removing them can be difficult. Some have a ''Do not load at startup'' option, but others are only removed by a registry change (or ''hack''). Still others, like AOL and Windows messenger, rise from the dead after being removed. If you upgrade windows you find them back again, not just on the desktop but also taskbar, startup menu, hard drive and registry.

If software that took over your computer for its own purposes were a person, it would be called selfish. Selfish people act in their own interests, treating others as pawns on their checkerboard of life. If one can have selfish genes (Dawkins, 1989), one can talk of *selfish software,* which runs itself at every opportunity. Its ideal is to load at startup, then run continuously.

In human society, such selfishness seems the anti-social fly in the ointment of social cooperation (Ridley, 1996). While we are all selfish, if we were completely so, game theory suggests the win-win synergies of society would reduce to lose-lose equilibria (Poundstone, 1992). The ''non-zero-sum'' benefits which modern society creates suggest there is more to humanity than selfishness (Wright, 2001). Our ability to be ''social'' can resolve situations selfish economics cannot, like the tragedy of the commons (Hardin, 1968). While sociability at first seems less adaptive than selfishness, our species may have succeeded as much from sociability as intelligence, as cooperative society is extremely productive.

The benefit of social values may extend to online interaction (Friedman, Howe, & Felten, 2002). If politeness is part of sociability, and sociability is productive, polite software may be more productive. Polite software would also make online society a nicer place to be. This paper aims to define, specify and illustrate a vision of polite computing.

2. What is politeness?

To apply politeness to software, it must be defined in information terms. Webster sees politeness as ''Exhibiting in manner or speech a considerate regard for others.''

*Corresponding author. Email: bwhitworth@acm.org

Behaviour & Information Technology
ISSN 0144-929X print/ISSN 1362-3001 online © 2005 Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/01449290512331333700

(Marckwardt, Cassidy, & McMillan, 1992). It is a fundamental social concept, yet actions polite in one culture can be rude in another. This does not make politeness arbitrary. The common form of politeness may be its goal, not its specific behaviors, as was found for the IS task concept (Zigurs, Buckland, Connolly, & Wilson, 1999).

It is useful to distinguish polite behaviors (whether prescribed or proscribed), from the general intent of thoughtful consideration (Miller, 2004). Miller calls both etiquette, but let us call the specific behaviors etiquette, and the general goal politeness. Then it can be argued that while *etiquette* varies between cultures, *politeness* is a common theme.

What is that theme? Reeves and Nass see politeness as simply being more agreeable to others when in social contact than when not (Reeves & Nass, 1996). For them, agreeing with another's self praise is one of the ''most fundamental rules of politeness'' (Nass, 2004, p36). They argue that when another says ''I think I'm a good teacher; what do you think?'', polite people must respond ''You're great'', even if they don't think so (Nass, 2004). Is politeness just fake niceness?

Being agreeable may attend politeness, but does not define it if one can be polite but not agreeable, or agreeable but not polite. Giving money to the poor is kind, and agreeable to them, but philanthropists may not be polite, i.e. kindness is not equal to politeness. Conversely, one can politely refuse, beg to differ, respectfully object and humbly criticize, i.e. disagree, but still be polite. These examples suggest that agreeableness is not the source concept behind politeness.

Suppose that somehow people tended to be socially considerate in their interactions, to be pleasant and cooperative and avoid conflict. The ensuing social productivity would reward this tendency. What is ''considerate'' would vary between societies, giving no common etiquette, but a general form could arise *if the person considered knows what is considerate for them*. Then it would always be considerate to give the other choices. This approach does not presume to know what they want, e.g. if I meet a mother with a baby at a door, hold the door open and say ''After you'', I am offering the mother choice, and being polite. Yet, at that moment the baby may need attention. The mother may say ''No thank you'', and attend the baby not the door. Without choice, my ''kind'' act could be a burden. Indeed any ''good'' act becomes bad if forced, including helping little old ladies cross the street.

When equal parties interact to jointly create common events, control is normally shared, e.g. in a conversation both parties take turns speaking, and *the interaction locus of control* passes back and forth between the parties. In this situation, giving another the control is considerate, and so polite.

Many examples of politeness follow this general form, of giving choice. Interrupting some one is impolite, as it takes away their choice to speak. Letting another to finish before talking is polite, as they can choose when to stop. Asking permission to do something is polite, because it gives the other the choice to say no. The polite words ''Please'' and ''Thank you'' imply choice. Please implies the other had a choice to say no, as one does not ask ''Please'' if the other has no choice over the matter. Conversely, ''Thank you'' implies the other need not have done what they did, i.e. that they had choice. One need not thank another for a forced action, e.g. one would not forcibly eject someone from a building then say ''Thank you for leaving'' (except sarcastically). When people acting under orders are thanked, like soldiers or firemen, it is the voluntary aspect of their service that is recognized.

If the giving/taking of choice is central to the polite/impolite distinction, politeness is more than kindness. To make an unwilling child learn the piano or play sport may benefit them in the long run, but an imposed good is not polite, as it denies another's choice.

However being offensive seems always impolite. Politeness has two aspects: to offer another choice, and not to disable their choice. When I politely say ''After you'' before a doorway, it is assumed I am not blocking the entrance. This duality corresponds to the linguistic concept of politeness as maintaining positive and negative ''face'' (Brown & Levinson, 1987). Creating positive face is the good image created when one gives choice, and avoiding negative face is not taking another's rightful choices (or having one's own taken). For linguists, politeness is managing ''face'', which corresponds to both giving and taking rightful choice, and not wrongfully taking or giving the same.

Politeness presumes an absence of that which denies choice, so is not just what one does but also what one does not do. In social interaction, rude or offensive acts may force a negative state upon another, so part of ''considerate regard'' is to not upset the other by offensive acts. That does not make politeness solely the avoidance of rudeness, i.e. being always agreeable. Hating someone means you don't love them, but not hating them does not mean you love them. Likewise politeness is more than not offending, it is the giving of choice.

If politeness is giving choice, this implies the giver has the choice to give. Can one offer another choice when one has no choice oneself? Hence, we attribute politeness to people, not machines. Politeness, by this definition, must be voluntary.

Equally, giving choice is only polite if the receiver wants the choice, e.g. letting another go first is only polite if they want to go first. ''After you'' is polite when jumping into a sinking ship's lifeboat, but not when someone must soothe a crying baby at night. Excessive choice can be

burdensome, which the issue of meta-choices later addresses.

Giving choice means the other may return it. If a given choice is required, it is coercion not politeness. It can be polite to talk in a conversation where the other would rather listen. Polite people monitor what the other wants. That the locus of control is with another means they can speak any time they wish. Giving the locus of control to another does not necessarily mean forcing the locus (and burden) of action upon them.

Finally, is the "consideration" of illegal and politeness? If opening a door for one who wants to enter is polite, is handing a gun to a serial killer who wants to kill someone also polite? If someone would punch you in the nose, is it polite to acquiesce? Most agree that politeness does not enjoin us to enable choices society considers illegitimate or illegal.

The above considerations can be summarized in a single definition:

> *Politeness is any unrequired support for situating the locus of choice control of a social interaction with another party to it, given that control is desired, rightful and optional.*

The implications of this informational definition of politeness will now be considered.

3. Politeness and the law

Politeness and laws have different social roles. Laws formally specify the minimum a citizen should do for social order, but not the maximum they could do. To give what the law requires is not politeness, precisely because it is required. One does not thank a driver who must stop at a red light, but one does thank the driver who lets you into a line of traffic when they don't have to. Hence politeness is offering more choice than the law requires.

Polite interaction, legitimate interaction and anti-social interaction can be ordered by the degree of choice offered (Figure 1). Anti-social acts, like theft, murder or rape, offer the other party the least choice. Legitimate interaction offers a fair sharing of choice, while polite interaction gives the most choice to the other party. Politeness seems to begin where fixed laws end.

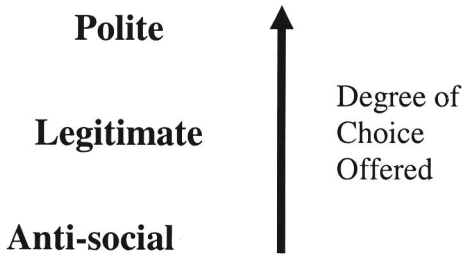

Figure 1. Choice by interaction type.

The value of politeness is flexibility. It covers the gray areas of social interaction, relieving the law from masses of detail. Life has so many possibilities that no laws can cover them all. Without politeness, one could wait endlessly to enter a busy traffic stream. This would require a law that after a certain wait time, entering traffic would have right of way. How could such a law be implemented? What people actually do is wait a while, then move forward until someone is polite enough to let them in. Laws cannot cover every case. Without politeness, rules could multiply like weeds, creating a legal system that is expensive for society to run. Saying sorry costs much less than a lawsuit. Politeness reduces legal costs by leaving situations to the social goodwill of the parties concerned.

Another advantage of politeness is that it is scalable, and reciprocated socially not personally. If someone in the street asks you for directions, it is unlikely the same person will one day give you directions. Yet it is likely that one day someone else in society will help you out. Hence, politeness scales to large societies, where most people don't know each other personally.

If criminal activity destroys the fabric of society, and legality maintains it, then politeness creates it by engendering goodwill. Further, politeness remains effective as social group size increases. It is well suited for a global Internet society. If software were more polite, people might be more willing to use it, and less willing to abuse it. Online society would improve if designed with politeness in mind.

3.1 *A research question*

The above discussion applies to interactions between people not machines. Yet it could apply to CHI, as people can interact with a computer as if it were a person, even when they know it is not (Reeves & Nass, 1996). Miller points out that if I accidentally hit my thumb with a hammer, I blame myself not the hammer, yet people may blame an equally mechanical computer for errors they initiate (Miller, 2004). Software it seems, with its ability to make choices, has crossed the threshold from inert to active. It has achieved "agent" status, and we react to agents on a social level.

This may not be as foolish as it seems. While the computer is a machine, people create it's programs. An installation program acts as directed by the company that designed it. If one party is directed entirely by a third party, their actions represent that party. They are an agent for them. If the third party is a social entity, then the interaction is social, even if agent is not. If software electronically mediates human-human interaction, while

the immediate interaction is computer-human, it makes sense to treat an agent for a social source in social terms. Studies show that people don't treat computers as people outside the mediation context (Goldstein, Alsio, & Werdenhoff, 2002), and show significantly more relational behaviors when the other party is definitely human (Shectman & Horowitz, 2003).

The politeness motive could be a general desire to exercise choice (Langer, 1975). If people have a natural desire to control their environment, having control feels good, and having no control will frustrate, giving anxiety or anger. If choice is generally desired, polite software should be preferred to impolite software. Politeness can be measured as the number of desired user choices given by the software minus the number that are taken away, e.g. if the choice to add desktop items is user desired, software that adds icons without asking would be, in that choice, impolite.

Users should more often remove or ignore applications that are impolite and reduce their choice. As users become more adept at exercising choice, the effect should be more pronounced. If computer human interaction is social, the logic of politeness becomes an online research question:

> *In a computer-human interaction involving valued physical, informational or cognitive resources, will polite software be used and accepted more, and deleted and disabled less, than impolite software?*

4. Impolite computing

If polite actions offer choice and impolite actions deny it, how polite is current CHI? Suppose one is browsing the Internet and a pop-up window suddenly appears. You were looking at one thing, then found yourself forced to look at another. Your cursor, or point of focus, was "hijacked". Your choice to look at what you want to look at was taken away. This is not illegal, but it is impolite. Users don't like their screen real estate being commandeered by pop-up windows (or pop-under windows that must later be closed).

Apologists suggest pop-up ads are the computer equivalent of TV commercials, but TV ads leave the viewer channel under viewer control. Pop-up ads initiate a new communication channel (a new window), and bypass user control in doing so. TV viewers expect commercials, but pop-up ads are unexpected. They are the equivalent of a TV commercial changing your channel for you. Browsers that repress pop-up ads are popular because pop-up ads are impolite.

Spam is also impolite, because it fills inboxes with messages users do not want – it is email without receiver choice (Whitworth & Whitworth, 2004). Telemarketing is the telephone equivalent of junk email. Over 50 million people signed up for the U.S. "Do Not Call" list, seeking some choice from telemarketers who always call, wanted or not. People's reaction to spam and telemarketing is the same - they want choice in communication. The common theme of spam, telemarketing and pop-up windows is impoliteness. People try to fight back. If they cannot, they reduce their interaction, whether it is email, telephone or browsing.

Getting "informed consent" before recording a person's activity is polite. It gives the person the choice to go "on the record" or not. Responsible magazines ask before printing a non-celebrity photo, and journals get permission to print an author's words. Yet only when Comet System's 1998 secret generation of "click-stream" data created a media storm did it disclose itself. Only when users objected to Intel's inclusion of a traceable Processor Serial Number (PSN) in its Pentium III in 1999 was the "feature" disabled. Only when Microsoft's Windows98 secret registration recording of user hardware and software details became public did they agree to stop. There remains concern that Microsoft's Media Player, bundled with Windows XP, quietly records the DVDs it plays and sends the data back to Microsoft (Editor, 2002). The privacy record of companies in cyberspace is at best weak (Privacy-International, 2002). If applications were polite, they would ask before they did such things.

That people will reveal personal information online is not the issue. They do. The issue is whether they should have the choice to do so or not. Software that secretly adds, deletes, changes, views, copies or distributes user data without permission is impolite as it denies rightful user choices. If the software norm is to take information first, then ask later, software usage, including e-commerce, will be less than it should be.

Installation programs are notorious for acting without asking, e.g. the Real-One Player adds a variety of desktop icons and browser links, installs itself in the system tray, and if the user is not careful, commandeers all video and sound file associations. It does this because it can, but customers resent the invasion. Software upgrades continue the impolite tradition. When Internet Explorer is upgraded, it makes your browser home page MSN, and adds to your Links bar without asking. What gives software the right to change my home page? An installation program that assumes it can change what it wants, is like furniture deliverers assuming they can rearrange your house because they happen to be in it.

The line between illegal and impolite is blurred, as what is impolite today may be illegal tomorrow, e.g. programs that dial long distance pornography on your phone bill, change your browser or email preferences, place cookies on your hard-drive to secretly record what you do, and record your mouse clicks as you surf the web. Whether spyware or stealthware, the common problem is taking user rightful choices, i.e. impoliteness. The common consequence is a reduced willingness to interact online.

5. The polite benefit

Some see politeness as a corporate cost, but the opposite may be true. Many successful online businesses are polite, and support rather than deny user choice. The rationale is simple: *customers given choices come back*. Politeness includes companies not pre-deciding what customers want, being visible, helping users choose, and remembering past choices, e.g. Amazon gives customers information on what books similar buyers buy. This information is not imposed by pop-up ads or flashing headers, but given as a view option below. Rather than a noisy demand to buy, it is a polite reminder of same-time purchases that could save the customer postage.

Ebay's customer reputation feedback is polite, and helps both business and customer. Reputation information, while about individuals, is created by the group as a whole. It is not a personal product, so one has no privacy rights to one's reputation. If reputation information is given as an average, the privacy of the individuals creating it is secure. Aggregate group data is one of the three levels of meaning people seek in social interaction (along with factual and personal data) (Whitworth, Gallupe, & McQueen, 2000), and group information can strongly influence online decision making (Whitworth, Gallupe, & McQueen, 2001). Giving optional access to valued choice relevant information is by the previous definition, polite.

Google also illustrates that politeness pays. Google ads do not scream from flashing headers but sit quietly at screen right. They are options not demands, so users do not mind them. Google e-mail (GMail) is free, but scans email and shows ads relevant to the message content. This seems to contradict privacy. Surely people will object to their email being scanned? Not if it is done politely. Privacy does not mean people want to be socially insulated. It is about the user choice to reveal personal data or not, not about keeping it secret (Nissenbaum, 1997). We are by nature social, so people are acceptant of video surveillance when it is disclosed, they know how the surveillance data is used, and there is a public good reason (like security) (George, 1996). If Google gives customers opt-in choice, fully discloses its data use, and does not force ads upon them, its offer of free email will likely succeed.

Amazon, Google and E-Bay have succeeded by giving customers what they want in a polite way. Customers want to have choice. Companies that support this get return custom. By giving rather than taking customer choice, polite companies win business. In trade, politeness means never having to say you are sorry. Impolite actions presume a seller-customer battle, which as with any war, reduces mutual benefits. Politeness in contrast, presumes a willing customer who wants to purchase. In this synergistic business model, the customer is not the enemy, but a partner in mutual gain. It is an excellent way to manage the customer relationship.

6. Specifying polite computing

The widespread problem of impolite software may reflect a general software design "blind spot". There seem currently no guidelines for designing polite software beyond common sense. Specifying politeness in information terms addresses Millers core etiquette question: "If the software were replaced by a polite human assistant, how would that assistant behave?" (Miller, 2004).

Software politeness can be modeled on human politeness. When two people simultaneously approach a physical door, they interact over a common resource. Polite people avoid a physical collision. They stop and say "Excuse me". Rather than rushing to be first, they stop and withdraw gracefully. Both are visible, so each sees the other has stopped, and knows to whom their interaction is directed. Then one party politely signals the other to go first, perhaps saying "After you", which offer may be returned ("No, after you"). Politeness means the interaction locus of control is offered, not taken. When an offer is accepted, the giver remembers and supports that choice. By offering choice (politeness), the situation is resolved without conflict.

A similar analysis applies to other polite examples, like a waiter in a restaurant. The waiter seats you, but respects your choice to be seated elsewhere. The waiter introduces him or herself, so you know to whom you are talking. The waiter offers menu choices in a helpful way, and explains choices you do not understand. Finally, a good waiter remembers your choices next time you come (Cooper, 1999).

Person-to-person politeness suggests the following CHI politeness rules:

1. *Respect* user choice.
2. *Disclose* yourself.
3. *Offer* useful choices.
4. *Remember* past choices.

6.1 *Respect user choice*

In social interaction choices may overlap, e.g. one party's choice may prevent another's. In the doorway example, if one person enters, the other is blocked. If politeness is giving choice, its first requirement is not to deny another's choice, and preemptive acts do just that. A program that changes a user's browser home page without asking preempts a user choice.

Polite computing means programs don't try to dominate the computer-human interaction. Polite software should not initiate actions on common resources without

permission, including the desktop, registry, hard drive, task bar, file associations, quick launch or other settings. To act unilaterally on a mutual resource is impolite. If people own the computers they purchase, applications should only act with user consent. Inexperienced users may give software permission to do whatever it thinks is best, as they don't know what is best. Experienced users may customize every option, because they know what they want. In both cases, polite software asks before it acts on or in another's domain.

> *Rule 1: Polite software respects, and does not preempt, rightful user choices.*

6.2 *Disclose yourself*

In online social interaction, that parties see and are seen seems important (Erickson & Kellog, 2000). To act secretly behind another's back, to sneak or hide for any reason, is considered impolite. Why is it polite to introduce yourself? If politeness increases another's interaction choices, then non self-disclosure reduces them. Each party is part of the interaction. If one party is hidden, part of the interaction is also hidden. A hidden party is untouchable, and thus unaccountable for their actions. Hence, when individuals interact, they want to know who the other party is. Disclosing oneself is part of being polite, while secrecy can imply malicious intent. When polite people introduce themselves and state their business, they become accountable for future responses, including retribution.

If polite people introduce themselves, polite software should do the same. Users should see not only what is happening on their computer, but also the source. In Windows, the Task Manager shows active applications/processes, but *who* is doing what is unclear. The social source is hidden behind cryptic component names like svchhost.exe. The cryptic names problem runs throughout Windows, as applications can give their components any names they like. If polite parties are visible, then the software actors on the computer stage are often impolite.

> *Rule 2: Polite software declares itself and its social source.*

6.3 *Offer useful choices*

The third property of politeness is to offer useful choices. To be useful, a choice must be understood as well as desired. To confuse users with complex options, or give insufficient information, is impolite and inconsiderate. A choice misrepresented, or misunderstood, may not be a choice at all. One can argue we are too ignorant to understand the complexities of a software install, but the same argument could apply to installing satellite TV. Most users do not understand satellite technology, but still expect installers to explain the options in a way they can understand. If not, they may choose to not interact at all, and find another installer. Likewise, software that does not support user choice may not be used at all. People want choice, and if not offered it, may go elsewhere.

Most users prefer simple to complex choices. A complex act like an installation can be made simpler by analyzing choice dependencies, how one choice affects another, e.g. letting a user install an application without an essential component is not a choice but a trap. Application-critical components should be part of an all or none choice. Analyzing choice dependencies can reduce many choices to one – install or not. It can separate optional from essential choices, so permission to install may imply hard drive, registry and start menu access, but not desktop, system tray, favorites or file associations access.

There is one situation where computers seem always willing to give users choice - when things go wrong. Yet these error message choice offerings are often like the classic DOS "Abort, Retry, or Fail?" Few users knew the difference between abort and fail. Usability has progressed since then, but considerate error messages are still uncommon. Programmers naturally report errors from a programmer perspective, giving data like ports, IP addresses, variables, records and sectors. To be polite, they must think in terms of user tasks and user choices.

> *Rule 3: Polite software helps users make desired choices.*

6.4 *Remember past choices*

Finally, it is not enough to give choices now but forget them later. If previous responses are forgotten, the user must redo them, which is inconsiderate. When polite people meet again they remember the last time. They do not ask, "What is your name?" at every meeting. It is polite to remember, yet every time I open Explorer it goes to its preferred directory, forcing me to wait, as it finds and fills it's list panel with innumerable files I don't want to see. Eventually the cursor returns to me, to select my directory of choice (which is *never* the one displayed). Each time I interact with Explorer, it acts as if it were the first time I had used it, although I am the only person it has ever known. Why does it force its default directory on me every time it starts up? Why can it not remember where I was last time, and return me there? The answer seems to be that currently programmers attach little value to politeness. Such "amnesia" is a trademark of impolite software.

When choices repeat, software should remember the last user choice as its default. The default reply to a user permission request should be the status quo (the last user choice). To make the default anything other than the status quo is to "forget" the past user choice, and is impolite.

If a choice repeats continually, to ask the same question repeatedly for the same reply is to pester, like the ''Are we there yet?'' of children on a car trip. The other party must reply again and again with the same answer, e.g. uploading a batch of files can create a series of overwrite questions. If the software continually asks ''Overwrite Y/N?'', I must hover by the upload to periodically press ''Yes'', lest the process stop. Polite software offers a general ''Yes to All'' option, i.e. it can remember from one upload to the next. The ''Do you want to connect?'' request in Windows is the same problem for a ''No'' choice. If you are reviewing email offline (perhaps someone else is using the line), actions like using Explorer trigger an Internet connection request every few minutes. No matter how many times you say ''No, don't connect'', it keeps asking. It has no interaction memory, which is impolite.

In general, software that takes in user choices in one operation should remember them the next time.

Rule 4: Polite software remembers past user choices.

6.5 *Summary*

Polite software allocates user resources with permission, discloses itself and its source, enables easy and simple choices and can remember past interactions. Conversely, impolite software acts preemptively, hides itself, confuses users and forgets past choices.

Some informal pointers are:

1. Don't preempt user choices.
2. Don't dominate the interaction.
3. Make it easy for the user to have control.
4. Don't interrupt the user, unless necessary.
5. Be brief.
6. Help users make desired choices.
7. Ask before acting on interface resources.
8. If in doubt ask; if you ask, remember.
9. Keep the user informed of your actions.
10. Don't pester.
11. Offer choices relevant to the user context.
12. Offer useful choices
13. Say excuse me, thank you and please.

How important politeness is to users can be determined by research. Impoliteness may constitute a new IS error type – *social error*. A logic error can cause an information flow failure that in turn causes a system failure or ''crash''. A social ''error'', like impoliteness, can cause a social failure, which is that the user ends the interaction. The software still works, but if people dislike it and don't use it, this can also be considered a system failure. The end effect of compiler and user rejection are the same - the application does not run. If users ignore, disable or delete software they find impolite, social failure becomes IS failure, especially on the web. For example, my new modem software loaded itself on startup, and regularly went online to check for updates, usually to find none. When its ''Searching for upgrades'' message would suddenly appear, I wondered, as most users would, ''Why are you doing this? Did I ask you to?'' The software was impolite so I removed it. Another example of social failure is ''Mr. Clippy''.

7. Mr. Clippy must die?

Mr. Clippy was Office '97's assistant, a paper clip figure who guessed you were having a problem and stepped in to lend a hand. Using advanced Bayesian logic, he was touted as the future of smart help. His most famous line: ''*It looks like your writing a letter...*'' came when you typed ''Dear ...''. Despite 25,000 hours of user testing (Horvitz, 2004), Mr. Clippy was so notable a failure that his *removal* was a Windows XP sales pitch (Levitt, 2001).

Searching the Internet for ''Mr. Clippy'' still gives comments like ''Die, Clippy, Die!'' and ''How do you turn that moronic paper clip off?'' According to a 2001 PC Magazine survey, Mr. Clippy was the third biggest software flop of the year, with same concept Microsoft Bob as the first (PCMagazine, 2001). One user wrote (Pratley, 2004):

> *''I HATED that clip. It hung around watching you with that nasty smirk. It wouldn't go away when you wanted it to. It interrupted rudely and broke your train of thought. It never actually had an answer to questions I had.''*

yet some liked the little guy:

> *''My favorite memory of the Office Assistant was when I was working in my home office, and my 2 year old niece walked into the room. When she saw Rocky [a Clippy variant], her eyes lit up. I could tell she was excited when she said, ''What's that?''''*

Chris Pratley, a Microsoft designer, in his Mr. Clippy weblog, asks ''If you think the Assistant idea was bad, why exactly?'' (Pratley, 2004). If Microsoft and other designers have not yet figured it out – *Mr. Clippy was impolite*.

Consider the four rules of politeness given earlier. Mr. Clippy was friendly and smart, but interrupted the user. You thought you were typing something? Sorry, you are now talking to Mr. Clippy, who will discuss some cool but irrelevant Word feature. Mr. Clippy broke the first rule of politeness by preemptively taking control of the cursor. Well, didn't he then offer choices? Yes, but this is no more being polite than kidnapping someone, then giving them a cup of tea, is being kind. One cannot take user choice in the name of giving it. The cursor was a common resource

between user and software, like the door in the example given earlier. Its preemptive use should not have been a software design option.

By opening a modal window and forcing a dialogue, Mr. Clippy chose the impolite way to offer help. The polite way is to present an optional side block, as Amazon and Google do. One is an offering to help (giving choice) the other is demanding to help (taking choice). Hence in XP, Mr Clippy was replaced by polite smart tags and task panes.

Mr. Clippy was a selfish application, so turning him off was not easy. Close his window and he would come back next time, as happy as before. He ignored continuous rejection. Choose the ''Hide'' option on the paper clip and Mr. Clippy seemed to run away, but when Word started again, he was right back in your world. He was like a guest who would not leave, no matter how many times you asked. To remove him you had to select options from the tool menu and deselect the service. Any other rejection was ignored. If Mr. Clippy had been polite, his control would have been obvious on the clip itself.

Yet Mr. Clippy did try, and many liked him. While smart help may not yet help experts, it may help novices or the young, who may trade politeness for usefulness. The control motive proposed to underlie politeness may vary with expertise. Novices or children may not expect control, and so not expect politeness. They are not usually in charge themselves, so may not mind Mr. Clippy taking charge. Given his usefulness to some, Mr. Clippy could have been tolerable ... if only he had listened.

Mr. Clippy ignored user disinterest, non-use and repeated help request denial. His designers seemed to assume they knew best (while politeness assumes the user knows their needs best). Sometimes users do not know what they want, so to interrupt once is tolerable. However, to interrupt repeatedly when repeatedly rejected is impolite. Mr. Clippy could watch *your* document actions, but had no memory whatsoever of *his* interaction with you. No matter how often you ignored or rejected his help, Mr. Clippy kept on coming. People build relationships by remembering past interactions, and expect ''intelligent'' software to do the same. Smart software should be smart enough to recognize rejection. The ability to remember past interactions could have turned Mr. Clippy's failure into success. All he had to do was know when he wasn't wanted.

By the usability theories of the day, Mr. Clippy should have been a winner. He had multi-media animation. He could be a cute little dog or a clever professor. He could express emotions like interest or boredom. He had Bayesian logic. However, none of this made him any the less rude. Mr. Clippy was a know-it-all who took over and did not listen. If he did know it all, it might have worked, but he did not. The failure was not of friendliness, cleverness or usefulness, but of politeness. The designers of Mr. Clippy forgot to make him polite.

8. Polite operating systems

If politeness is a desired property of online interaction, it needs operating system support. Why should applications attend politeness if operating systems do not? Some ways this could happen are now suggested, but as cultures differ in etiquette, so software could support politeness in many ways. Only the goal is common – valid user choices.

8.1 *A property rights approach*

If politeness is a social concept, we can apply a social property rights framework to information systems (Rose, 2001), moving the physical concept of ownership to information worlds (Whitworth & deMoor, 2003). Ownership is a complex social concept, yet most people find it familiar and understandable. Similarly, space perception is also complex, but users find it familiar, so it is useful in GUI interactions. The term ''screen real estate'' suggests users easily think of information in property terms. All computer resources (CPU, memory, hard drive, registry, ports etc) can be considered ''real estate''. Interface areas, like the desktop, start menu, taskbar, quick launch and favorites, can be considered the same way. The concept of screen real estate can be extended to all CHI resources.

The question then raised is, how should this real estate be governed? If the parties are the user and the software, options include autocracy, anarchy or polite power sharing.

Currently Windows applications *may* ask user permission to change system resources, but *need not* do so. In the latter case, their actions are invisible except by their effect. People can see where a T.V. installer is going, but have no idea where a computer installation program is going, or what it is doing. If programs can do what they want, who owns the computer? If one pays for a computer hard drive, screen, memory and ports, should one not own it? If I own my ''personal computer'' (PC), should I not have choice over what happens within it?

Like citizens in a society, most computer applications are useful, some are idle, but a few are thieves who abuse common resources. When social parties act by what they can do, not what they should do, society calls it ''disorder''. On the PC estate, disorder arises when application ''subjects'' can act at will. If currently running applications can and do change any system resource, then software ''might'' has prevailed over user ''right''. The degree of disorder depends on whether the operating system ''Sheriff'' is maintaining order or sleeping on the job. He is not powerless, as his private estate (the operating system kernel) is protected from application intrusion. If the operating system can shield and protect itself from applications, can it not do the same for users? The operating system could encourage applications to ask before they act, i.e. support politeness.

Why bother with politeness? Politeness is an antidote to social disorder. The problems of social disorder are waste and instability. The waste occurs when people must undo changes they didn't want with anti-spyware utilities, check software they don't trust with firewalls, and in general, act as if they are at war with their PC. The instability means Windows degrades over time, due to the ''freedom'' applications have. Programs put files wherever they like, with no accountability, and uninstalled applications are not removed cleanly. Over time, Windows becomes bloated by an ever increasing ''residue'' of left-over files and registry records. The degradation occurs when an operating system gives selfish applications too much freedom. Eventually, only reinstalling the entire operating system (and reinstalling all applications and resetting all preferences), recovers system performance.

Windows XP gives users more choice than before and so is more polite than before, but seems a trend not an arrival. Two general deficiencies are inadequate choice and uneven choice, suggesting two politeness improvements:

1. *An application source registry (ASR)*
2. *A general meta-choice console (GMCC)*

Both could increase user goodwill and reduce system degradation.

8.2 *Application source registry*

To manage their information estate, users need information, like:

1. *What* are the requested choices?
2. *Who* are the choice agents?
3. *Why* take these choices?

The Windows Task Manager shows current applications/processes and lets users end any one. The MSConfig Service shows startup applications and lets users prevent any one. In both cases, the description field answers the ''What'' question, but ''Who?'' and ''Why?'' are left for users to guess. Who, in social terms, means an accountable person or company. To help applications disclose, the operating system could offer an application source registry, where each new installation would create a source record, giving contact and other details. Users would see registered sources, like Microsoft or Acrobat, as actors in their computer estate. Sources would link to applications and applications to information resources. ''Source'' could be a property of every desktop icon, menu item, taskbar icon and hard drive listing. Clicking this property, say on a desktop icon, would show who is responsible for it. No longer would it be unclear whether a cryptic file is system critical, or just left behind by an application long ago uninstalled. One could view the resources each source allocated, and for example, delete all icons, items and files from a given source.

Polite self-disclosure need not be forced, so an application source registry could be fully backward compatible. Applications need not disclose, but a source record would still be created with the title ''anonymous''. Of course users might reject anonymous sources. Letting users know who is doing what would let the social marketplace create more polite software.

8.3 *A general meta-choice console*

In Windows XP, users can right click to add icons to the desktop (the ''New'' option), but cannot add items to the start menu or taskbar, though any installation program can do this. Does the desktop belong to the user, while the taskbar and start menu belong to applications? Why is the add choice available for some interfaces but not others?

Perhaps too many choices could overwhelm users? Offering *choice in the offering of choice* (meta-choices) could counteract such information overload. An example is the ''Do not ask me this question again'' meta-choice checkbox. Firewalls like Zone-Alarm would be unwieldy if users had to confirm every channel access every time. Meta-choices like:

1. Always accept
2. Always reject
3. Let me choose

mean users need only set repeated access permissions once. The firewall then remembers their choice, and does not pester the user.

The *initiation problem* is when users choose always reject for a choice they later want. It affects marketers and system critical requests. A new user who accidentally chooses ''Don't Ask Me Again'' for an update, might never undo the error. People change their minds, and may choose ''Never'' for offers they would later gladly accept. A general meta-choice console (GMCC) would give a common place to change meta-choices. Even so, security choices, like virus updates, may be entitled to ''nag'', letting you delay your virus update 3 days but not forever. Some choices should not be permanently turned off.

Different applications handle meta-choices in different ways, or not at all, which confuses users. A GMCC could manage all meta-choices consistently, as Windows MSConfig offers a common interface for all startup choices. The Services Computer Management Console offers these meta-choices for all running programs:

- Automatic – as software requests
- Manual – as user requests (at the time)
- Disabled – do not run

Whatever a program does, operating system meta-choices give users overriding control. Why not extend the concept to all CHI resources, putting all resource meta-choices in a single place? A general meta-choice console (GMCC) could manage all meta-choices, not just those for startup and CPU. It would record and remember all application resource permissions. If a desktop icon were deleted, the GMCC Desktop tab could put it back. An application that insisted on putting icons on the desktop every update could be denied. Permissions could be organized by social source, like Adobe or Mozilla. Like any estate manager, most users are busy people, happy to delegate. A general meta-choice console would give users the final say, and make applications more polite.

9. Discussion

Politeness is a valid software requirement, but should it be forgotten while security is an issue? Politeness and security may be two sides of the same coin of social health. By analogy, a gardener defends his or her crops from weeds, but does not wait for every weed to be killed before fertilizing. If politeness encourages social acts and security reduces anti-social acts, they are complementary not mutually exclusive.

Politeness may aid security, by addressing a source of attacks - resentment or anger against a system where the powerful are perceived to predate the weak (Power, 2000). Politeness counters this attitude by giving away choice, contradicting the idea that others take what they can in cyberspace, so I can too. When some people voluntarily offer choice, others may do the same, and those who attack society may think again. If politeness reduces anti-social acts at their bitter source, polite computing should accompany security not follow it.

It is not proposed that people be polite to software, but that software be polite to humans. That people want this seems enough reason for it to happen. Politeness seems a development of usability and human-centered computing. As users become computer-literate, the view that ''software knows best'' is hard to sustain. Perhaps once, computer users were like children, but children grow up, and as they do, they want more choice.

The unpleasant alternative to politeness is war. Today, many users feel at war with their software: removing what they didn't want added, resetting changes they didn't want changed, closing windows they didn't want opened, and deleting e-mails they didn't want to receive. User weapons in this war include third party tools like Black Ice, SpyBot, Spam-Killer, Pop-up blocker and Zone Alarm. At download sites, these are the most popular accesses. The user goal is to wrestle a semblance of control from selfish programs. This human-computer conflict has a cost.

Electronic commerce, though a billion-dollar industry, has consistently performed below expectations. In online trade, *both seller and buyer gain*. The seller reduces costs, and the buyer gets more choice. If electronic commerce benefits both customers and companies, why is it not the majority of all trade? Every day millions of customers seek hundreds of thousands of products and services from web sites that offer them. Yet most purchase from brick and mortar rather than online sources (Salam, Rao, & Pegels, 2003). Something is wrong.

Customer conflict is not the norm of brick and mortar businesses, and should not be so for online business. Social-technical systems are both social and technical. Their success depends on the user response as well as the compiler response. Making choices is what computers do, but it is also what people do. To avoid choice conflict, software must be polite. That people control their computers, not the reverse, is a non-negotiable HCI requirement. Software designers should not underestimate the importance of choice to people. In human history, freedom and choice have been the stuff of revolutions, so a grass-roots Internet movement against impolite software is not inconceivable.

If customers prefer polite to impolite software, politeness may be a critical software success factor. Polite computing could be taught in system design classes, along with other system requirements. This would require an information specification of politeness, such as that polite software offers rather than takes choice, discloses rather than hides its social source, helps rather than hinders user choice, and remembers rather than forgets past interactions. Perhaps a ''seal of politeness'' could credit applications that satisfy these specifications. Polite computing can transfer what society knows to what IS designers do. Software designers could benefit from one thing that physical society has learned over two thousand years: that in the long run, *politeness pays*.

References

Brown, P., & Levinson, S. C. (1987). *Politeness: Some universals in language usage*. Cambridge: Cambridge University Press.

Cooper, A. (1999). *The Inmates are Running the Asylum- Why High Tech Products Drive us Crazy and How to Restore the Sanity*. USA.

Dawkins, R. (1989). *The Selfish Gene* (2nd ed.): Oxford University Press.

Editor. (2002, Sunday, Feb 24, section 4). Technology threats to privacy. *New York Times,* pp. 12.

Erickson, T., & Kellog, W. (2000). Social translucence: An approach to designing systems that support social processes. *ACM Transactions on Computer-Human Interaction,* **7**, 59–83.

Friedman, B., Howe, D. C., & Felten, E. (2002). *Informed Consent in the Mozilla Browser: Implementing Value-Sensitive Design*. Paper presented at the Hawaii International Conference on the System Sciences, Hawaii.

George, J. F. (1996). Computer-based monitoring: Common perceptions and empirical results. *MIS Quarterly, December*, 459–480.

Goldstein, M., Alsio, G., &Werdenhoff, J. (2002). The media equation does not always apply: People are not polite to small computers. *Personal and Ubiquitous Computing,* **6**, 87–96.

Hardin, G. (1968). The tragedy of the commons. *Science,* **162**, 1243–1248.

Horvitz, E. (2004). *Lumiere Project: Bayesian Reasoning for Automated Assistance*. Available: http://reserach.microsoft.com/~horvitz/lum.htm [2004].

Langer, E. J. (1975). The illusion of control. *Journal of Personality and Social Psychology,* **32**, 311–328.

Levitt, J. (2001). *Internet Zone: Good help is hard to find*. Information Week: Listening Post. Available: http://www.informatonweek.com/835/35uwjl.htm [2004].

Marckwardt, A. H., Cassidy, F. G., & McMillan, J. G. (Eds.). (1992). *Webster Comprehensive Dictionary: Encyclopedic Edition*. Chicago: J. G. Ferguson Publishing Company.

Miller, C. A. (2004). Human-Computer Etiquette: Managing expectations with intentional agents. *Communications of the ACM,* **47**, 31–34.

Nass, C. (2004). Etiquette Equality: Exhibitions and expectations of computer politeness. *Communications of the ACM,* **47**, 35–37.

Nissenbaum, H. (1997). Toward an approach to privacy in public: Challenges of information technology. *Ethics and Behavior,* **7**, 207–219.

PCMagazine. (2001). *20th Anniversary of the PC Survey Results*. Available: http://www.pcmag.com/article2/0,1759,57454,00.asp [2004].

Poundstone, W. (1992). *Prisoner's Dilemma*. New York: Doubleday, Anchor.

Power, R. (2000). *Tangled Web: Tales of digital crime from the shadows of cyberspace*. Indeanapolis: QUE Corporation.

Pratley, C. (2004). *Chris_Pratley's One Note WebLog*. http://weblogs.asp.-net/chris_pratley/archive/2004/05/05/126888.aspx [2004].

Privacy-International. (2002). *Big Brother Awards International*. Available: http://www.bigbrother.awards.at/org/ [2002].

Reeves, B., & Nass, C. (1996). *The Media Equation: How people treat computers, television, and new media like real people and places*. New York: Cambridge University Press/ICSLI.

Ridley, M. (1996). *The Origins of Virtue: Human Instincts and the Evolution of Cooperation*. New York: Penguin.

Rose, E. (2001). Balancing internet marketing needs with consumer concerns: A property rights framework. *Computers and Society, March*, 17–21.

Salam, A. F., Rao, H. R., & Pegels, C. C. (2003). Consumer-Perceived Risk in E-Commerce Transactions. *CACM,* **46**.

Shectman, N., &Horowitz, L. M. (2003). *Media inequality in conversation: How people behave differently when interacting with computers and people*. Paper presented at the CHI (Computer Human Interaction) 2003, Ft Lauderdale, Florida.

Whitworth, B., & deMoor, A. (2003). Legitimate by design: Towards trusted virtual community environments. *Behaviour & Information Technology,* **22**, 31–51.

Whitworth, B., Gallupe, B., & McQueen, R. (2001). Generating agreement in computer-mediated groups. *Small Group Research,* **32**, 621–661.

Whitworth, B., Gallupe, B., & McQueen, R. J. (2000). A cognitive three process model of computer-mediated groups: Theoretical foundations for groupware design. *Group Decision and Negotiation,* **9**, 431–456.

Whitworth, B., & Whitworth, E. (2004). Reducing spam by closing the social-technical gap. *Computer* (October), 38–45.

Wright, R. (2001). *Nonzero: The logic of human destiny*. New York: Vintage Books.

Zigurs, I., Buckland, B., Connolly, J. R., & Wilson, V. (1999). A test of task-technology fit theory for group support systems. *The DATA BASE for Advances in Information Systems,* **30**, 34–50.