We enter a shop and are greeted with a smile from the clerk behind the counter. This informs us that we’re welcome but also implies that she has recognized our presence and that we will be served as soon as possible. No words need to be exchanged. When meeting people, we say “hello,” “hola,” or “bonjour”—a polite greeting that serves as a code to initialize conversation. In addition, we use facial expressions and gestures, such as smiling and shaking hands, to emphasize our intentions. When we leave, we say “goodbye,” “adiós,” or “adieu,” indicating that our meeting is at an end.

Thus, many of the polite phrases we use in everyday conversation also have an important use as protocols. Without them, we might miss the start of a conversation, we might not know that our presence is recognized, or we might not realize that the other party has more to say before we leave. With these phrases, we acknowledge that we understand the other party. Confirmation is given by a nod, or what we say in return. Eye contact is an indication that we (probably) are listening. Losing eye contact, we expect that the other party didn’t hear what we were saying. Giving no response at all would be considered very strange or rude. When using the telephone, these protocols are even more important since we don’t have the clues provided by gestures. Try to avoid the initial “hello” the next time you take a call, and see what happens.

Politeness, both as a form of respect and as a protocol, is as important in the virtual world as in the physical. While we can let technology handle the low-level communication processes when using e-mail, Web forms, or the short message service (SMS), we must explicitly add these functional higher-level “protocols of politeness.” This is essential both to show respect for other human beings and to ensure smooth and trouble-free communication.

FEEDBACK
Asynchronous communication devices have many advantages. We can send e-mail or use SMS whenever it suits us, and recipients can reply whenever it suits them. The drawback is that we don’t get any instant acknowledgment that the other party really has received the message. E-mail may therefore be a cause of anxiety. We can check the sent box to see that we have done our part, but we have no idea of what has taken place at the other end. Did I send the message? Was it the right address? Was it intercepted by a spam filter? Was the message too long or did it have attached files that weren’t accepted by the recipient’s security systems? Perhaps there was a technical error? Explicit acknowledgment is required. This should be no problem when we receive personal mail. A simple reply containing a sentence or two can act as an acknowledgment that a message was received. In more formal situations, politeness is often just a question of setting up the correct routines. For example, every message sent to job@company.com could get an automatic “message received” with a description of how the job application will be handled and when the applicant can expect a full answer. Some companies have a routine like this, but many are impolite and don’t offer an acknowledgment.

MANAGING ERRORS
Data from customers is often entered in a form and sent to a webservice that uses a back-office system. This setup works fine until the system fails and the save operation can’t be executed. Still, many webservers continue to provide the forms, working as though everything is up and running.
CHOICE OF LANGUAGE

In marketing school, students are taught to speak the customer's language. Interface designers should take this lesson to heart. A computer interface should only present terminology users can understand, in the language they’ve selected. This isn’t always easy to achieve. Word processors might use the term “document” instead of the more technical word “file,” but in error situations they can still tell the user “file not found.” Many Web systems use confusing terminology, employing the term “booking number” on one page and “reference number” on the next.

A computer interface should only present terminology users can understand, in the language they’ve selected.

Most websites allow the customer to choose a language. However, the choice of language is often inconsistent. When I bought a PC from Dell, the process was initiated in Norwegian, the technical specifications were in English, and the order confirmation came in Swedish. This made me recollect a Calvin and Hobbes comic strip in which Hobbes found a user manual extremely complex, as reading it seemed to require competence in many languages.

RUDE SYSTEMS

One brutal method to express superiority is to ask questions that can’t be answered. Operating systems do this when a user tries to install programs, plug-ins, or certificates. The warnings may be severe, explaining that data might be lost or the PC might stop working if the user continues the installation. Still, a user who has initiated the process to access an Internet banking account or increase the PC’s functionality knows that the only option is to continue the installation process. We can be glad that we still don’t have these systems in cars. Imagine what kind of disclaimers and warnings we would have to go through just to start the vehicle!

System arrogance can also be expressed by asking the user to remember complex sequences, such as passwords or reference numbers. My college’s employee self-service system asks for passwords of eight characters that must include letters, numbers, and special characters. And, of course, the password has to be changed after 120 days. If we forget the password, the system is service-oriented and will send us a new password via e-mail, but this option makes the whole login process superfluous.

It has been said that usability can be measured as the inverse of the length of the customer identification number companies provide on invoices—that is, the number that we enter when paying bills. Norwegian tax authorities ask me to enter 19 digits when paying my taxes, this in a population of 5 million. That is, they could receive 10,000 payments a day from every person in the world and would still have unique Norwegian taxpayer IDs for 300 years.

ARROGANT SYSTEMS

One brutal method to express superiority is to ask questions that can’t be answered. Operating systems do this when a user tries to install programs, plug-ins, or certificates. The warnings may be severe, explaining that data might be lost or the PC might stop working if the user continues the installation. Still, a user who has initiated the process to access an Internet banking account or increase the PC’s functionality knows that the only option is to continue the installation process. We can be glad that we still don’t have these systems in cars. Imagine what kind of disclaimers and warnings we would have to go through just to start the vehicle!

System arrogance can also be expressed by asking the user to remember complex sequences, such as passwords or reference numbers. My college’s employee self-service system asks for passwords of eight characters that must include letters, numbers, and special characters. And, of course, the password has to be changed after 120 days. If we forget the password, the system is service-oriented and will send us a new password via e-mail, but this option makes the whole login process superfluous.

It has been said that usability can be measured as the inverse of the length of the customer identification number companies provide on invoices—that is, the number that we enter when paying bills. Norwegian tax authorities ask me to enter 19 digits when paying my taxes, this in a population of 5 million. That is, they could receive 10,000 payments a day from every person in the world and would still have unique Norwegian taxpayer IDs for 300 years.
desktop, a new menu in Word, or automatic updates?” Then at least we'd feel that we're in control.

**INTERRUPTING**

It’s discourteous to interrupt. We wait until the other party has finished talking or try to time our interjection as smoothly as possible. This is much more difficult to achieve in the virtual world, when context might not be known. While some polite programs understand this, many behave like a bull in a china shop. Messages pop up at the most inconvenient time. While we’re giving a speech using a presentation program, the computer might tell us that new mail has arrived, offer to clean the desktop or, worse, require a reboot.

There are some situations in the virtual world in which a direct interruption is allowed. For example, we must accept the operating system interrupt telling us that the laptop’s batteries are running low. Nevertheless, as in the real world, the offense of barging in must be balanced with the message’s importance.

**LONG-STANDING CUSTOMERS**

When consulting your long-time family doctor, you would be quite astonished if asked for your name. At some places we expect to be recognized. But it’s only in the movies that they say, “Welcome Mr. Bond—the usual, shaken not stirred?” However, polite computer systems can offer some conciliation.

The airline that you use frequently could show the availability of flights between your typical destinations, perhaps on its homepage, based on your IP address or a cookie. It should retain all the data it needs and try to book your favorite seat automatically. When you call the airline, the booking system could automatically retrieve the information you need. Thus, I could be greeted with, “Hello, Mr. Olsen. Yes, your flight tonight is confirmed.”

But we can do even better. For example, I’m in an important meeting that’s running overtime and have to change my booking to a later flight. Using my cell phone, I first look for the e-mail with the reference number for the ticket; then I go into the airline’s webpage, navigate to booking, type the reference number, select the correct menu item, and choose a new flight. This is an awkward process to perform on a cell phone, and I can’t participate in the meeting while this goes on. But all the data is already there. Using SMS to contact the airline saying “later flight” should be sufficient in such a case.

As customers, we don’t expect to experience rudeness in the physical world. Instead, helpful employees are there to offer personal service, a smile, and help whenever needed. This hasn’t come out of nothing. While most people are polite, to ensure that customers are treated correctly, companies offer their employees courses in customer handling, focus on the importance of politeness, and enforce standards.

On the Web or in other virtual worlds, customers are on their own, perhaps with limited experience in using the new technology. The communication channels are more restricted, the system might not know the context, and misunderstandings aren’t so easy to clear up. However, from the few examples provided here, we see that it’s not difficult to program both helpful and polite computer systems. As in the real world, there’s a need to focus on service. Then we can implement systems that reassure users instead of antagonizing them, help instead of confuse, and efficiently aid users to avoid errors. Since the infancy period of computing is over, we must call the companies that don’t offer this level of service impolite.

**Kai A. Olsen** is a professor at the University of Bergen and Molde University College in Norway and an adjunct professor at the School of Information Services, University of Pittsburgh. Contact him at kai.olsen@himolde.no.