16. Network-Based Interaction

Alan Dix

vfridge limited, aQtive limited and Lancaster University

alan@hcibook.com

http://www.hcibook.com/alan

16.1. Introduction

In some ways this chapter could be seen as redundant in a HCI book – surely networks are just an implementation mechanism, a detail below the surface, all that matters are the interfaces that are built on them. On the other hand, networked interfaces, especially the web, but increasingly also mobile devices, have changed the way we view the world and we view society. Even those bastions of conservatism, the financial institutions have found themselves in sea-change and a complete re-structuring of the fundamentals of businesses ... just an implementation detail.

16.1.1. structure

The chapter will begin with a brief overview of **types of networks** (section 16.2) and then deal with network-based interaction under four main headings:

- **networks as enablers** (16.3)
  things that are only possible with networks
- **networks as mediators** (16.4)
  issues and problems because of networks
- **networks as subjects** (16.5)
  understanding and managing networks
- **networks as platforms** (16.6)
  algorithms and architectures for distributed interfaces

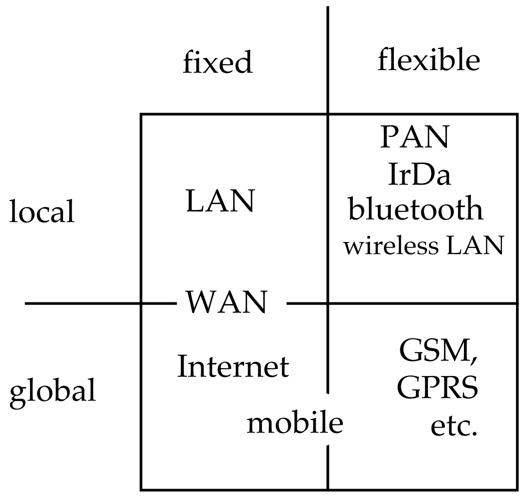
In addition, there will be a section (16.7) taking a broader view of the **history and future** of network interaction and the **societal effects and paradigm changes** engendered, especially by more resent developments in global and wireless networking.

16.2. About Networks

The word network will probably make many think of accessing the Internet and the web. Others they may think of a jumble of ethernet wires between the PCs in their office. In fact, the range of networking standards including physical cabling (or lack of cabling) and the protocols that computers use to talk down those cables is extensive. Although most of the wire-based networks have been around for some time, they are in a state of flux, due to increases in scale and the demands of continuous media. In the wireless world things are changing even more rapidly with two new generations of data service being introduced over the next 2 years.

As an aid to seeing the broader issues surrounding these changing (and, in some cases potentially ephemeral) technologies, we can use the following two dimensions to classify them:

- **global vs. local**
  How spatially distant are the points connected – ranging from machines in the same room (IrDa, bluetooth), through those in a building/site (LAN) to global networks (Internet, mobile phone networks)
- **fixed vs. flexible**
  How permanent are the links between points of the network, from physically fixed machines, to self reconfiguring devices that recognise other devices in their vicinity.



The fixed vs. flexible dimension is almost, but not quite terrestrial vs. wireless. The 'not quite' is because fixed networks increasingly involve wireless links. Also, it is often possible, when visiting another organisation, to plug a portable computer into a (wired) ethernet network and find you have access to the local printers, Internet connections etc. – flexible wire-based networking.

Let's look at a few Network technologies against these dimensions. Traditional office LANs (local area networks) are squarely in the local–fixed category whereas the Internet is largely global–fixed. Corporate WANs (wide area networks), connecting offices within the same national or international company, sit somewhere between.

Mobile phones have been placed within the global–fixed category as well. This may seem strange – the phone can go anywhere. However, the interconnections between phones are fixed and location-independent. If two mobile phones are in the same room it is no easier to connect between them than if they at opposite ends of the earth (bar a shorter lag time perhaps).

Similarly, the Internet although increasingly accessible through mobile devices and phones is largely based on fixed domain names, IP numbers and URLs.

Given the placement of mobile phones is a little ambiguous and it is possible to detect the location of phones and thus deliver location based content, some of the phone technologies have been listed in the global–flexible category (GSM and GPRS are the names of the first and second generation data services). There is obviously a steadily increasing data rate and third generation services are looking towards being able to cope with heavy media content (a Hewlett-Packard advert has two climbers at the top of a mountain watching a television soap over a mobile connection). However, the most significant differences are the charging and connectivity model. With GSM you connect when required to the Internet and this is treated like any other telephone call, usually meaning pay per minute whilst connected. In contrast GPRS is based on sending small packets of data (the P in the acronym). The connection to the Internet is treated as 'always on' and packets of data are sent to or from the phone as required. Charging is also typically by data use.

In the local–flexible category there is a host of existing and emerging technologies. At the most mundane are wireless ethernet networks (so called 802.11 devices such as Apple airport or PC WaveLAN cards) [[IEEE 802.11]]. These merely treat the machine the same as if plugged into the local fixed network. At a more local scale, infra-red (IrDa) enabled devices can talk to one another if their infra-red sensors are within line-of-sight. These are also more flexible as a PDA placed near a mobile phone will be able to use the phone's modem with little intervention (except perhaps to carefully align the devices).

Bluetooth has been much hyped and uses radio to connect close devices in a similar fashion to IrDa (but faster and more intelligently) [[Bluetooth, 2001]]. For example, a bluetooth hands-free headset can connect to a bluetooth phone without having to plug in with a piece of wire.

Finally, research in wearable computers have suggested using the body itself as the connection between worn devices in a personal area network (PAN) [[Zimmerman, 1996]]. Not only is the future networked, but we will become the network!

On the whole we have seen in the last 10 years the main focus of network-based interaction has moved anti-clockwise in this picture from fixed/local networks (mainly LAN), through fixed global networks (the Internet and web explosion), through global mobile networks (mostly phone-based, but including WAP, i-mode etc.) and moving towards flexible local connections between devices.

16.3. Networks as Enablers things that are only possible with networks

It can be the case that the network is no more than an implementation detail – for example, using a networked disk rather than a local one. However, there are also many applications which are only possible because the network is there, for example video-conferencing. The key feature of networks is the access to remote resources of some kind or other.

16.3.1. remote resources

Four kinds of remote things are made accessible by networks:

- **people**
- **physical things**
- **data**
- **computation**

These may be remote because they are far away from where you normally are, or because you are yourself on the move and hence away from one's own resources (colleagues, databases etc.). Hence **mobility** can create a need for any or all the above.

people

Networks mean we can communicate and work with others in distant places. This is often a direct action – emailing someone. engaging in a video-conference. These are all the normal study of CSCW and groupware (see chapter 29).

Interaction with remote people may also be indirect. Recommender systems gather information about people's preferences and use this to suggest further information, services or goods based on your own preferences and those of others who have similar tastes [[Resnick, 1997]]. Because the people making recommendations are in different locations from each other, the data on who selected what must be stored centrally.

Collaborative virtual environments also offer the ability for remote people to interact, but by embedding them within an apparently local virtual reality world. Although the people you are dealing with may be half a world away, their avatar (a virtual presence, perhaps a cartoon character, photo, or robot-like creature) may seem only a few yards or metres away in then virtual world.

physical things

We can also view and control remote things at a distance. For example, live web cams in public places allow us to see things (and people) there. Similarly the cameras mounted around rockets as they prepare to take off (and then usually destroyed during the launch) allow the mission controllers to monitor critical aspects of the physical system as do the numerous telemetrv sensors which will also be related via some sort of closed network. And of course the launch command itself will be relayed to the rocket by the same closed network as will the ongoing mission, perhaps the Mars robots, via wireless links.

In the rocket example it would be dangerous to be in the actual location, in other circumstances it is merely expensive or inconvenient. Telescopes are frequently mounted in distant parts of the world where skies are clearer than those above the laboratories to which they belong. In order to avoid long international trips to remote places, some of these now have some form of remote control and monitoring using the Internet [[Lavery, 1994]].

At a more personal level the systems within certain high end cars are controlled using a within car network (called CAN). Even an adjustable heated seat may require dozens of control wires, but with a network only one power and one control cable is needed. The engine management system, lighting assemblies, radio, CD player, wind screen wipers, each have a small controller that talks through the network to the drivers console (although critical engine systems will usually have a separate circuit).

Many household appliances are now being made Internet-ready. In some cases this may mean an actual interface – for example an Internet fridge that can scan the bar-codes of items as you put them in and out and then warn you when items are getting out of date, generate a shopping list of items for you and even order from your favourite store [[Electrolux, 1999]]. Others have instead or in addition connectivity for maintenance purposes, sending usage and diagnostic data back to the manufacturer so that they can organise service or repair visits before the appliance fails in some way.

data

Anyone using the web is accessing remote data. Sometimes, data is stored remotely purely for convenience, but often data is necessarily stored remotely:

- because it is shared by many remote people;
- because reasons of control, security or privacy demand remote storage;
- because it is used by a single user at different locations (web email);
- because it is too extensive to be stored locally (e.g. large databases and fat client/server).

In the case of the web the data is remote because it is accessed by different people at different locations, the author(s) of the material and all those who want to read it.

Even the web is quite complex we may perceive a web page as a single entity, but in fact it exists in many forms. The author of the page will typically have created it off-line on their own PC. They then upload the page (which effectively means copying it) onto the web server. Any changes the author makes after uploading the page will not be visible to the world until it is next uploaded. When a user wants to see the page and enters a URL or clicks on a link their browser asks the web server for the file which is then the browser has a copy as you can disconnect from the Internet and still scroll within the file. If you access the same page again quite soon, you browser may choose to use the copy it holds rather than going back to the web server, again potentially meaning you see a slightly out-of-date copy of the page. Various other things may keep their own cached copies including web proxies and firewalls.

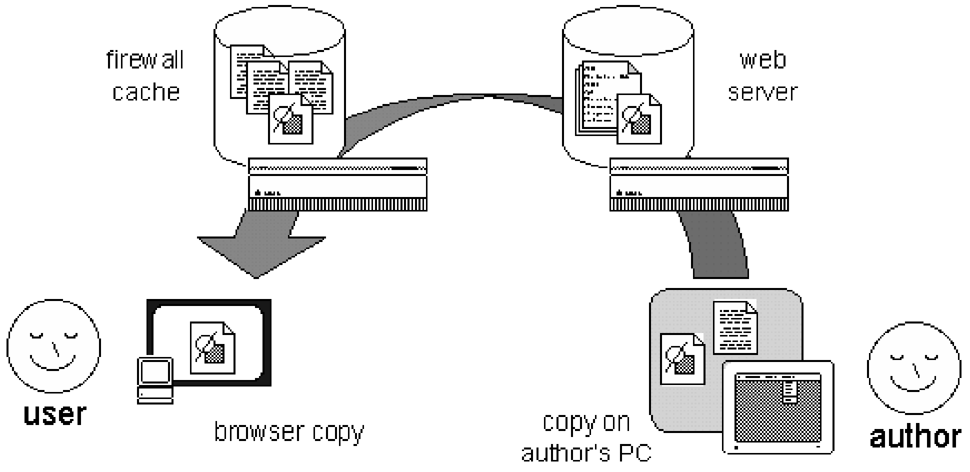


Figure 16.1 copies of a web page in many places

This story of copied data in various places is not just about the web, but true to some extent or other of all shared networked data. With people or physical things, we do not expect to have the actual person or thing locally, just a representation. This is equally true for shared data, except that the representation is so much like the 'real thing' it is far less obvious to the user.

And for shared networked data even the 'real thing' may be problematic. – there may be no single 'golden copy', but instead many variants all with equal right to be called 'the real data'.

You don't even escape networking issues if you only access data locally on your own PC – networking issues may still arise if your data is backed-up over the network.

computation

Sometimes it is remote computational resources that are accessed over the network. The most obvious example of this are large supercomputers. These have enormous computational power, and scientists wishing to use them will often pre-book time slots to perform particularly intensive calculations such as global weather simulations, analysis of chemical structure, stress calculations, etc. Because these machines are so expensive programs for them are typically developed on other less powerful computers and then uploaded over the network when the supercomputer is available.

If the data required as input or output for the calculation is not too great 'fairly simple' means can be used to upload the programs and data. However, some calculations work on large volumes of data – for example, data from microwave readings of the upper atmosphere to probe the ozone hole generate terabytes (millions of millions of bytes) of data per second. High-capacity networks are being created in many countries to enable both high-volume data for this sort of application and also the expected data required for rich media [[Foster, 1999, 2000; GRID, 2001]].

Sometimes calculations need to be performed centrally, not because the central computer is powerful, but because the local device is a computational lightweight. For example, one may want to create a remote analysis package where engineers in the field enter data into a PDA or phone interface, but where complex stress calculations are carried out on a small server back in the office. The data on materials and calculations involved may not be extensive by supercomputer standards, but may still be too much for a hand-held device.

Because transporting large volumes of data is not always practical, calculations are often performed where the data is. (In performing any computation program, data and computational engine must all be in the same place. If they aren't together, then one or other must moved or be copied to bring them together [[Ramduny, 1997]].) For example, when you perform a database access the request for the data is usually transmitted to the database server as an SQL query, for example, "SELECT name, salary FROM payroll WHERE salary > 70000". In principle the complete contents of the payroll database could be downloaded to your PC and the selection of appropriate records carried out locally, however, it would be more costly to transmit the data hence the calculation is effectively transmitted to the database server.

Even when the volume of data is not large or the frequency of access would make it cost effective to transmit it, security or privacy reasons may prevent the download of data. For example, some datasets are available to search to a limited degree on the web, but charge for a download or CD of the complete dataset. My own hcibook.com site [[Dix, 1998]] is rather like this, allowing searching of the book's contents online and displaying portions of the text, but not allowing a full download as readers are expected to buy the book!

Security considerations may also prohibit the distribution of programs themselves if they contain proprietary algorithms. Also if the source of the program is not fully trusted, one may not want to run these programs locally. The latter is the reason that Java applets are run in a software 'sandbox' confining the ability of the applet to access local files and other potentially vulnerable resources.

SETI is an interesting example of remote computation [[SETI@home]]. Normally remote computation involves a device of low-computational power asking a central computer to do work for it. In the case of SETI large calculations are split up and distributed over large numbers of not particularly powerful computers.

For those who haven't come across it the SETI (the Search for Extra-Terrestrial Intelligence) project is analysing radio signals from outer space looking for patterns or regularities that may indicate transmissions from an alien civilisation. You can download a SETI screensaver that performs calculations for SETI when you are not using your machine. Each SETI screensaver periodically gets bits of data to analyse from the central SETI servers and then returns results. This means that the SETI project ends up with the combined computational resources of many hundreds of thousands of PCs.

The same technique is used in 'PC farms'. These are when large numbers of PCs are networked together to act as a form of super-computer. For example, in CERN (the home of the web), data from high-energy collisions may consist of many megabytes of data for each event, with perhaps hundreds of significant events per second [[CERN]]. The data from each event is passed to a different PC which then performs calculations on the data. When the PC finishes it stores its results and then adds itself back to a pool of available machines.

During coming years we are likely to see both forms of remote computation. As devices become smaller and more numerous, many will become simply sensors or actuators communicating with central computational and data servers (although central here may mean one per room, or even one per body). On the other hand, inspired by SETI several companies are pursuing commercial ways of harnessing the spare, and usually wasted, computational power of the millions of home and office PCs across the world.

16.3.2. applications

The existence of networks, particularly the global networks offered by the Internet and mobile phone networks, have made many new applications possible and changed others.

Several of the more major application areas made possible by networks are covered in their own chapters: groupware (chapter 29), online communities (chapter 30), mobile systems (chapter 32), e-commerce (chapter 39), telecommunications (chapter 40) and, of course, the web (chapter 37).

In addition, networking impinges on many other areas. Handheld devices (chapter 32) can operate alone, but are increasingly able to interact with one another and with fixed networks via wireless networking. Similarly wearable computing (chapter 33) are expected to be interacting with one another via short-range networks, possibly carried through our own bodies (makes mobile phones seem positively safe!) and information appliances (chapter 38) will be Internet connected to allow remote control and maintenance. In the area of government and citizenship (chapter 41) terms such as e-democracy and e-government are used to denote not just the technological ability to vote or access traditional government publications online, but a broader agenda whereby citizens feel a more intimate connection to the democratic process. Of course, education, entertainment and game playing are also making use of networks.

Throughout the chapter we will also encounter broader issues of human abilities, especially concerned with time and delays, involving aspects of virtually all part II (human perception, cognition, motor skills etc.). Also we will find networking raises issues of trust and ethics (chapters 65 and 62) and of course the global network increases the importance of culturally and linguistically accessible information and interfaces (chapter 23).

Networking has already transformed many people's working lives allowing telecommuting, improving access to corporate information whilst on the move and enabling the formation of virtual organisations. Networks are also allowing whole new businesses areas to develop, not just the obvious applications in e-shopping and those concerned with web-design.

The Internet has forced many organisations to create parallel structures to handle the more direct connections between primary supplier and consumer (disintermediation). This paradoxically is allowing more personalised (if not personal) services and often a focus on customer–supplier and customer–customer communication [[Siegal, 1999; Light, 2001]]. This restructuring may also allow the more flexible businesses to revolutionise their high street (or mall) presence – allowing you to buy shoes in different sizes, or next day fitting services for clothes [[Dix, 2001]].

The complexity of installing software and the need to have data available anywhere at any time has driven the nascent application service provider (ASP) sector. You don't install software yourself, but use software hosted remotely by providers who charge on a usage rather than once-off basis. By storing the data with third-parties and organisation can off-load the majority of its backup and disaster management requirements.

16.4. Networks as Mediators issues and problems because of networks

This section takes as a starting point that an application is networked and looks at the implications this has for the user interface. This is most apparent in terms of timing problems of various kinds. This section is really about when the network is largely not apparent except for the unintended effects it has on the user.

We'll begin with a technical introduction to basic properties of networks and then see how these affect the user interface and media delivery.

16.4.1. Network Properties

bandwidth and compression

The most commonly cited network property is **bandwidth** – how much data can be sent per second. Those who have used dial-up connections will be familiar with 56K modems, and those with longer memories or using mobile phone modems may recall 9.6K modems or less. The 'K' in all of these refers to thousands of bits (0/1 value) per second (strictly Kbps) rather than bytes (single character) that are more commonly seen in disk and other memory sizes. A byte takes 8 bits, taking into account a small amount for overhead, you can divide the bits per second by 10 to get bytes per second.

Faster networks between machines in offices are more typically measured in mega bits per second (again strictly Mbps but often just written M) – for example, the small 'telephone cable' ethernet is rated at either 10Mbps or 100Mbps.

As numbers these don't mean much, but if we think about them in relation to real data the implications for users become apparent.

A small word processor document may be 30 Kb (kilo bytes). Down a 9.6K modem this will take approximately half a minute, on a 56K modem this is reduced to 5 seconds, for a 10Mb ethernet this is 30 milliseconds. A full screen web quality graphic may be 300Kb taking 5 minutes of 9.6K modem, less than a minute on a 56K modem or 1/3 second on 10Mb ethernet. (N.B. these are theoretical minimum times if there is nothing else using the network.)

Note that I am using the formula:

download time $T = \frac{F \times 10}{M}$

where:
F = size of file in bytes
10 is the number of raw bits per byte
M = modem speed in bits per second

Rich media such sound or video put a greater load again. Raw, uncompressed HI-FI quality sound needs over 200 kilo bits per second and video tens of mega bits per second. Happily there are ways to reduce this otherwise digital AV would be impossible over normal networks.

Happily real media data has a lot of redundant information – areas of similar colour in a picture, successive frames in a video are similar, sustained notes in music. **Compression** techniques use this similarity to reduce the actual amount of data that needs to be sent (e.g. rather than sending a whole new frame of video, just send the differences from the last frame). Also some forms of compression make use of human perceptual limits: for example, MP3 stores certain pitch ranges with greater fidelity than others as the human ear's sensitivity is different at different pitches [[MPEG, 2001]], also JPEG images give less emphasis to accurate colour hue than the darkness/lightness [[JPEG, 2001]]. Between them these techniques can reduce the amount of information that needs to be transferred significantly, especially for richer media such as video. Thus the actual bandwidth and the effective bandwidth, in terms of the sorts of data that are transmitted may be very different.

latency and start-up

Bandwidth measures how much data can be transferred – **latency** is how long each bit takes. In terms of a highway: bandwidth would be how many lanes and latency is the time it takes to travel the length of the highway. The latency is due to two factors. The first the speed of transmission of electricity through wires or light through optical networks. This may seem insignificant, but for a beam of light to travel across the Atlantic would take 20ms and in practice this hop takes more like 70ms. For satellite based communications the return trip to an from a geostationary satellite takes nearly a second. Think about the typical delay you can hear on a trans-continental telephone call. The second factor contributing to latency is that every electronic switch or computer routing signals will have to temporarily store, then decide what to do with the signal before passing it on to the next along the chain. Typically this is a more major influence and in practice trans-Atlantic Internet traffic will take nearer 250ms from source to final destination, most of which in various computer centres at one end or other.

Latency is made worse by **set-up time**. Every time you establish an Internet connection a conversation is established between your computer and the machine hosting the web-server: 'hello are you there', 'yes I'm here what do you want', 'I'd like to send you some data', 'great I'm waiting', 'OK here it is then' (this is called handshaking). Each turn in this conversation involves a round trip, network latency on both outward and return paths and processing by both computers. And this is before the web server proper even gets to look at your request. Similar patterns happen as you dial a telephone call.

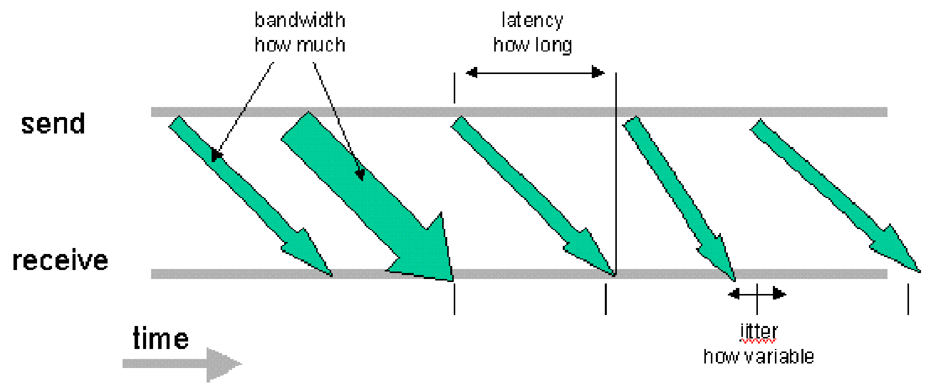


Figure 16.2 bandwidth, latency and jitter

Latency and set-up time are critical as they often dominate the delay for the user except for very large files or streaming audio/visual media. Early web design advice (by those concerned about people with slow connections, but who clearly had never used one!) used to suggest having only as much text that would fit on a single screen. This was intended to minimise the download time. However, this ignores set-up times. A long text page doesn't take long to load even on a slow connection, once the connection to the web server has been established. Then it is far faster to scroll in the browser than to click and wait for another small page to load. A similar problem is the practice of breaking large images up into a jigsaw of small pieces. There are valid reasons for this - allowing rollover interaction or where parts of the image are of different kinds (picture/text) – however, it is also used without such reasons and each small image requires a separate interaction with the server encountering latency and set-up delays.

jitter and buffering

Suppose you send letters to a friend every three days and the postal service typically takes 2 days to deliver letters (the average latency in network terms). Your friend will receive letters every three days, just delayed from when you sent them. Now imagine that the postal system is a little variable, sometimes letters take 2 days, but occasionally they are faster and arrive the next day and sometimes slower and take 3 days. You continue to send letters every three days, but if a slow letter is followed by a fast one, your friend will receive them only one day apart, if on the other hand a fast letter is followed by a slow one the gap becomes five days. This variability in the delay is called **jitter**. (Note that the fast letters are just as problematic as the slow ones – a fast letter followed by a normal speed one still gives a four day gap.)

Jitter doesn't really matter when sending large amounts of data, or when sending one-off messages. However it is critical for continuous media. If you just played video frames or sound when it arrived, jitter would mean that the recording would keep accelerating and slowing down (see figure 16.3 (i) and (ii)).

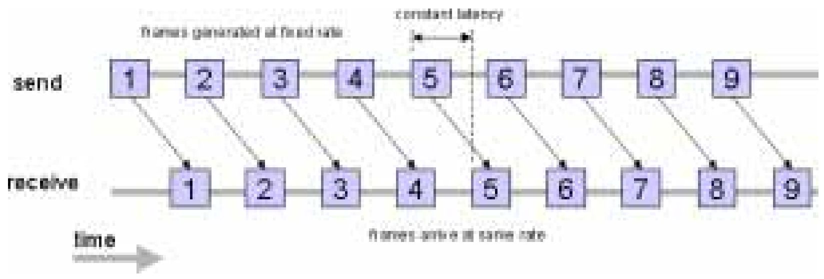


Figure 16.3 (i) no jitter – no problem

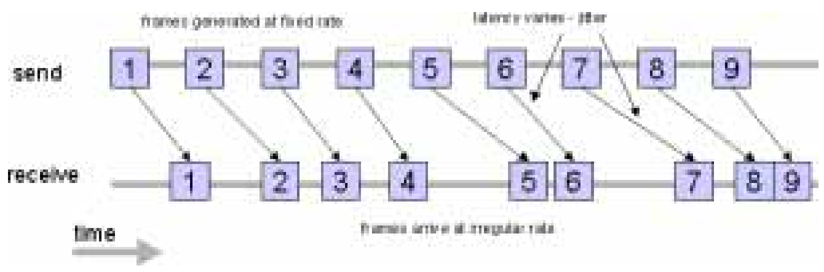

frames generated at fixed rate
latency varies - jitter
send
receive
time
frames arrive at irregular rate

Figure 16.3 (ii) jitter causes irregular reception

Jitter can be partially alleviated by **buffering**. Imagine your friend's postman holds back one letter for three days and then starts giving letters to your friend one every third day. If your mail always arrives in exactly two days the postman will always hold exactly one letter as mail will arrive as fast as he passes it on. If however a letter arrives quickly he will simply hold two letters for a few days and if it is slow he will have a spare letter to give. Your friend's mail is now arriving at a regular rate, but the delay has increased to (a predictable) five days. Buffering in network multimedia behaves exactly the same holding back a few seconds audio/video data and then releasing it at a constant rate.

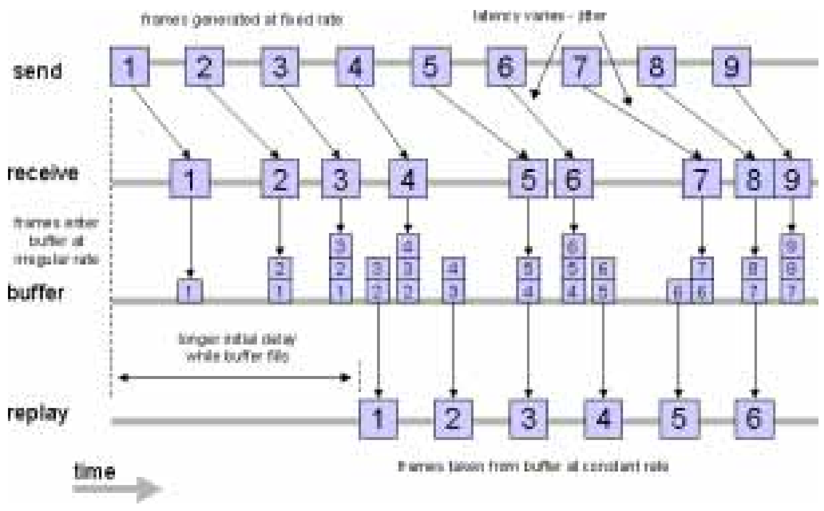

frames generated at fixed rate
latency varies - jitter
send
receive
frames enter buffer at irregular rate
buffer
longer initial delay while buffer fills
replay
time
frames taken from buffer at constant rate

Figure 16.4 buffering smooths jitter, but adds delay

reliability and loss, datagram and connection-based services

Virtually all networks are designed on the principle that there will be some loss or damage to data en-route. This arises for various reasons – sometimes there is electrical interference in a wire, sometimes the internal with. This is normal and network software is built to detect damaged data and cope with lost data.

Because of this the lowest layers of a network are assumed to be lossy. Any date damaged in transit are discarded and computers and hardware en-route can choose to discard data if they get busy. So when one computer sends a packet of data to another it can assume that if the packet of data arrives it will be intact, but it may never arrive at all.

Some network data, in particular certain forms of real-time multimedia data are deliberately sent in this unreliable, message-at-a-time, form (called **datagrams**).

However, it is usually easier to deal with reliable channels and so higher levels of the network create what are called **connection-based** services on top of the unreliable lower level service. Internet users may have come across the term TCP/IP. IP is the name of an unreliable low-level service that routes packets of data between computers. TCP is a higher level connection-based service built on top of IP. The way TCP works is that the computer wanting to make a connection contacts the other and they exchange a few (unreliable IP) messages to establish the link. Once the link is established the sending computer tags messages it sends with sequence data. Whenever the receiving computer has all the data up to a certain point it sends an acknowledgement. If the sending computer doesn't get an acknowledgement after a certain time it resends the data [[Stevens, 1998, 1999]].

With TCP the receiving computer cannot send a message back when it notices a gap it has to wait for the sending computer to resend after the timeout. While it is awaiting the resend it cannot process any of the later data. Notice what this means reliability is bought at the price of potential delays.

quality of service (QoS) and reservation

The above properties are not just determined by a raw network's characteristics, such as the length of wires, types of routers, modems etc.. They are also affected by other traffic – its volume and nature. If 10 PCs are connected to a single 10Mbps network connection and require high-volume data transfers (perhaps streaming video), then there is only, on average, 1Mbps available for each. If you are accessing a network service that requires trans-Atlantic connections during peak hours, then intermediate routers and hubs in the network are likely to become occasionally overloaded leading to intermittent packet loss, longer average delays, and more variability in delay hence jitter.

For certain types of activity, in particular real-time or streaming rich-media, one would like to be able to predict or guarantee a minimum bandwidth, maximum delay and jitter etc. These are collectively called **quality of service (QoS)** issues [[Campbell, 1997]]. Some network protocols allow applications to reserve a virtual channel with guaranteed properties, however, the most common large-scale network, the Internet, does not have such guarantees, it operates solely on a best endeavour basis. Upgrades to the underlying protocol that have been under development for some years (called by the catchy name IPv6), will allow some differentiation of different types of traffic. This may allow routers to make decisions to favour time-critical data, but it will still not be able to reserve guaranteed capacity.

encryption, authentication and digital signatures

Some networks, such as closed office networks, offer no greater worries about security of information than talking together (both are capable of being bugged, but with similar levels of difficulty). However, more open networks such as the Internet, or phone networks, mean that data is travelling through a third party and public infrastructure to get to its recipients. Increasing use of wireless devices also mean that the data sent between devices is more easily able to be monitored or interfered with by third parties. One option is to only use physically secure networks, but for economic reasons this is often not an option. Furthermore, solutions which do not rely on the network itself being secure are more robust. If you rely on, for example, a private dedicated line between two offices and assume it is secure, then if someone does manage to tap into it, all your inter-office communication is at risk.

The more common approach now is to assume the networks are insecure and live with it. This gives rise to two problems:

secrecy – how to stop others from seeing your data

security – how to make sure data is not tampered with

The first problem is managed largely by **encryption** methods – ensuring that even if someone reads all your communications they cannot understand them [[Schneier, 1996]]. The "https" in some URLs is an example of this denoting that the communication to the web server is encrypted.

The second problem, security, has various manifestations. Given communications are via a network, how do you know that you are talking to the right person/machine. **Authentication** mechanisms deal with this. In various ways they allow one machine to verify (usually by secret information that can only be known by the true intended party) that it is talking to the right party.

Even if you know that you are talking to the right person/machine, how do you know that the data you receive hasn't been changed? this is like receiving a signed letter, but unbeknown to you someone has added some lines of text above the signature, although it really comes from the person you think, the message is not as was sent. If data is being encrypted then this may often implicitly solve this problem as any tampered data is uninterpretable by a third party, who therefore cannot alter it in a meaningful way.

If secrecy is not an issue however, encryption is an unnecessary overhead and instead **digital signatures** generate a small data block that depends on the whole of the message and secret information known to the sender. It is possible for the recipient to verify that the signature block corresponds to the this are 'signed applets' where the Java code is digitally signed so that you can choose to only run Java programs from trusted parties.

16.4.2. UI properties

network transparency

One of the goals of many low-level network systems is to achieve transparency – that is to make it invisible to the user where on the network a particular resource lies. When you access the web you use the same kind of URL and same kind of interface whether the web server is in Arizona, Australia or Armenia. I know that when at home I send an email between two machines less than 2 metres apart, the message actually goes all the way across the Atlantic and back – but this is only because I have quite a detailed understanding of the computers involved – as a user I press 'send mail' on one machine and it arrives near instantaneously on the other.

Although network transparency has many advantages to the user – you don't care about routes through the network etc., there are limits to its effectiveness and desirability. Some years ago I was at a Xerox lab in Welwyn Garden City in the UK. Randy Trigg was demonstrating some new features of Notecards (an early hypertext system [[Halasz, 1987]]). The version was still under development and every so often would hit a problem and a LISP debugger window would appear. After a whole of using it, it suddenly froze – no debugger window, no error message, just froze. After a few embarrassed seconds he hit a control key and launched the debugger. A few minutes of frantic scanning through stack dumps, program traces etc., and the reason became clear to him. He had demonstrated a feature that he had last used on his workstation at Palo Alto. The feature itself was not at fault, but required an obscure font that he had on his own workstation, but not on the machine there in Welwyn. When Notecards had requested the font the system might have thrown up an error window, or substituted a similar font. However, in the spirit of true network transparency, the location of the font should not matter. having failed to find it on the local machine it proceeded to interrogate machines on the local network to see if they had it, it then proceeded to scan the Xerox UK network and world network. Eventually, if we had waited long enough, it would have been found on Randy's machine in Palo Alto.

Network transparency rarely extends to timing!

Transparency has also been critiqued for CSCW purposes [[Mariani , 1991]]. It may well be very important to users where resources and people are. For mobile computing also, an executive takes a laptop on the plane only to discover that the files needed are residing on a network file server rather than on the machine itself. If the interface hides location how can one predict when and where resources will be available.

delays and time

As is evident, one of the issues that arises again and again when considering networks is time – how long are the delays, how long to transfer data, how variable etc. Networking is not the only reason for delays in applications but is probably one of the most noticeable – the web has often been renamed the 'world-wide wait'. There is a long-standing literature on time and delays in user-interfaces. This is not as extensive as one might think, largely because for a long time the prevailing perception in the HCI community was that temporal problems would go away (with some exceptions) leading to what I called the "myth of the infinitely fast machine" [[Dix, 1987]].

One of earlier influential papers was Ben Shneiderman's review of research findings on delays [[Shneiderman, 1984]] – mainly based on command line interfaces. More recently there have been a series of workshops and special journal issues on issues of time, sparked largely by web delays [[Johnson, 1996; Clarke, 1997; Howard, 1999]].

There are three main timescales that are problematic for networked user-interfaces:

100 ms	feedback for hand-eye coordination tasks need to be less than 100–200 milliseconds to feel fluid. This is probably related to the fact that there are delays if this length in our motor-sensory system anyway. For aural feedback, the timescales are slightly tighter again.
1 second	timescale for apparent cause–effect links such as popping a window after pressing a button. If the response is faster than this the effect seems 'immediate'.
5-10 seconds	waits longer than this engender annoyance and make it hard to maintain task focus and engender annoyance. This may be related to short-term memory decay.

The 100ms time is hard to achieve if the interaction involves even local network traffic. The 1 second time is usually achievable for local networks (as are assumed by X-Windows systems), but more problematic for long haul networks. The 5-10ms time is in principle achievable for even the longest trans-continental connections, but when combined with bandwidth limitations or overload of remote resources may become problematic. This is especially evident on web-based services where the delay between hitting a link and retrieving a page (especially a generated page) may well exceed these limits, even for the page to begin to draw.

The lesson for UI designers is to understand the sort of interaction required and to ensure that parts of the user interface are located appropriately. For example, if close hand-eye coordination is required it must run locally on the user's own machine – in the case of the web in an applet, in JavaScript code etc. If the nature of the application is such that parts of the application cannot reside close enough to the user for the type of interaction required, then, of course, one should NOT simply have a slow version of (say) dragging an icon around, but instead change the overall interaction style to reflect the available resources.

Two of the factors that alleviate the effects of longer delays are predictability of the delay and progress indicators. Both give the user some sense of control or understanding over the process, especially if users have some indication of expected delay before initiating an action [[Johnson' 1997]]

The many variable factors in networked systems make predicting delays very difficult, increasing the importance of giving users some sense of progress. The psychological effect of progress indicators is exploited (cynically) by those web browsers that have progress bars that effectively lie to the user, moving irrespective of any real activity (try unplugging a computer form the network and attempting to access a web page, some browsers will hit 70% on the progress bar before reporting a problem). Other network applications use recent network activity to predict remaining time for long operations (such as large file downloads). Other solutions include generating some sort of intermediate low-quality or partial information while the full information is being generated or downloaded (e.g. progressive image formats or web pages designed to partially display).

For virtual reality using head-mounted displays, as well as hand-eye coordination tasks, we also have issues of the coordination between head movements and corresponding generated images. The timescales here are even tighter as the sensory paths are faster within our bodies hence less tolerant of external delays. The brain receives various indications of movement: the position and changes of neck and related muscles, the balance sensors in the inner ear and visual feedback. Delays between the movement of the generated environment and head movement lead to dissonance between these different senses and have an effect rather like being at sea, with corresponding disorientation and nausea. Also any delays reduce the sense of immersion – being there within the virtual environment. Early studies of VR showed that users sense of immersion was far better when they were given very responsive wire frame images than if they were given fully rendered images at a delayed and lower frame rate [[Pausch, 1991]].

coping strategies

People are very adaptable. When faced with unacceptable delays (or other user interface problems) users develop ways to workaround or ameliorate the problem – **coping strategies**. For example, web users may open multiple windows so that they can view one page whilst reading another [[McManus, 1997]] and users of 'telnet' for remote command line interfaces may type 'test' characters to see whether the system has any outstanding input [[Dix, 1994]].

Coping strategies may hide real problems, so it is important not to assume that just because users do not seem to be complaining or failing that everything is all right. We can also use the fact that users are bright and resourceful by building interface features that allow users to adopt coping strategies where it would be impossible or impractical to produce the interface response we would like. For example, where we expect delays we can ensure that continual interaction is not required (by perhaps amassing issues requiring user attention in a 'batch' fashion) thus allowing users to more easily multi-task. Unfortunately, this latter behaviour is not frequently seen – the 'myth' lives on and most networked programs still stop activity and await user interaction whenever problems are encountered.

timeliness of feedback/feedthrough, pace

Although **feedback** is one of the most heavily used terms in HCI, we may often ignore the complex levels of feedback when dealing with near instantaneous responses of GUI interfaces. In networked systems with potentially long delays we need to unpack the concept. We've already discussed some of the critical timescales for feedback. For hand-eye coordination getting feedback below the 100ms threshold is far more important than fidelity – quickly moving wire frames or simple representations are better than dragging a exact image with drop shadow.

For longer feedback cycles, such as pressing a button, we need to distinguish:

- **syntactic feedback** – that the system has recognised your action
- **intermediate feedback** – that the system is dealing with the request implied by your action (and if possible progress towards that request)
- **semantic feedback** – that the system has responded and the results obtained

The direct manipulation metaphor has lead to an identification between these levels and many systems provide little in the way of syntactic or intermediate feedback relying solely on semantic feedback. In networked systems where the semantic feedback includes some sort of remote resource, it is crucial to introduce specific mechanisms to supply syntactic and intermediate feedback otherwise the system may simply appear to have ignored the users action (leading to repeated actions with potentially unforeseen consequences) or even frozen or crashed.

This also reminds us of a crucial design rule for slow systems: wherever possible make actions idempotent – that is that invoking the same action twice should, where possible, have the same effect as a single action. This means that the 'try again' response to a slow system does not lead to strange results.

For collaborative systems or those involving external or autonomous resources (remote controlled objects, environmental sensors, software agents), we must also consider **feedthrough**. Feedback is experiencing the effect of ones own actions, feedthrough is the effect of one's own actions on other people and things and experiencing the effects of their actions yourself. For example, in an online chat system, you type a short message and press 'send' and your message appears in your transcript – feedback – then sometime later it also appears in the transcript of the other chat participants – feedthrough.

Feedback is needed to enable us to work out whether the actions we have performed are appropriate, hence (typically) need to be much quicker than feedthrough responses. This is fortunate as feedthrough by its very nature usually requires network transmission and ensuing delays. the exception to the rule that feedthrough can afford to be slower is where the users are attempting to perform some close collaborative task (e.g. positioning some items using direct manipulation) or where there is a second fast communication channel (e.g. on the telephone user A says to user B "see the red box", but the relevant item hasn't appeared yet on B's screen).

Potentially more important to users of collaborative systems than bandwidth or even raw delays is **pace** – the rate at which it is possible to interact with a remote resource or person. This is partly determined by lower level timings, but is also heavily influenced by interface design. For example, you know that someone is sitting at their desk and send them an urgent email. The time that it takes to get a response will be hardly affected by the raw speeds between your machine and your colleague, and more determined by factors such as how often the email client checks the server for new email and whether it sounds an alert when new email arrives, or simply waits there until your colleague chooses to check the in-box.

race conditions and inconsistent interface states

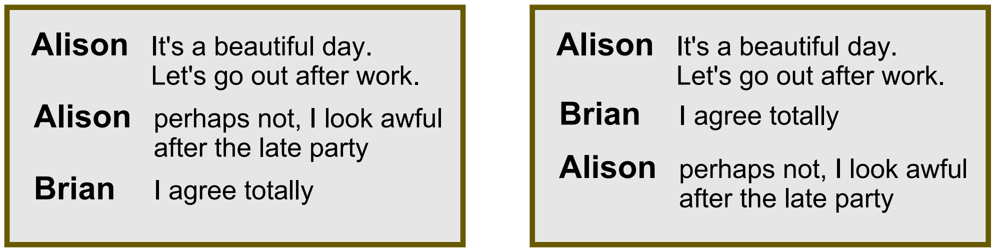
Alison and Brian are using an online chat program.

Alison writes "It's a beautiful day. Let's go out after work" and then begins to think about it.

Brain writes "I agree totally" and then has to leave to go to a meeting.

at almost the same time Alison writes "perhaps not, I look awful after the late party"

Unfortunately the messages are so close to simultaneous that both Alison and Brain's machines put their own contribution first so Alison sees the chat window as in figure 16.5.a and Brian sees it as in figure 16.5.b. Brian thinks for a few moments, and then writes "no you look lovely as ever", but unfortunately Alison never sees this as she takes one look at Brian's previous remark and shuts down the chat program.



Alison It's a beautiful day. Let's go out after work.
Alison perhaps not, I look awful after the late party
Brian I agree totally

Alison It's a beautiful day. Let's go out after work.
Brian I agree totally
Alison perhaps not, I look awful after the late party

(a) Alison's chat window (b) Brian's chat window

Figure 16.5. consistency breakdown

This type of incident where two events happen so close together that their effects overlap is called a **race condition**. Race conditions may lead to **inconsistent states** for users as in this example, or may even lead to the software crashing. Although in principle race conditions are possible however fast the underlying network, the likelihood of races occurring gets greater as the network (and other) delays get longer.

Even some of the earliest studies in collaborative systems have shown the disorienting effects of users seeing different views of their shared information space, even when this is simply a matter of seeing different parts of the same space [[Stefik, 1987]].

Consistency becomes an even greater problem in mobile systems where wireless connections may be temporarily lost, or devices may be unplugged from fixed networks whilst on the move. During these period of **disconnection**, it is easy for several people to be updating the same information leading to potential problems when their devices next become network connected.

In section 16.6 we will discuss mechanisms and algorithms that can be used to maintain consistency even when delays are long and race conditions likely to occur.

awareness

Returning to Alison and Brian. After Brian has typed his response he may not know that Alison hasn't seen his second contribution.

Awareness of who is around and what they are doing is a major issue in CSCW (e.g. Dourish [[1992]] and McDaniel [[1997]]). It has various forms:

- being able to tell easily, when you want to know, what other people are doing
- being made aware (via alerts, very salient visual cues etc.) when significant events occur e.g., new user arrives, someone makes a contribution)
- having a peripheral awareness of who is around and what they are up to

Awareness isn't just about other people. In any circumstance where the environment may change, but not through your own direct action, you may need to know what the current state is and what is happening. This is not confined to networked applications, but applies to any hidden or invisible phenomena, for example, background indexing of your hard disk contents. In networked applications, anything distant is invisible unless it is made visible (audible) in the interface.

One of the earlier influential experiments to demonstrate the importance of peripheral awareness was ArKola [[Gaver, 1991]]. This was a simulated bottling factory where two people worked together to maintain the factory, supplying, maintaining etc. the process. The participants couldn't see all of the factory at once, so relied on the sounds produced to be aware of its smooth running or if there are any problems. For example, the sound of breaking glass might suggest that the end of the production line has run out of crates, but if it immediately stopped one would assume that the other participant had sorted the problem out.

The numerous forms of shared video and audio spaces are another example of this – several people, usually in distant offices establish long term, always-on audio, video or audio video links between their offices [[Buxton, 1990; Olson, 1991]]. Sometimes these are used for direct communication, but most of the time just give a peripheral awareness that the other person is purposes (e.g. knowing when the other person is interruptable), but also for social purposes – feeling part of a larger virtual office. Other systems have allowed larger numbers of, usually deliberately low quality and so less intrusive, web-cam views of colleagues' offices and shared areas [[Roussel, 1999, 2001]]. the aim is the same, to build social cohesion, to allow at-a-glance reading of one another's situation, and to promote 'accidental' encounters.

A form of awareness mechanism is now common on the web with buddy-lists that tell you when friends are online [[ICQ]]. Currently I know of no examples of rich media experiments in a domestic environment, for example a virtual kitchen share with your elderly mother in Minnesota (although the Casablanca project at Interval [[Hindus, 2001]] has worked with shared electronic sketch pads in the home). However, the increasing prevalence of continuously connected households and information appliances will surely make this common in the future (see chapter 38).

Trying to capture all this information within a computer display can be distracting, use up valuable screen space, and of course assumes that computer is there and on. For this reason, several projects have looked at **ambient interfaces**, which in various ways make the physical environment reflect the virtual. These interfaces monitor various events in the electronic worlds and then change things in the physical environment: lights on the wall, moving strings hung from the ceiling, even a shaking pot plant [[Lock, 2000]]. Again, this is not fundamentally limited to networked environments, but is of course not very useful when the relevant activity is close at hand anyway.

The other side of this is finding out what people are doing in order to signal this to others. For computer activity – are you logged on, have you been typing recently, what web page are you viewing - this is in principle available, although the various layers of software may make it hard for an awareness service to discover. For non-computer aspects this is more problematic – are you in the room, busy, with other people - and may require of a range of sensors in the environment ultrasound, video etc., with corresponding privacy issues (see for example Bellotti [[1993]]). Monitoring of everyday objects is another way to achieve this, for example, one experiment used electronic coffee cups with sensors to tell when they were picked up and moved around [[Gellersen, 1999]]. As more and more devices become networked it may be that we don't need special sensors, just use the combined information from those available, although the privacy issues remain.

In collaborative virtual reality environments knowing that other people are around (as avatars) is as important as in a physical world, but harder due to limited senses (usually just vision and sound), limited field of view. Furthermore there are computational costs in passing information such as audio or even detailed positional information around the network when there are tens, hundreds or thousands of users. Various **spatial models** have been developed for analyse and implement the idea of proximity in virtual space [[Benford, 1994; Rodden, 1996, Sandor, 1997; Dix, 2000]]. These seek to formalise concepts of: (i) where your focus of attention is within the virtual world and thus whether you require full quality audio and visual representation of others; (ii) broader areas where you would expect some peripheral awareness where potentially degraded information can be used; and (iii) those parts of the space for which you need no awareness information.

16.4.3. media issues

When describing the intrinsic network properties, issues for continuous media were mentioned several times. This is because, with the possible exception of close hand-eye coordination tasks, continuous media put some of the tightest requirements on the underlying networks.

interactive conversation and action

Most demanding of all are audio-visual requirements of interactive conversation. Anyone who has had a trans-continential telephone conversation will have some feeling for the problems a second or two delay can cause. While actually speaking the delays are less significant, however turntaking becomes very problematic. This is because the speaker in a conversation periodically (and subconsciously) leaves short (200-300ms) gaps in the flow of speech. These moments of silence act as entry points for the other participant who is expected to either acknowledge with a 'go on' sound such as "uhm" or perhaps a small nod or the head, or can use to break in with their own conversation. Entries at other points would be seen as butting in and rude, and lack of feedback responses can leave the speaker uncertain as to the listeners understanding. The 200-300ms is again almost certainly related to the time it takes for the listener's sensory system to get the relevant aural information to the brain, and for it to signal the relevant nod, acknowledgement, or start to speak. Clearly our conversational system is finely tuned to the expected intrinsic delays of face-to-face conversation.

When network delays are added it is no longer possible to response within the expected 200-300ms window. The speaker therefore gets no responses at the appropriate points and it is very hard for the listener to break into the flow of speech without appearing rude (by the time they hear the gap and speak the speaker has already restarted). Some telephone systems are **half-duplex**, that is only allow conversation in one direction at a time, this means that the various vocalisations ('uhu', 'hmm' etc.) that give the speaker feedback will be lost entirely while the speaker is actually talking. It is not uncommon for the speaker to have to resort to saying 'are you there' due to a loss of sense of presence.

These effects are similar whether one is dealing with pure audio stream (as with the telephone), video streams (as with desktop conferencing) or distributed virtual environments. One VR project in the UK conducted all its meetings using a virtual environment in which the participants were represented by cuboid robot-like avatars (called blockies) [[Greenhalgh, 1997]]. The project ended with an online virtual party. As the music played the participants (and their avatars) danced. Although clearly enjoying themselves, the video of the party showed an interesting phenomena

Everyone danced alone. There are various reasons for this, for example, it was hard to determine the gender of a potential dancing partner. However, one relates directly to the network delays. Although everyone hears the same music they all were hearing it at slightly different time, furthermore, the avatars for other people will be slightly delayed from their actual movements. Given popular music rhythms operate at several beats per second, even modest delays means that you partner appears to dance completely out of time!

reliability

As well as delays, we noted in section 16.4.1 that network connections may not always be reliable – that is information may be lost. Video and audio streams behave very differently in the presence of dropped information. Imagine you watching a film on a long airflight. The break in the sound when the pilot makes an announcement is much more difficult than loosing sight of the screen for a moment or two as the passenger in front stands up. At a smaller scale a fraction of a second loss of a few frames of video just makes the movie seem a little jerky, a smaller loss of even a few tens of milliseconds of audio signal would make an intrusive click or distortion. In general reliability is more important for audio than video streams and where resources are limited it is typically most important to reserve the quality of service for the audio stream.

sound and vision

Why is it that audio is more sensitive than video? Vision works (largely) by looking at a single snapshot – try walking around the room with your eyes shut, but opening them for glances once or twice a second. Apart from the moment or two as your eyes refocus you can cope remarkably well. Now turn a radio on with the sound turned very low and every second turn the sound up for a moment and back to silent – potentially an interesting remixing sound, but not at all meaningful. Sound more than vision is about change in time. Even the most basic sounds, pure tones, are measured in frequencies – how long between peaks and troughs of air pressure. For more complex sounds the shape of the sound through time – how its volume and frequency mix changes – are critical. For musical instruments it is hard to hear the difference between instruments of they are playing a continuous note, but instantly differentiable by the their **attack** – how the note starts. (To get some idea of the complexity of sound see Mitsopoulos' thesis and web pages [[Mitsopoulos , 2000]] and for an insight into the way different senses affect interaction see my own AVI'96 paper [[Dix, 1996]].)

compression

As we discussed in section 16.4.1, it is possible to produce reliable network connections, but this introduces additional delays. Compression can also help by reducing the overall amount of audio-visual data that needs to be transmitted, but again may introduce additional delays. Furthermore, simple compression algorithms require reliable channels (both kinds of delays). Special algorithms can be designed to cope with dropped data, making sure that the most important parts of the signal are replicated or 'spread out' so that dropped data leads to a loss in quality rather than interruption.

jitter

As noted in section 16.4.1, **jitter** is particularly problematic for continuous media. Small variations in delay can lead to jerky video play back, but is again even worse for audio streams. First of all a longer than normal gap between successive bits of audio data would lead to a gap in the sound just like dropped data. And perhaps even more problematic, what do you do when subsequent data arrives closer together – play it faster? Changing the rate of playing audio data doesn't just make it jerky, but changes the frequency of sound rendering it meaningless.

(*Aside:* Actually there are some quite clever things you can do by digitally speeding up sound but not changing its frequency. These are not useful for dealing with jitter, but can be used to quickly overview audio recordings, or catch up on missed audio streams [[Arons, 1997; Stifelman, 2001]])

For real-time audio streams, such as video-conferencing or voice over the Internet, it is hard to do anything about this and the best one can do is drop late data and do some processing of the audio stream to smooth out the clicks this would otherwise generate.

broadcast and pre-recorded media

Where media is pre-recorded or being broadcast but where a few seconds delay are acceptable it is possible to do far better. Recall that in several places we saw that better quality can be obtained if we are prepared to introduce additional delays.

If you used heard real-audio or real-video broadcasts, you will know that the quality is quite acceptable and doesn't have many of the problems described above. This is partly because of efficient compression meaning that video is compressed to a fraction of a percent of its raw bandwidth and so can fit down even a modem line. However, this would not solve the problems of jitter. To deal with this then player at the receiving end buffers several seconds of audio-visual data before playing it back. The buffering irons out the jitter giving continuous quality.

Try it out for yourself – tune onto a radio channel and simultaneously listen to the same broadcast over the Internet with real-audio. You'll clearly hear up to a minute delay between the two.

16.4.4. public perception: ownership, privacy and trust

One of the barriers to consumer e-commerce has been distrust of the transaction mechanisms – especially giving credit card details over the web. Arguments that web transactions are more secure than phone-based credit cards transactions or even using a credit card at your local restaurant (which gets both card number and signature) did little to alleviate this fear. This was never as major a barrier in the US as it was, for example, in Europe, but across the world has been a concern, slowing down the growth of e-

It certainly is the case that transactions via secure channels can be far more secure than physical transactions, where various documents can be stolen or copied en-route and are in a format much more easy to exploit for fraud. However, knowing that a transaction is secure is more than the mechanisms involved, it is about human trust. Do I understand the mechanisms well enough, and the people involved well enough, to trust my money to it?

In fact, with a wider perspective this distrust is very well founded. Encryption and authentication mechanisms can ensure that I am talking to a particular person or company and that no-one else can overhear. But how do I know to trust that person? Being distant means I have few of the normal means available to assess the trustworthiness of my virtual contact. In the real world I may use the location and appearance of a shop to decide whether I believe it will give good service. For mail order good I may use the size, glossiness and publication containing an advert to assess its expense and again use this to give a sense of quality of the organisation. It is not that these physical indicators are foolproof, but that we are more familiar with them. In contrast virtual space offers few intrinsic **affordances**. It is easy and quite cheap to produce a very professional web presence, but it may be little more than facade. This is problematic in all kinds of electronic materials, but perhaps most obvious when money is involved.

Even if I trust the person at the other end, how do I know whether the network channel I am using is of a secure kind? Again the affordances of the physical world are clear: in a closed office vs. in the open street vs. in a bar frequented by staff of a rival firm. We will say different things depending on the perceived privacy of the location. In the electronic world we rely on "https" at the beginning of a URL (how many ordinary consumers know what that means?), or an icon inserted by the email program to say a message has been encrypted or signed. We need to trust not only the mechanisms themselves, but also the indicators that tell us what mechanisms are being used [[Millett, 2001; Fogg, 2001]].

In non-financial transactions issues of privacy are also critical. We've already seen several examples where privacy issues occur. As more devices become networked, especially via wireless links and our environment and even our own bodies become filled with interlinked sensors, issues about who can access information about you become more significant. This poses problems at a technical level – ensuring security between devices, at an interface level – being able to know and control what or who can see specific information, and at a perception level – believing in the privacy and security of the systems. there are also legal implications. For example, in the UK currently (2001) it is illegal for mobile telecomms operators to give location information to third parties [[Sangha, 2001]].

The issue of perception is not just a minor point, but perhaps the dominant one. Networks, and indeed computer systems in general, are by their nature hidden. We do not see the bits travelling down the wires or through the air from device to device, but have to trust the system at even the most basic level. As HCI specialists we believe ourselves a little above the mundane software engineers who merely construct computer systems, as we take a wider view and understand that the interaction between human and electronic systems has additional emergent properties and that it is this complete socio-technical unit which achieves real goals. For networked systems this view is still far too parochial.

Imagine if the personal email of millions of people was being sucked into the databanks of a transnational computer company, and only being released when accessed through the multinational's own web interface. The public outcry! Imagine hotmail, yahoo mail etc. How is it that although stored on distant computers, perhaps half the world away, millions of people feel that it is 'their' mailbox and trust the privacy of web mail more than perhaps their organisation's own mail system. This feeling of ownership of remote resources is more than the technology that protects the security of such systems, it is a cultural phenomena and a marketing phenomena. The web mail 'product' is not just technology, or interface, but formed by every word that is written about the product in adverts, press releases and media interviews. [[Dix, 2001]]

16.5. Networks as Subjects understanding and managing networks

When using as networked application you don't really care what kind of network is being used, whether the data is sent over copper wires, fibre optic, microwave or satellite. All you care is that the two ends manage to communicate and the effects any of the above have on the end-to-end network properties such as bandwidth discussed in the previous section. However, there are times when the network's internals can't be ignored.

Those involved in installing or managing networks need to understand the internal workings of the network in order to optimise performance and find faults. For ordinary users, when things go wrong in a networked application, they effectively become a network manager and so understanding something of the network can help them to deal with the problem. Even when things are working having some awareness of the current state of the network may help one predict potential delays, avoid problems and minimise costs.

We'll start this section by looking at some of the technical issues that are important in understanding networks. This parallels section 16.4.2, but is focused on the internal properties of the network. We'll then look ate the interface issues for those managing networks and the ways in which interfaces can make users aware of critical network state. Finally, we'll look briefly at the way models of networks can be used as a metaphor for some of the motor and cognitive behaviours of humans.

16.5.1. network models

layers

Networking is dominated by the idea of layers – lower levels of the network offer standard interfaces to higher levels so that it its possible to change the details of the lower level without changing the higher levels. For example, imagine you are using your PDA to access the Internet whilst in a train. The web browser establishes a TCP/IP connection to the web server, requests a web page and then displays it. However, between your PDA and the web server the message may have travelled through an infra-red link to your mobile phone, which then used a cell-based radio to send it to a mobile-phone station, then via a micro-wave link to a larger base station onto a fibre optic telephone backbone and via various copper wires to your ISP's modem bank. Your ISP is then connected into another fibre optic Internet backbone and eventually via more fibre optic, copper and microwave links to the web server. To complicate things even further it may even be that the telephone and Internet backbones may share the same physical cabling at various point. Imagine if your poor PDA had to know about all of this.

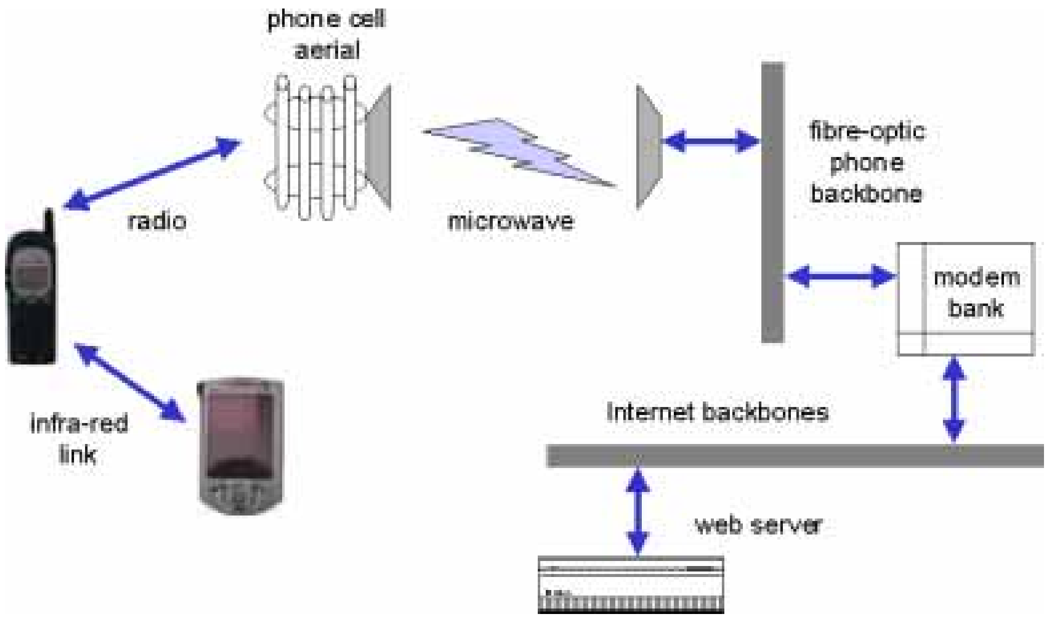


Figure 16.6. the long path from PDA to web

In fact, even your PDA it will know about at least 5 layers:

- Infra-Red – how the PDA talks to the phone
- modem – how the PDA uses the phone to talk to your ISP
- IP – how your PDA talks to the web-server computer
- TCP – how data is passed as a reliable connection-based channel between the right program on your PDA and the web-server computer
- HTTP – how the browser talks to the web server

Each of these hides most of the lower levels, so your browser needs to know nothing about IP, the modem or the infra-red connection whilst accessing the web page.

The nature of these layers differs both between different types of network, for example WAP, for sending data over mobile phones and devices, has 5 defined layers [[Arehart, 2000]], the ISO OSI reference model has 7 layers [[ISO/IEC7498, 1994]].

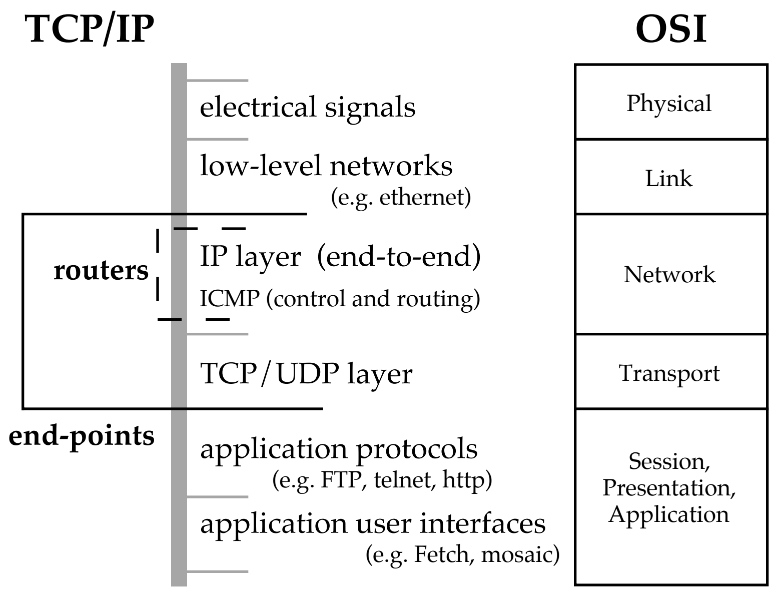


Figure 16.7 OSI 7 layers and TCP/IP

protocols

Systems at the same layer typically require some standard language to communicate. This is called a **protocol**. for higher levels this may be quite readable. For example, to send an Internet email message your mail program connects to an SMTP server (a system that relays messages) using TCP/IP and has the exchange shown in figure 16.8.

mail program says **SMTP server replies**

```
                      220 mail.server.net ESMTP
HELO mypc.mydomain.com
                      250 mail.server.net Hello
                      mypc.mydomain.com[100.0.1.7], pleased to meet you
MAIL From:<myself@mydomain.com>
                      250 <myself@mydomain.com>... Sender ok
RCPT To: <a-friend@theirdomain.com>
                      250 <a-friend@theirdomain.com>... Recipient ok
DATA
                      354 Enter mail, end with "." on a line by itself

first line of message
..dotty 2nd line
last line
.
                      250 KAA24082 Message accepted for delivery
QUIT
                      221 mail.server.net closing connection
```

Figure 16.8 protocol to send email via SMTP

At lower levels data is usually sent in small **packets**. which contain a small amount of data plus header information saying where the data is coming from, where it is going to and other bookkeeping information such as sequence numbers, data length etc.

header				body
to address	from address	info	data length	data .....

Figure 16.9 typical network packet format (simplified)

Even telephone conversations, except those using pre-digital exchanges, are sent by chopping up your speech into short segments at your local telephone exchange, sending each segment as a packet and then reassembling the packets back into a continuous stream at the other end [[Stevens, 1998, 1999]].

internetworking and tunnelling

This layering doesn't just operate within a particular network standard, but between different kinds of networks too. The Internet is an example of an internet (notice little 'i') that is a network which links together different kinds of low level network. For example, many PCs are connected to the Internet via an ethernet cable. Ethernet sends its own data in packets like those of figure 16.9. The Internet protocol (IP) also has packets of a form like figure 16.9. When you make an Internet connection via ethernet the IP packets are placed in the data portion of the ethernet packet, so you get something a bit like figure 16.10.

ethernet header			ethernet body			
			IP header			IP body
ethernet to addr.	ethernet from addr.	IP flag data len. etc.	IP to addr	IP from addr	other header IP info	Internet data .....

Figure 16.10 Internet IP packet inside ethernet packet

This placing of one kind of network packet inside the data portion of another kind of network is also used in virtual private networks (VPNs) in a process called **tunnelling**.. These are used to allow a secure network to be implemented using a public network like the Internet. Imagine a company has just two offices, one in Australia and the other in Canada. When a computer in the Sydney office sends data to a computer in the Toronto office, the network packet is encrypted, put in the data portion of an Internet packet and sent via the Internet to a special computer in the Toronto office. When it gets there, the computer at the Toronto office detects it is VPN data, extracts the encrypted data packet, decrypts it and puts it onto its own local network where the target computer picks it up. As far as both ends are concerned it looks as if both offices are on the same LAN and any data on the Internet is fully encrypted and secure.

routing

If two computers are on the same piece of physical network each can simply, listen out for packets that are destined for them, so sending messages between them is easy. If however messages need to be sent between distant accessing a web server in the US, the message cannot simply be broadcast to every machine on the Internet (figure 16.11). Instead at each stage it needs to be passed in the right direction between different parts of the network. Routers perform this task. They look at the address of each packet and decide where to pass it to. If it is a local machine this might be to simply put it onto the relevant local network, but if not it may need to pass it on to another intermediate machine.

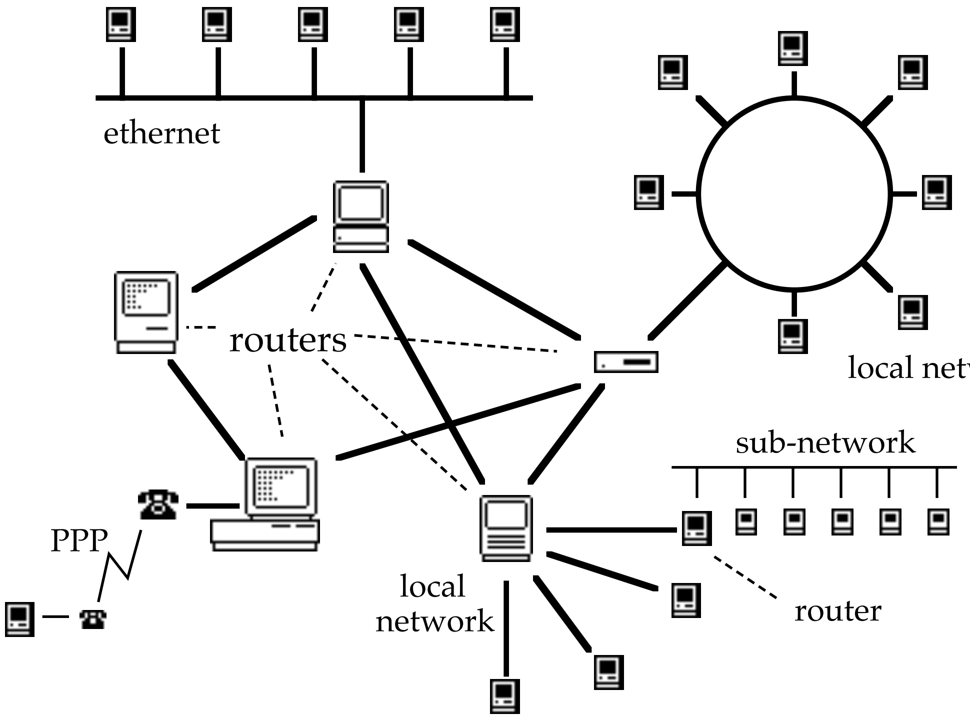


Figure 16.11 routers send messages in the right direction through complex networks

Routers may be stand-alone boxes in network centres, or may be a normal computer. Often a file server acts as a router between a LAN and the global network.

As well as routers networks are also linked by hubs and switches, that make several different pieces of physical network behave as if they were one local network; and gateways, that link different kinds of network. The details of these are not important, but they add more to the shear complexity of even small networks.

addresses

In order to send messages on the Internet or any other network you need to have the address of where they are to go (or at least your computer needs it). In a phone network this is the telephone number and on the Internet it is an **IP number**. The IP number is a 32 bit number, normally represented as a group of 4 numbers between 0 and 255 (e.g. 212.35.74.132), which you will have probably seen at some stage when using a web browser or other Internet tool. It is these IP numbers that have are used by routers to send Internet data to the right place.

However, with any network there is a problem of how you get to know the address. With phone numbers you simply look up the persons name in a telephone directory or by phoning the operator. Similarly, most networks have a naming scheme and some way to translate these into addresses. In the case of the Internet, domain names (e.g. acm.org, www.hcibook.com. magisoft.co.uk) are the naming system. There are so many of these and they are changed relatively rapidly, so there is no equivalent of a telephone directory, but instead special computers called **domain name servers** (DNS) act as the equivalent of telephone directory enquiries operators. Every time an application needs to access a network resource using a domain name (e.g. to lookup a URL) the computer has to ask a DNS what IP address corresponds to that domain name. Only then can it use the IP address to contact the target computer.

The 32 bit IP number space allows for 4 billion addresses. This sounds like quite a lot, However, these have been running out due to the explosive growth in the number of Internet devices and 'wasted' IP numbers due to the way ranges of numbers get allocated to sub-networks. The new version of TCP/IP, IPv6, which is beginning to be deployed, will have 128 bit addresses which will require 16 numbers [[IPng, 2001]]. This will allow unique IP numbers for every phone, PDA, Internet enabled domestic appliance or even electronic paper-clip.

The DNS system is an example of a '**white pages**' system. You have an exact name and want to find the address for that name. In addition there are so called '**yellow pages**' systems where you request, for example, a colour postscript printer and are told of addresses of systems supplying the service. Sometimes this may be mediated by brokers which may attempt to find the closest matching resource (e.g. a non-postscript colour printer) or even perform translations (for example, the Java JINI framework [[Edwards, 2001]]).

This latter form of resource discovery system is most important in mobile systems and ubiquitous computing where we are particularly interested in establishing connections with other geographically close devices.

A final piece in the puzzle is how one gets to know the address of the name server, directory service or brokering service. In some types of network this may be managed by sending broadcast requests to a network "is there a name server out there". In the case of the Internet this is normally explicitly set for each machine as part of its network settings.

all together ...

If you are in your web browser and you try to access the URL http://www.meandeviation.com/qbb/ the following stages happen:

1. send IP-level request to DNS asking for www.meandeviation.com
2. wait for reply
3. DNS sends reply 64.39.13.108
4. establish TCP level connection with 64.39.13.108
5. send HTTP request ("GET /qbb/ HTTP/1.1")

7. close TCP connection

Most of these stages are themselves simplified, all will involve layering on top of lower level networks and most stages involve several substages. (e.g. establishing a TCP level connection requires several IP-level messages)

The basic message is that network internals are multi-level, multi-stage and pretty complicated.

16.5.2. network management

Those most obviously exposed to this complexity and the engineers managing large national and international networks, both data networks, such as the main Internet backbones, and telecoms networks. The technical issues outlined above are compounded by the fact that the different levels of network hardware and network management software are typically supplied by different manufacturers. Furthermore, parts of the network may be owned and managed by a third party and shared with other networks. For example, a trans-Atlantic fibre optic cable may carry telecoms and data traffic from many different carriers.

When a fault occurs in such a network it is hard to know whether it is a software fault or a hardware fault, where it is happening, who is responsible. If you send engineers out to the wrong location it will cost both their time and also increase the time the service is unavailable. Typically the penalties for inoperative or reduced quality services are high – you need to get it right fast.

This is a specialised and complex area, but clearly of increasing importance. It posed many fascinating UI challenges: how to visualise complex multi-layered structures, and how to help operators trace faults in these. Although I know that it is a topic being addressed by individual telecoms companies, published material in the HCI literature is minimal. The exception is visualisation of networks, both physical and logical, which is quite extensive (Dodge and Kitchin's Atlas of Cyberspace and associated web site [[Dodge, 2001]] is the comprehensive text in this area).

Ordinary systems administrators in organisations face similar problems albeit on a smaller scale. From near universal experience of misconfigured email systems, continual network failures and performance problems certainly suggests this is an area ripe for effective interface solutions, but again there is very little in the current HCI literature.

Finally, it appears that everyone is now a network administrator, even a first time home PC user must manage modem settings, name server addresses, SMTP and POP servers, and more. It is interesting that for most such users the interface they use is identical to that supplied for full systems administrators. Arguably this may ease the path for those who graduate from single machines to administering an office or organisation, perhaps less than 5% of users. Unfortunately, it makes life intolerable for the other 95%! The only thing that makes this possible at all is that the 'welcome' disks for many ISPs offer step-by step instructions or may automatically configure the system. These complications are compounded if the user wishes to allow access through more than one ISP, or connect into a fixed network. As many home users now have several PCs and other devices that need to be networked this is not a minor issue.

If we look at the current state of the two most popular PC systems, Microsoft Windows and Mac OS, the picture is not rosy.

On Windows at least most of the settings for an ISP are concentrated within one configuration file and so dialling into an ISP means selecting the right file and double clicking. Unfortunately, swopping between fixed networks is far more complicated involving recording half a dozen IP numbers and other settings and changing these in different control panels every time one changes networked. For those on portables moving between different sites this is very difficult.

On MacOS settings up a new ISP involves at least three control panels (a side effect of organising the control panels to correspond to 'logical' network layers). On the positive side the MacOS Location Manager uses the same interface for switching between ISPs or between different fixed networks and both are equally easy (once configured). Unfortunately, the interface model for editing configurations for a particular location is perhaps one of the most incomprehensible I've encountered, involving a confusion between the location being edited and current location. Given the slickness of most other aspects of MacOS and the ease of networking Macs with AppleTalk, this appears to indicate a lack of perceived importance somewhere within the design process, made more surprising my the emphasis places on Internet readiness in the marketing of iMacs and iBooks.

MacOS Location Manager

to edit a location:

- **select it as the 'edit location' (different from the 'current location')**
- **select the configuration item you want to change in the edit interface**
- **change and conform the corresponding control panel for the current location**
- **confirm the change in the edit interface**
- **remember to change everything back for the current configuration!**

I would hope that both these Windows and MacOS interfaces will improve over the lifetime of this book. The problems in both emphasise partly the intrinsic complexity of networking – yes it does involve multiple logically distinct settings, many of which relating to low-level details. However, it also exposes the apparent view that those involved in network administration are experts who understand the meaning of various internal networking terms. This is not the case even for most office networks and certainly not at home.

And this is just initially setting up the system to use. For the home user debugging faults has many of the same problems as large networks. You try to visit a web site and get an error box ... Is the web site down? Are there problems with the phone line, the modem, the ISP's hardware? Are all your configurations settings right? Has a thunderstorm 3000 miles away knocked out a vital network connection? Trying to understand a multi-layered, non-localised and, when things work, largely hidden system is intrinsically difficult and where diagnostic tools for this are provided, they assume an even greater degree of expertise.

The much heralded promise of devices that connect up to one another within our homes and about our bodies is going to throw up many of the same problems. Some old ones may ease as explicit configuration becomes automated by self-discovery between components, but this adds further to the hiddenness and thus difficulty in managing faults, security etc. You can imagine the scenario – the sound on my portable DVD stops working and produces a continuous noise – why? There are no cables to check of course (wireless networking), but hours of checking and randomly turning devices on and off narrows the problem down to a fault in the washing machine which is sending continuous "I finished the clothes" alerts to all devices in the vicinity.

16.5.3. network awareness

One of the problems noted above is the hiddenness of networks. This causes problems when things go wrong as one hasn't an appropriate model of what is going on, but also sometimes even when things are working OK.

We discussed in section 16.4.1 some of the network properties that may affect usability – bandwidth, delay, jitter etc. These are all affected to some extent by other network load, the quality of current network connections etc. So predicting performance (and knowing whether or not to panic if things appear to go slow) needs some awareness of the current state of the network.

PCs using wireless networks usually offer some indication of signal strength (if one knows where to look and what it means) although this is less common for line quality for modems. As wireless devices and sensors become smaller they will not have suitable displays for this and explicitly making users aware of the low-level signal strength of an intelligent paper clip may not be appropriate. However, as interface designers we do need to think how users will be able to cope and problem solve in such networks.

Only more sophisticated network management software allows one to probe the current load on a network. This does matter. Consider a small home network with several PCs connected through a single modem. If one person starts a large download they will be aware that this will affect the performance of the rest of their web browsing. Other members of the household will just experience a slowing down of everything and not understand why. In experiments at MIT the level of network traffic has been used to 'jiggle' a string hanging from the ceiling so that heavy traffic leads to a lot of movement [[Wisneski, 1998]]. The movements are not intrusive so give a general background awareness of network activity. Although supplying a ceiling mounted string may not be the ideal solution for every home, other more prosaic interface features are possible.

Cost awareness is also very important. In the UK current generation mobile data services are charged by connection time. So knowing how long you have been connected and how long things are likely to take becomes critical. If these charges differ at peak hours of the day, calculating whether to read your email now, or do a quick check and download the big attachments later can become a complex decision. With moves to data-volume-based charging the calculation will be replaced by some attempt to estimate the volume of data that is likely to be involved in initiating and action versus the value of that data – do you want to click on that link if the page it links to includes large graphics, perhaps an applet or two?

network confusion

If the preceding doesn't sound confusing enough, the multi-layered nature of networked applications mean that it is hard to predict the possible patterns of interference between things implemented at different levels or even at the same level. Again this is often most obvious when things go wrong, but also because unforeseen interactions may mean that two features that work perfectly well in isolation may fail when used together.

These problem, **feature interaction**, has been studied particularly in standard telecoms (although certainly not confined to it). Let's look at an example of feature interaction. Telephone systems universally apply the principle that the caller, who has control over whether and when the call is made, is the person who pays. In the exceptions (free-phone numbers, reverse charges) special efforts are made to ensure that subscribers understand the costs involved. Some telephone systems also have a feature whereby a caller who encounters a busy line can request a call-back when the line becomes free. Unfortunately at least one company implemented this call-back feature so that the charging system saw the call-back as originating from the person who had originally been called. Each feature seemed to be clear on its own, but together meant you could be charged for calls you didn't want to make.

With N features there are N(N-1)/2 possible pairs of interactions to consider, N(N-1)(N-2)/6 triples etc. This is a well recognised (but not solved) problem with considerable efforts being made using, for example, formal analysis of interactions to automatically detect potential problems. It is worth noting that this is not simply a technical issue. the charging example shows that it is not just who pays that matter, but the perceptions of who pays. This particular interaction would have been less of a problem if the interface of the phone system had, for example, said (in generated speech) "you have had a call from XXX" press call back, you will be charged for this call". The hybrid feature interaction research group at Glasgow are attempting to build a comprehensive list of such problems in telecoms [[HFIG, 2001]], but these sorts of problems are likely to be found increasingly in related areas such as ubiquitous computing and resource discovery.

16.5.4. the network within

So far the story is pretty bleak from a user interface viewpoint – a complex problem, of rapidly growing importance, with relatively little published work in many areas. One good thing as a HCI practitioner about understanding the complexity of networks, is that they help us understand better the workings of the human cognitive and motor system.

For at least five decades computational models have been used to inspire cognitive. Also of course cognitive and neurological models have been used to inspire computational models in artificial intelligence and neural networks. However, our bodies are not like a single computer, but in various ways more like a networked system.

First because several things can happen at once. The interacting cognitive subsystems (ICS) model from APU Cambridge [[Barnard, 1995] takes this into account looking at various parts of the cognitive system, the conversions between representations between these parts and the conflicts that arise if the same part is used to perform different tasks simultaneously. Similarly, the very successful PERT-style GOMS analysis used on the NYNEX telephone operators interface used the fact that the operator could be doing several things simultaneously with no interfering parts of their bodies and brains [[John, 1990; Gray, 1992]].

We are also like a networked system in that signals take an appreciable time to get from our senses to our brains and from our brains to our muscles. The famous homunculus from Card, Moran and Newell's Model Human processor makes this very clear with timings attached to various paths and types of mental processing [[Card, 1983]]. In fact, the sorts of delays within our bodies (from 50-200ms on different paths) are very similar to those found on international networks.

In industrial control one distinguishes between open-loop and closed-loop control systems. Open-loop control is where you give the machine an instruction and assume it does it correctly (like a treasure map – "ten steps North, turn left, three steps forward and dig"). This assumes the machine is well calibrated and predictable. In contrast closed-loop control uses sensors to constantly feedback and modify future actions based on the outcomes of previous ones (e.g. ""follow the yellow brick road until you come to the emerald city"). Closed-loop control systems tend to be far more robust, especially in uncontrolled environments, like the real world.

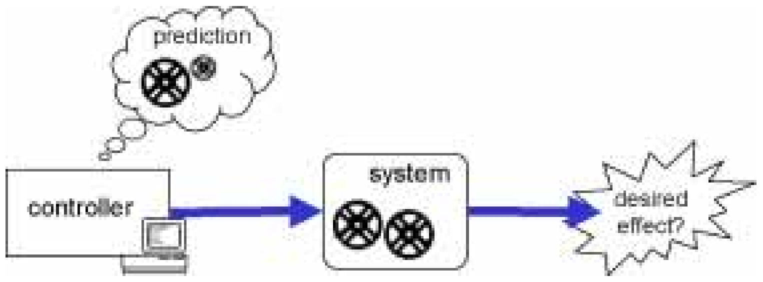


Figure 16.12 (i) open-loop control

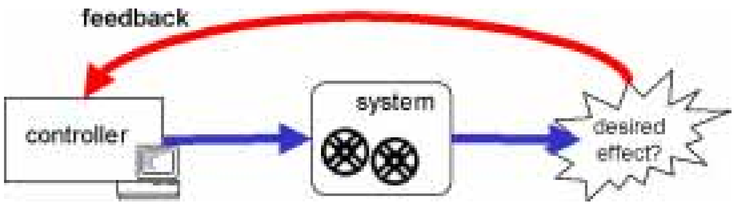


Figure 16.12 (ii) closed-loop control

Not surprisingly our bodies are full of closed-loop control systems (e.g. the level of carbon dioxide in your lungs triggers the breathing reflex). However closed-loop control can become difficult if there are delays – you haven't got feedback from the previous action when starting the next one. Delays either mean one has to slow down the task or use some level of prediction to work out what to do next based on feedback of actions before the last one. This breakdown of closed-loop control in the face of (especially unexpected) delays is one of the reasons hand-eye coordination tasks, such as mouse movement, breakdown if delays exceed a couple of hundred milliseconds (section 16.4.2). The feedback loops in our bodies for these tasks assume normal delays of around 200ms and are robust to variations around this figure, but adding delays beyond this start to cause breakdown.

The delays inside our bodies cause other problems too. The path from our visual cortex into our brain is far faster (by a 100ms or so) than that from our touch and muscle tension sensors around our bodies. If we were designing a computer system to use this information, we might consider having a short 100ms tape loop, so that we could store the video input until we had the appropriate information from all senses. However, the shear volume of visual information means that our brains do not attempt to do this. Instead there is a part of our brains that predicts where it 'thinks' our bodies are and what it is feeling based on previous nerve feedback and what it knows the muscles have been asked to do. The same bit of the brain then monitors what actually did happen (when the nerve signals have made their way up the spinal column to the brain) and gives an uncomfortable, or shocked sensation when a mismatch occurs. For example, if you go to pick something up, but because of poor light or a strange shaped object you touch it earlier or later than you would expect. Tickling is also connected with this lack of ability to predict the sensations (why it is difficult to tickle yourself).

Race conditions also occur within this networked system of our bodies – for example, getting letter inversions whilst typing where signals to the two hands get processed in the wrong order. Steve Brewster and I also used race conditions to understand what goes wrong in certain kinds of mis-hits of on-screen buttons [[Dix, 1994]]. In certain circumstances two almost simultaneous 'commands' from our brain to our hand to release the mouse button and to our arm to move to a new mouse location can get out of order meaning the mouse moves out of the target before it is released. this analysis allowed us to design an experiment that forced this very infrequent error to occur much more frequently and therefore make it easier to assess potential solutions.

16.6. Networks as Platforms
algorithms and architectures for distributed interfaces

User interfaces are hard enough to construct on a single machine, concurrent access by users on networked machines is a nightmare!

Happily, appropriate algorithms, architectures, toolkits and frameworks can help ... a bit.

16.6.1. accessing shared objects

We saw in section 16.4.2 how race conditions within networked systems can lead to inconsistencies within the user interface and within the underlying data structures. Happily there are a range of techniques for dealing with this.

locking

The standard technique, used in databases an file systems, for dealing with multiple accesses to the same data object is locking. When a user's application wants to update a particular database record, it asks the database manager for a lock on the record. It can then get a copy of the record and send back the update knowing that nothing else can access it in the mean time. Users are typically unaware that locking is being performed, the act of opening a file or opening a edit form for a database record establishes the lock and closing the file or the edit form releases the lock.

Although this is acceptable for more structured domains there are problems in more dynamic domains such as shared editing. Locking a file when one user is editing it is no good as we want several people to edit the same file at the same time. In these cases more lightweight forms of locking can be used at finer granularities: at paragraph, sentence or even per character level. for example, the act of clicking over a paragraph to set a text entry position may implicitly request a paragraph lock, which is released when you go on to edit another paragraph. However, implicit and informal locks, because they are not apparent, can lead to new problems. For example, a user may click on a paragraph, do some changes, but before moving on to another part of the document get interrupted. No-one else can then edit the paragraph. To avoid this, the more informal locks are often time limited or can be forcibly broken by the server if another user request a lock on the same object.

replication

In systems such as Lotus Notes, users do not lock central copies of data, but instead each user (or possibly each site) has their own replica of the complete Notes database. Periodically theses replicas are synchronised with central copies or with each other. Updates can happen anywhere by anyone with a replica. Conflicts may of course arise if two people edit the same note between synchronisations. Instead of preventing such conflicts the system (and software written using it) accepts that such conflicts will occur. When the replicas synchronise conflicts are detected and various (configurable) actions may occur: flagging the conflicts to users adding conflicting copies as versions etc.

This view of replicate and worry later is essential in many mobile applications as attempts to lock a file whilst disconnected would first of all require waiting until a network connection could be made, and, worse, if the network connection is lost whilst the lock is still in operation could lead to files being locked for very long periods. Other examples of replication in research environments include the CODA at CMU, which allows replication of a standard UNIX file system[[Kistler, 1992]] and Liveware, a contact information system that replicates and synchronises when people meet in manner modelled after the spread of computer viruses [[Witten, 1991]]. Although Liveware is now quite a few years old and was 'spread' using synchronising floppy disks, the same principle is now being suggested for PDAs and other devices to exchange information via IrDA or bluetooth.

optimistic concurrency for synchronous editing

The "do it now and see if there are conflicts later" approach is also called **optimistic concurrency**, especially in more synchronous settings. For example, in a shared editing system the likelihood of two users editing the same sentence at the same time is very low. An optimistic algorithm doesn't bother to lock or otherwise check things when the users start to edit in an area, but in the midst or at the end of their edits checks to see if there are any conflicts and attempts to fix them.

There are three main types of data that may be shared:

orthogonal data – where the data consists of attributes of individual objects/records that can all be independently updated

sequential data – particularly text, but any form of list where the order is not determined by an attribute property

complex structural data – such as directory trees, taxonomic categories etc.

In terms of complexity for shared data these are in increasing difficulty.

Orthogonal data, although by no means trivial, is the simplest case. There is quite a literature on shared graphical editors, which all have this model – independent shapes and objects with independent attributes such as colour, size, position. When merging updates from two users all one has to do is look at each attribute in turn and see whether there it has been changed by only one user, in which case the updated value is used, or if it has been changed by both, in which case either the last update is used or the conflict is flagged.

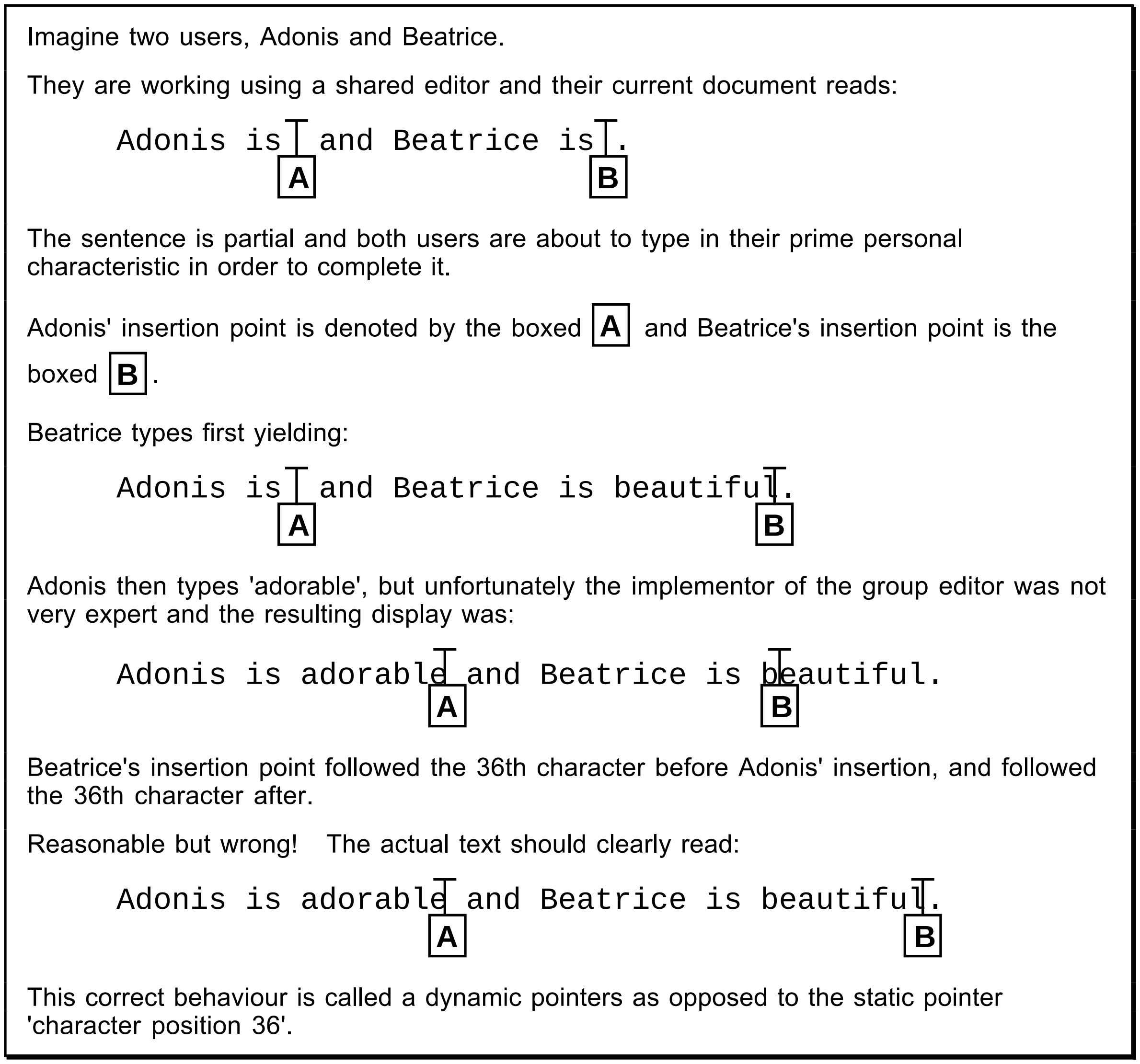
Structured data is most complicated – what do you do if someone has created a new file in directory D, but at the same time someone else has deleted the directory? I know of know optimistic algorithms for dealing with this in the CSCW literature. CODA deals with directory structures (normal UNIX file system) but takes a very simple view of this as it only flags inconsistencies and doesn't attempt to fix them.

Algorithms for shared text editing sit somewhere between the two and have two slightly different problems, both relating race conditions when two or more users are updating the same text:

dynamic pointers – if user A is updating an area of text in front of user B, then the text user B's is editing will effectively move in the document.

deep conflict – what happens if user A and user B's cursors are at the very same location and they perform insertions/deletions

Figure 16.13 shows an example of the first of these problems. The deeper conflict occurs when both cursors are at the same point, say after the 'Y' in "XYZ". Adonis types 'A' and at the same time Beatrice types 'B' should we have "XYABZ" or "XYBAZ" or perhaps even loose one or other character? Or if Adonis types 'A' and Beatrice presses 'delete' should we have "XYZ" or "XAZ"?



Imagine two users, Adonis and Beatrice.

They are working using a shared editor and their current document reads:

```
Adonis is and Beatrice is.
         A                B
```

The sentence is partial and both users are about to type in their prime personal characteristic in order to complete it.

Adonis' insertion point is denoted by the boxed $\boxed{A}$ and Beatrice's insertion point is the boxed $\boxed{B}$.

Beatrice types first yielding:

```
Adonis is and Beatrice is beautiful.
         A                         B
```

Adonis then types 'adorable', but unfortunately the implementor of the group editor was not very expert and the resulting display was:

```
Adonis is adorable and Beatrice is beautiful.
                 A                 B
```

Beatrice's insertion point followed the 36th character before Adonis' insertion, and followed the 36th character after.

Reasonable but wrong! The actual text should clearly read:

```
Adonis is adorable and Beatrice is beautiful.
                 A                          B
```

This correct behaviour is called a dynamic pointers as opposed to the static pointer 'character position 36'.

Figure 16.13 dynamic pointers from [[Dix, 1995]]

A number of algorithms exist for dealing with this [[Sun, 1998; Mauve, 2000; Vidot, 2000]] most stemming from the dOPT algorithm of used in the Grove transformations that allow you to reorder operations. For example, if we have two insertions (labelled a and b) performed at the same time at different locations:

a) insert texta at location n

b) insert textb at location m

but decide to give insert a preference, then we have to transform b to b' as follows:

b') if (m < n) insert textb at location m (i)

if (m = n) insert textb at location m (ii)

if (m > n) insert textb at location m+length(texta) (iii)

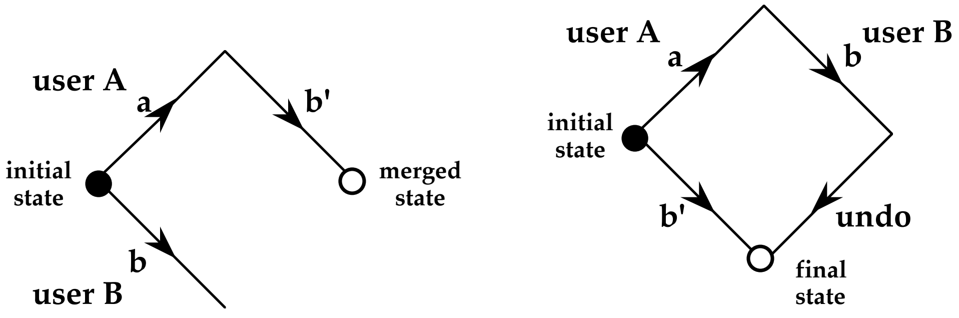
Case (i) says that if the location of insert b is before insert a you don;t have to worry. Case (iii) says that if it is after insert a you have to shift your insert along accordingly. Case (ii) is the difficult one where a conflict occurs and has to be dealt with carefully to ensure that the algorithm generates the same results no matter where it is. The version above would mean that B's cursor gets left behind by A's edit. The alternative would be to make case (ii) the same as case (iii) which would mean B's cursor would be being pushed ahead of A's.

In early work in this area I proposed making dynamic pointers first class objects and using these in all representations of actions [[Dix, 1991, 1995]]. This means that rules like case (i) and case (iii) happen 'for free', but the deep conflict case still needs to be dealt with specially.

groupware undo

The reason that undo is complicated in groupware is similar to the problems of race conditions in optimistic concurrency.

In the case of optimistic concurrency user A has performed action a and user B has performed action b, both on the same initial state. the problem is to transform user B's action into one b' that can be applied to the state after action a yet still mean the 'same things' as the original action b.



optimistic concurrency group undo

Figure 16.14 multi-user transformations

In the case of groupware undo we may have the situation where user A has performed action a, followed by user B performing action b, then user decides to undo action a. How do we transform action b so that the transformed b' means the same before action a as b did after.

Similar, but slightly different transformation rules can be produced for the case of undo and also dynamic pointers can be used for most cases.

As with optimistic concurrency there is slightly more work on group undo in shared graphical editors where the orthogonal data makes conflicts easier. [[Berlage, 1992]]

real solutions?

Although these various algorithms can ensure there is no internal inconsistency and that all participants see the same thing, they do not necessarily solve all problems. Look again at the case of Alison and Brian's chat in section 16.4.2. Certainly in the case of group undo, when Gregory Abowd and I published the first paper on the topic [[Abowd, 1992]], we proposed various solutions, but also recommended that as well as an explicit undo button systems ought to provide sufficient history to allow users to recreate what they want to without using the undo button. This is not because it is impossible to find a reasonable meaning for the undo button, but because in the case of group undo, there are several reasonable meanings. Choosing the meaning a user intends is impossible and so it may sometimes be better not to guess.

16.6.3. architectures for networked systems

Software architecture is about choosing what (in terns of code and functionality) goes where. For applications on a single machine these are often logical distinctions between parts of the code. For networked systems then 'where' includes physical location and choice of location makes and enormous difference to the responsiveness of the system.

The simplest systems are almost always **centralised client–server** architectures where the majority of computation and data storage happens in the central server. many of the problems of race conditions and potential inconsistencies disappear. However, this means that every interaction requires a network interaction with the server meaning that feedback may be very slow. The opposite extreme are **replicated peer–peer** architectures where all the code is running on users' own PCs and the PCs communicate directly with one another. Feedback can now be instantaneous, but the complexity of algorithms to maintain consistency, catch-up late joiners etc. can be very complex. Most system operate somewhere between these two extremes with a central 'golden copy' of shared data, but with some portion of the data on individual PCs, PDAs or other devices in order to allow rapid feedback.

In web applications options are constrained by the features allowed in HTML and web browsers. Note that even a web form is allowing some local interaction (filling in the form) as well as some centralised interaction (submitting it). **Applets** allow more interaction, but the security limitations mean that they can only talk back to the server where they originated. Thus by chat servers and similar programs that relay messages between clients. For Intranets it is easier to configure browsers so that they accept applets as trusted and thus with greater network privileges, or to include special plug-in components to perform more complicated actions.

Mobile systems have yet more issues as they need to be capable of managing disconnected operation (when they have no physical or wireless connection tot he network). This certainly means keeping cached copies of central data. For example, the AvantGo browser for PDAs downloads copies of web pages onto a PDA and synchronises these with their latest versions every time the PDA is docked into a desktop PC [[AvantGo, 2001]]. Although this is now beginning to change with higher bandwidth mobile data services, it is still the case that access whilst mobile is slower and more expensive than whilst using a fixed connection. This pushes one towards replicated architectures with major resynchronisation when connected via cheap/fast connections and more on-demand or priority driven synchronisation when on the move.

16.6.4. supporting infrastructure

In order to help manage networked applications various types of supporting infrastructure are being developed.

awareness servers

These keep track of which users are accessing particular resources (e.g. visiting a particular web page) so that you can be kept informed as to whether others are 'near' you in virtual space, or whether friends are online and active. [[Palfreyman, 1996; ICQ; Sun, 2001]]

notification servers

These serve a similar role for data allowing client programs to register an interest in particular pieces of shared data. When the data is modified or accessed the interested parties are 'notified'. For example, you may be told when a web page has changed or when a new items has been added to a bulletin board. Some notification servers also manage the shared data [[Patterson, 1996]] whilst others are 'pure' just managing the job of notification [[Ramduny, 1998]].

event/messaging systems

These allow different objects in a networked environment to send messages to one another in ways which are more convenient than that allowed by the raw network. For example, they may allow messages to be sent to objects based on location independent names, so that objects don't have to know where each other are.

resource discovery

Systems such as the Java JINI framework and Universal Plug-and-Play allow devices to find out about other devices close to them, for example, the local printer, and configure themselves to work with one another. As ubiquitous and mobile devices multiply this will become increasingly important.

16.7. History, Futures and Paradigm Shifts

16.7.1. history

Given the anarchic image of the web, it is strange that the Internet began its development as a US military project. The suitability of the Internet for distributed management and independent growth stems not from an egalitarian or anti-centralist political agenda, but from the need to make the network resilient to nuclear attack with no single point of failure.

In the 1970s when the Internet was first developing it and other networks were mainly targeted at connecting together large computers. It was during the 1980s, with the rise of personal computing that local networks began to become popular.

Timeline – key events for the Internet

1968	First proposal for ARPANET – military & gov't research Contracted to Bolt, Beranek & Newman
1971	ARPANET enters regular use
1973/4	redesign of lower level protocols leads to TCP/IP
1983	Berkeley TCP/IP implementation for 4.2BSD – public domain code
1980s	rapid growth of NSFNET – broad academic use
1990s	WWW and widespread public access to the Internet
2000	WAP on mobile phones web transcends the Internet

However, even before that point very local networks at the lab-bench level had been developed to link laboratory equipment, for example, the IEEE488 designed originally to link Hewlet Packard's proprietary equipment and then becoming an international standard. Ethernet too began life in commercial development at Xerox before becoming the de facto standard for local networking.

Although it is technical features of the Internet (decentralised, resilient to failures, hardware independent) that have made it possible for it to grow, it is the web that has made it become part of popular consciousness. Just as strange as the Internet's metamorphosis from military standard to anarchic cult, is the web's development: from medium of exchange of large high-energy physics data sets to e-commerce, home of alternative web pages and online sex!

16.7.2. paradigm shift

During the 1970s through to the mid 1990s, networks were a technical phenomena, enabling many aspects of business and academic life, but with very little public impact. However this has changed dramatically over the last 5 years and now we think of a networked society. The Internet and other network technologies, such as SMS text messages, are not only transforming society, but at a popular and cultural level defining an era.

International transport, telecommunications and broadcasting had long before given rise to the term global village. However, it seems this was more a phrase waiting for a meaning. Until recently the global village was either parochial – telephoning those you already know, or sanitised – views of distant cultures through the eyes of the travel agent or television camera.

It is only now that we see chat rooms and web home pages allowing new contacts and friendship around the world – or at least amongst those affluent to get Internet access.

Markets too have changed due to global networks. It is not only the transnationals who can trade across the world and even my father-in-law runs a thriving business selling antiques through eBay.

Marketing has also had to face a different form of cross-cultural issue. Although selling the same product, a hoarding in Karachi may well be different from an advert in a magazine in Kentucky, reflecting the different cultural concerns. Global availability of web pages changes all that. You have to create a message that appeals to all cultures – a tall order. Those who try to replicate the targeting of traditional media by having several country specific web sites may face new problems – the global access to event these country specific pages means that the residents of Kentucky and Karachi can compare the different adverts prepared for them, and in so doing see how the company views them and their cultures.

Economics drives so much of popular as well as business development of networked society. One of the most significant changes in the UK in recent years was due to changes in charging models. In the US local calls have long been free and hence so were Internet connections to local points of presence. The costs of Internet access in the UK (and even more important the perception of the cost) held back widespread use. The rise of free or fixed-charge unmetered access changed nearly overnight the acceptability and style of use. Internet access used to be like a lightening guerrilla attack into the web territory, quick in and out before the costs mounted up, but is now a full scale occupation.

The need for telecoms companies across the world to recover large investments in wireless band franchises combined with use rather than connection-based charging made possible by GPRS and third generation mobile services, make it likely that we will see a similar growth in mobile access to global networked information and services.

In an article in 1998, I used the term PopuNET to refer to a change in society that was not yet there, but would come. PopuNET is characterised by network access:

everywhere, everywhen by everyone

This pervasive, permanent, popular access is similar to the so called 'Martini principle' applied more recently to mobile networking – anytime, anyplace, anywhere. Of course Martini never pretends to be anything but exclusive, so not surprisingly these differ on the popular dimension! 'Anyplace, anywhere' does correspond to the pervasive 'everywhere' and 'anytime' tot he permanent 'everywhen'. However, there is a subtle difference, especially between 'anytime' and 'everywhen'. 'Anytime' means that at any time you chose you can connect. 'Everywhen' means that at all places and all times you are connected. When this happens, one ceases to think of 'connectedness' and it simply become part of the backdrop of life.

The changes in charging models have brought the UK closer to 'everywhen' and the US and many other parts of the world are already well down the path. Always-on mobile connectivity will reinforce these changes.

PopuNET will demand new interfaces and products, not just putting web pages on TV screens or spreadsheets on fridge doors. What these new interfaces will be is still uncertain.

16.7.3. futures (near)

It is dangerous to predict far into the future in an area as volatile as this. One development that is already underway, which will make a major impact on user interfaces, is short range networking which will enable various forms of wearable and ubiquitous networks. Another is the introduction of network appliances, which will make the home 'alive' the network.

We have considered network aspects of continuous media at length. The fact that the existing Internet TCP/IP protocols do not enable guaranteed quality-of-service will put severe limits on its ability to act as an infrastructure for services such as video-on-demand or video sharing between homes. The update to TCP/IP which has been underdevelopment for several years, IPv6, will allow prioritised traffic, it falls short of real guaranteed QoS [[IPng, 2001]].

It seems that this in impasse. One of the reasons that IPv6 has taken so long is not the technical difficulty, but backwards compatibility and the problems of uptake on the existing world-wide infrastructure. So evolutionary change is hard. However, revolutionary change is also hard, one cannot easily establish a new parallel international infrastructure overnight. Or perhaps one can.

Mobile phone services have started with an infrastructure designed for continuous voice and are, through a series of quite dramatic changes, moving this towards a fully mixed media/data service. And it is a global network. Furthermore, more an more non-web-based Internet services are using HTTP, the web protocol to talk to one another in order to be 'firewall friendly'. This means that mobile phones and PDAs can be web connected without being Internet connected. So perhaps the future is not an evolution of Internet protocols, but a gradual replacement of the Internet by nth-generation mobile networks. Instead of the Internet on phones, we may even see mobile phone networking standards being used over wires.

References

All web links below and other related links and material at:
<http://www.hiraeth.com/alan/hbhci/network/>

- G. D. Abowd and A. J. Dix (1992). Giving undo attention. Interacting with Computers, 4(3): 317-342.
  <http://www.hcibook.com/alan/papers/undo92/>
- C. Arehart, N. Chidambaram, S. Guruprasad, A. Homer, R. Howell, S. Kasippillai, R. Machin, T. Myers, A. Nakhimovsky, L. Passani, C. Pedley, R. Taylor and M. Toschi (2000). Professional WAP. Wrox Books, ISBN 1861004044.
  The Wrox series includes some of the best programmer-level texts on web and related technology: <http://www.wrox.com/>
- B. Arons (1997). SpeechSkimmer: a system for interactively skimming recorded speech. ACM Transactions on Computer Human Interaction (TOCHI), 4(1):3–38, 1999.
- AvantGo (2001). <http://www.avantgo.com/>
- Barnard, P. and May, J. (1995) Interactions with Advanced Graphical Interfaces and the Deployment of Latent Human Knowledge. In F. Paterno' (ed) Eurographics Workshop on the Design, Specification and Verification of Interactive Systems, Berlin: Springer Verlag.
  More information about ICS at:
  <http://www.mrc-cbu.cam.ac.uk/personal/phil.barnard/ics/>
- V. Bellotti. Design for Privacy in Ubiquitous Computing Environments. Proceedings of CSCW'93. ACM Press, 1993, pp. 77–92
- Benford, S., J. Bowers, L. Fahlen, J. Mariani, T. Rodden (1994). Supporting Cooperative Work in Virtual Environments. The Computer Journal, 37(8):635–668.
- Berlage, T., Spenke, M., The GINA Interaction Recorder. In Proceedings of the IFIP WG2.7 Working Conference on Engineering for Human-Computer Interaction (Ellivuori, Finland, Aug 10--14, 1992).
- Bluetooth (2001). Official Bluetooth SIG Website.
  <http://www.bluetooth.com/>
- W. Buxton and T. Moran. EuroPARC's Integrated Interactive Intermedia Facility (IIIF): Early Experiences. In Multi-User Interfaces and Applications, pages 11–34. S. Gibbs and A.A. Verrijn-Stuart, North-Holland, September 1990. Proceedings of IFIP WG8.4 Confer-ence, Heraklion, Greece.
- A. Campbell and K. Nahrstedt, editors. (1997). Building QoS into Distributed Systems. Kluwer, Boston. ISBN 0-412-80940-0
- S. K. Card, T. P. Moran, and A. Newell. The Psychology of Human Computer Interaction. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1983

- CERN. European Organisation for Nuclear Research. <http://public.web.cern.ch/Public/>
- D. Clarke, A. Dix, D. Ramduny and D. Trepess, editors. (1997). Workshop on Time and the Web, Staffordshire University, June 1997. Extended abstracts at: <http://www.soc.staffs.ac.uk/seminars/web97/papers/> Also report in SIGCHI Bulletin, 30(1), 1998, pp. 30-33. <http://www.acm.org/sigchi/bulletin/1998.1/dix.html>
- Dix, A.J. (1987). *The myth of the infinitely fast machine*. in *People and Computers III — Proceedings of HCI'87*. 1987. Cambridge University Press. p. 215-228.
- A. J. Dix (1991). Formal Methods for Interactive Systems. Academic Press. <http://www.hiraeth.com/books/formal/>
- A. Dix and S. A. Brewster (1994). Causing Trouble with Buttons. Ancilliary Proceedings of HCI'94, Glasgow, Scotland. Ed. D. England. <http://www.hcibook.com/alan/papers/buttons94/>
- A. J. Dix (1994). Seven Years on, the Myth Continues. RR9405, University of Huddersfield. <http://www.hcibook.com/alan/papers/myth95/7-years-on.html>
- A. J. Dix (1995). Dynamic pointers and threads. Collaborative Computing, 1(3): pp. 191-216. More about dynamic pointers at: <http://www.hcibook.com/alan/topics/dynamic/>
- A. J. Dix (1996). Closing the Loop: modelling action, perception and information. (Keynote) AVI'96 - Advanced Visual Interfaces, Eds. T. Catarci, M. F. Costabile, S. Levialdi and G. Santucci. Gubbio, Italy, ACM Press. pp. 20-28. <http://www.hcibook.com/alan/papers/AVI96/>
- A. Dix, J. Finlay, G. Abowd and R. Beale (1998). Human-Computer Interaction, second edition. Prentice Hall. ISBN 0-13-239864-8. Web site including searchable contents: <http://www.hcibook.com/>
- Dix, A., T. Rodden, N. Davies, J. Trevor, A. Friday, K. Palfreyman (2000). Exploiting space and location as a design framework for interactive mobile systems. ACM Transactions on Computer-Human Interaction. (TOCHI). 7(3), pp. 285–321, September 2000
- A. Dix (2001). Cyber-economies and the Real World. Keynote – South African Institute of Computer Scientists and Information Technologists Annual Conference, SAICSIT 2001. Pretoria, September 2001. <http://www.hcibook.com/alan/papers/SAICSIT2001/>
- A. Dix (2001). artefact + marketing = product. Interfaces, no. 48, Autumn 2001, pp. 20-21 <http://www.hiraeth.com/alan/ebulletin/product-and-market/>

- Martin Dodge and Rob Kitchin (2001). Atlas of Cyberspace. Adison Wesley, ISBN 0-201-74575-5 <http://www.cybergeography.org/atlas/>
- P. Dourish and V. Bellotti. Awareness and Coordination in Shared Workspaces. Proceedings of CSCW'92, 1992. pp. 107–114.
- W. Keith Edwards and Tom Rodden (2001). Jini, Example by Example. SUN Microsystems press. ISBN 0-13-033858-3
- Electrolux (1999). Screenfridge. <http://www.electrolux.com/screenfridge/>
- C.A. Ellis and S.J. Gibbs (1989). Concurrency control in groupware systems. Proceedings of 1989 ACM SIGMOD International Conference on Management of Data. SIGMOD Record, 18(2):399–407.
- B. J. Fogg, J. Marshall, O. Laraki, A. Osipovich, C. Varma, N. Fang, J. Paul, A. Rangnekar, J. Shon, P. Swani and M. Treinen (2001). What Makes Web Sites Credible? a report on a large quantitative study. Proceedings of CHI2001, Seattle, 2001. CHI Letters 3(1). ACM Press. pp. 61–68.
- Foster, I. and Kesselman, C. (Eds). The Grid: Blueprint for a New Computing Infrastructure. Morgan-Kaufmann (1999).
- I. Foster (2000). Internet Computing and the Emerging Grid. Nature, WebMatters, 7 December 2000. <http://www.nature.com/nature/webmatters/grid/grid.html>
- W. W. Gaver, R. B. Smith, and T. O'Shea. Effective sounds in complex situations: The ARKola simulation. In S. P. Robertson, G. M. Olson, and J. S. Olson, editors, Reaching through technology - CHI'91 conference proceedings, pages 85-90. ACM Press, New York, April, 1991.
- H-W. Gellersen, M. Beigl, and H. Krull. The MediaCup: Awareness Technology embedded in an Everyday Object. Handheld & Ubiqutious Computing, Lecture notes in computer science; Vol 1707, ISBN 3-540-66550-1; H-W Gellersen ed., Springer, 1999, pp 308-310 <http://www.teco.uni-karlsruhe.de/~michael/publication/mediacuphtml/>
- W. D. Gray, B. E. John, and M. E. Atwood. The precis of project ernestine or an overview of a validation of goms. In P. Bauersfeld, J. Bennett and G. Lynch, editors, Striking a Balance, Proceedings of the CHI'92 Conference on Human Factors in Computing Systems, pages 307-312. ACM Press, 1992.
- C. Greenhalgh (1997). Analysing movement and world transitions in virtual reality tele-conferencing. Proc. ECSCW 97, John A. Hughes, Wolfgang Prinz, Tom Rodden and Kjeld Schmidt (eds.), 1997, Kluwer Academic Publishers, ISBN 0-7923-4638-6, pp. 313-328. <http://citeseer.nj.nec.com/greenhalgh97analysing.html>
- GRID (2001). GRID Forum home page. <http://www.gridforum.org/>

- Halasz, F., T. Moran, and R. Trigg, *NoteCards in a nutshell,* in *Proceedings of the CHI+GI*. 1987, ACM, New York: p. 45-52.
- HFIG (2001). Hybrid Feature Interaction Group, Glasgow UK. <http://www.dcs.gla.ac.uk/research/hfig/>
- D. Hindus, S. D. Mainwaring, N. Leduc, A. E. Hagström and O. Bayley (2001). Casablanca: designing social communication devices for the home. Proceedings of CHI 2001. ACMP Press. pp. 325 - 332
- S. Howard and J. Fabre, editors. Temporal Aspects of Usability, special issue of Interacting with Computers, vol, 11, no. 1, 1999.
- ICQ. <http://www.icq.com/products/whatisicq.html>
- IEEE 802.11 Working Group <http://www.ieee802.org/11/>
- IPng (2001). IP Next Generation (IPng) Working Group home page. <http://playground.sun.com/pub/ipng/html>
- ISO/IEC 7498 (1994). Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model. International Standards Organisation. <http://www.iso.org>
- B. E. John. Extensions of GOMS analyses to expert performance requiring perception of dynamic visual and auditory information. In J. C. Chew and J. Whiteside, editors, Empowering People - Proceedings of CHI'90 Human Factors in Computer Systems, pages 107-115. ACM Press, 1990.
- C. Johnson and P. Gray, editors (1996). Temporal Aspects of Usability, (report of workshop in Glasgow June 1995), SIGCHI Bulletin, ACM, Vol.28 No.2, April 1996. <http://www.acm.org/sigchi/bulletin/1996.2/timeintro.html>
- C. Johnson (1997). What's the Web Worth? The Impact of Retrieval Delays on the Value of Distributed Information. In [[Clarke, 1997]]. <http://www.soc.staffs.ac.uk/seminars/web97/papers/johnson.html>
- JPEG (2001). Joint Photographic Experts Group home page <http://www.jpeg.org/public/jpeghomepage.htm>
- J. J. Kistler and M. Satyanarayanan. Disconnected operation in the CODA file system. ACM Transactions on Computer Systems, 10(1):3–25, February 1992.
- D. Lavery, A. Kilgourz and P. Sykeso (1994). Collaborative Use of X-Windows Applications in Observational Astronomy. People and Computers IX, G. Cockton, S. Draper and G. Wier (eds), Cambridge University Press, 1994, pp. 383–396.
- A. Light and I. Wakeman (2001). Beyond the Interface: Users' Perceptions of Interaction and Audience on Websites. Dave Clarke and Alan Dix (eds.). Special Issue Interfaces for the Active Web (Part 1). Interacting with Computers, 13(3), February 2001. pp. 401–426

- A. Light (2001). Interaction at the Producer-User Interface: An Interdisciplinary Analysis of Communication and Relationships through Interactive Components on Websites for the purpose of improving Design.. DPhil Thesis. University of Sussex, UK.
  <http://www.cogs.susx.ac.uk/users/annl/summary.html>
- S. Lock, J. Allanson, P. Phillips: User-Driven Design of a Tangible Awareness Landscape. Symposium on Designing Interactive Systems 2000: 434-440.
  <http://www.acm.org/pubs/citations/proceedings/chi/347642/p434-lock/>
- J. A. Mariani and T. Rodden. The impact of CSCW on database technology. In Proc. ACM Conference on Computer Supported Cooperative Work, 1991.
  Includes critique of 'transparency' in a CSCW setting.
- M. Mauve (2000). Consistency in replicated continuous interactive media. Proceedings of CSCW'2000. Philadelphia, USA, ACM Press. pp. 181-190
- E. Mitsopoulos (2000). A Principled Approach to the Design of Auditory Interaction in the Non-Visual User Interface. DPhil Thesis, University of York,UK.
  Thesis:
  <http://www.cs.york.ac.uk/ftpdir/reports/YCST-2000-07.zip>
  Web pages on auditory scene analysis and auditory illusions:
  <http://www-users.cs.york.ac.uk/~enm/asa/asa.html>
- S. E. McDaniel and T. Brinck. Awareness in Collaborative Systems: A CHI 97 Workshop (report). SIGCHI Bulletin, ACM, Vol.29 No.4, October 1997.
  Workshop report:
  <http://www.acm.org/sigchi/bulletin/1997.4/mcdaniel.html>
  Workshop web pages:
  <http://www.usabilityfirst.com/groupware/awareness/workshop/>
- B. McManus (1997). Compensatory Actions for Time Delays. In [[Clarke, 1997]]
  <http://www.soc.staffs.ac.uk/seminars/web97/papers/barbara.html>
- L. I. Millett, B. Friedman and E. Felten (2001). Cookies and Web Browser Design: toward informed consent online. Proceedings of CHI2001, Seattle, 2001. CHI Letters 3(1). ACM Press. pp. 46–52.
- MPEG (2001). Moving Picture Experts Group home page
  <http://www.cselt.it/mpeg/>
- M. Olson and S. Bly. The Portland Experience: a report on a distributed research group. International Journal of Man-Machine Studies, 34:211–228, 1991.

- Palfreyman, K. and Rodden, T. A Protocol for User Awareness on the World Wide Web, in Proceedings of CSCW'96, (Boston, Massachusetts, Nov. 1996), ACM Press, 130–139.
- Patterson, J.F, Day, M. and Kucan, J. Notification Servers for Synchronous Groupware, in Proceedings of CSCW'96 (Cambridge Massachusetts, 1996), ACM Press, 122–129.
- Pausch, R., *Virtual reality on five dollars a day*, in *CHI'91 Conference Proceedings*, S.P. Robertson, G.M. Olson, and J.S. Olson, Editors. 1991, Addison Wesley: p. 265-270.
- D. Ramduny and A. Dix (1997). Why, What, Where, When: Architectures for Co-operative work on the WWW. Proceedings of HCI'97, Eds. H. Thimbleby, B. O'Connaill and P. Thomas. Bristol, UK, Springer. pp. 283-301.
<http://www.hcibook.com/alan/papers/WWWW97/>
- D. Ramduny, A. Dix and T. Rodden (1998). Getting to Know: the design space for notification servers. Proceedings of CSCW'98. pp. 227-235
<http://www.hcibook.com/alan/papers/GtK98/>
- Resnick, P. and H.R. Varian (guest editors), (1997). Special Issue on Recommender Systems. CACM. 40(3):56–89.
- Rodden, T. 1996. Populating the application: a model of awareness for cooperative applications. In Proceedings of the 1996 ACM Conference on Computer-Spported Cooperative Work (CSCW '96, Boston, MA, Nov. 16–20), M. S. Ackerman, Ed. ACM Press, New York, NY, 87–96.
- N. Roussel. Exploring new uses of video with videoSpace. In Proc. of EHCI'01, the 8th IFIP Working Conference on Engineering for Human-Computer Interaction, Lecture Notes in Computer Science. Springer-Verlag, May 2001.
<http://www-iiuf.unifr.ch/%7erousseln/publications/EHCI01.pdf>
- N. Roussel. Beyond Webcams and Videoconferencing: Informal Video Communication on the Web. In Proceedings of The Active Web, Stafford, pages 65-69, January 1999
<http://www.visualize.uk.com/conf/activeweb/proceed/pap15/>
- Sandor, O., Bogdan, C., And Bowers, J. 1997. Aether: An Awareness Engine for CSCW. In Proceedings of the Fifth European Conference on Computer Supported Cooperative Work (ECSCW'97), J. Hughes, Ed. Kluwer Academic, Dordrecht, Netherlands, 221–236.
- A. Sangha. Legal Implications Of Location Based Advertising. Interview for the WAP Group, June 2001.
<http://www.thewapgroup.com/53762_1.DOC>
- B. Schneier (1996). Applied Cryptography, Second Edition. Wiley. ISBN: 0471-11709-9.
This is the best single point for cryptography, encryption, authentication,

digital signatures, including full algorithms and source code on CD-ROM.

- SETI@home – the search for extra-terrestrial intelligence. <http://setiathome.ssl.berkeley.edu/>
- D. Siegal (1999). Futurize your Enterprise. Wiley, ISBN 0471-35763-4. <http://www.futurizenow.com>
- Shneiderman, B., *Response time and display rate in human performance with computers.* ACM computing surveys, 1984. **16**(3): p. 265-286.
- M. Stefik, D. G. Bobrow, G. Foster, S. Lanning, and D. Tatar. WYSIWIS revisited: early experiences with multiuser interfaces. ACM Transactions on Office Information Systems, 5(2):147-167, 1987.
- R. Stevens. UNIX Network Programming, Volume 1, Second Edition: Networking APIs: Sockets and XTI, Prentice Hall, 1998.
- R. Stevens. UNIX Network Programming, Volume 2, Second Edition: Interprocess Communications, Prentice Hall, 1999.
  W. Richard Stevens books are classics on Internet protocols and programming. <http://www.kohala.com/start/>
- L. Stifelman, B. Arons and C. Schmandt (2001). The Audio Notebook: paper and pen interaction with structured speech. Proceedings of CHI2001, Seattle, 2001. CHI Letters 3(1). ACM Press. pp. 182–189.
- Chengzheng Sun and Clarence Ellis (1998). Operational transformation in real-time group editors: issues, algorithms, and achievements. Proceedings of CSCW'98. Seattle, USA. ACM Press pp. 59-68.
- SUN Microsystems (2001). Awarenex. <http://www.sun.com/research/features/awarenex/>
- Nicolas Vidot, Michelle Cart, Jean Ferriz and Maher Suleiman (2000). Copies convergence in a distributed real-time collaborative environment. Proceedings of CSCW'2000. Philadelphia, USA, ACM Press. pp. 171-180 in ACM Digital Library
- Wisneski, G., Ishii, H., Dahley, A., Gorbet, M., Brave, S., Ullmer, B., Yarin, P. (1998). Ambient Display: Turning Architectural Spache into an Interface between People and Digital. Information. In: Proceedings of the First International Workshop on Cooperative Buildings (CoBuild'98), Darmstadt, Germany (February 25-26, 1998). Lecture Notes in Computer Science, Vol. 1370. Springer-Verlag, Heidelberg.
- Witten, I. H., Thimbleby, H. W., Coulouris, G., and Greenberg, S. (1991). Liveware: a new approach to sharing data in social networks. International Journal of Man-Machine Studies, 34:337–348.
- T. G. Zimmerman (1996). Personal Area Networks: Near-field intrabody communication. IBM Systems Journal, Vol. 35, Nos. 3&4, 1996. <http://isj.www.media.mit.edu/projects/isj/SectionE/609.htm>