What is groupware?

Software *specifically* designed

- to support group working
- with cooperative requirements in mind

NOT just tools for communication

Groupware can be classified by

- *when* and *where* the participants are working
- the *function* it performs for cooperative work

Specific and difficult problems with groupware implemention

The Time/Space Matrix

Classify groupware by:

when the participants are working,
at the same *time* or not

where the participants are working,
at the same *place* or not

	same place	different place
same time	face-to-face conversation	telephone
different time	post-it note	letter

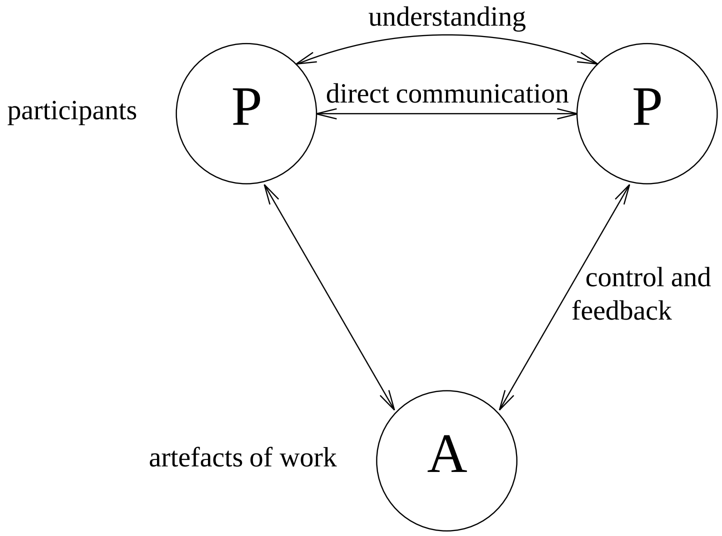
Common names for axes:

time: synchronous/asynchronous
place: co-located/remote

Classification by Function

Cooperative work involves:

Participants who are working
Artefacts upon which they work



What interactions does a tool support?

computer-mediated communication
direct communication between participants

meeting and decision support systems
common understanding

shared applications and artefacts
control and feedback with shared work objects

Email and bulletin boards

asynchronous/remote

familiar and most successful groupware

Recipients of email:
    direct in `To:` field
    copies in `Cc:` field
delivery identical — difference is *social* purpose

differences between email and BBs

fan out

one-to-one	—	email, direct communication
one-to-many	—	email, distribution lists
		BBs, broadcast distribution

control

sender	—	email, private distribution list
administrator	—	email, shared distribution list
recipient	—	BBs, subscription to topics

Structured message systems

asynchronous/remote

- 'super' email — cross between email and a database
- sender fills in special fields
- recipient filters and sorts incoming mail based on field contents

```
Type: Seminar announcement
To: all
From: Alan Dix
Subject: departmental seminar

Time: 2:15 Wednesday
Place: D014
Speaker: W.T. Pooh
Title: The Honey Pot
Text: Recent research on socially constructed
      meaning has focused on the image of the
      Honey Pot and its dialectic interpretation
      within an encultured hermeneutic.
      This talk ...
```

but, work by the sender ... benefit for the recipient

conflict

global structuring by designer
vs.
local structuring by participants

Video conferences and communication

synchronous/remote

Technology emerging: ISDN + video compression

major uses:

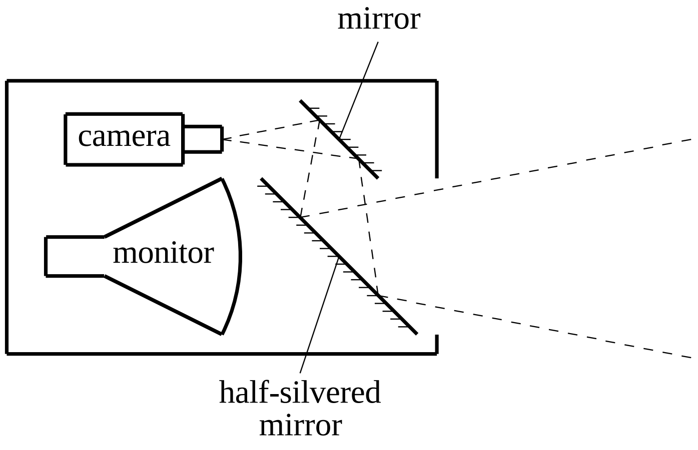
- video conferences
- pervasive video for social contact
- integration with other applications

often cheaper than face-to-face meetings
(telecommunications costs vs. air flights)

but not a substitute:

- small field of view
- lack of reciprocity
- poor eye contact

One solution for lack of eye contact — the video-tunnel



Meeting and decision support systems

In design, management and research, we want to:

- generate ideas
- develop ideas
- record ideas

primary emphasis — common understanding

Three types of system:

argumentation tools

asynchronous co-located

recording the arguments for design decisions

meeting rooms

synchronous co-located

electronic support for face-to-face meetings

shared drawing surfaces

synchronous remote

shared drawing board at a distance

argumentation tools

asynchronous co-located

hypertext like tools to record *design rationale*

Two purposes:

- remining the designers of the reasons for decisons
- communicating rationale between design teams

Mode of collaboration:

- very long term
- sometimes synchronous use also

Example: gIBIS (issue based information system)

various node types including:

issues	e.g., 'number of mouse buttons'
positions	e.g., 'only one button'
arguments	e.g., 'easy for novice'

linked by relationships such as:

argument *supports* position

e.g., 'easy for novice' *supports* 'only one button'

Meeting rooms

synchronous co-located

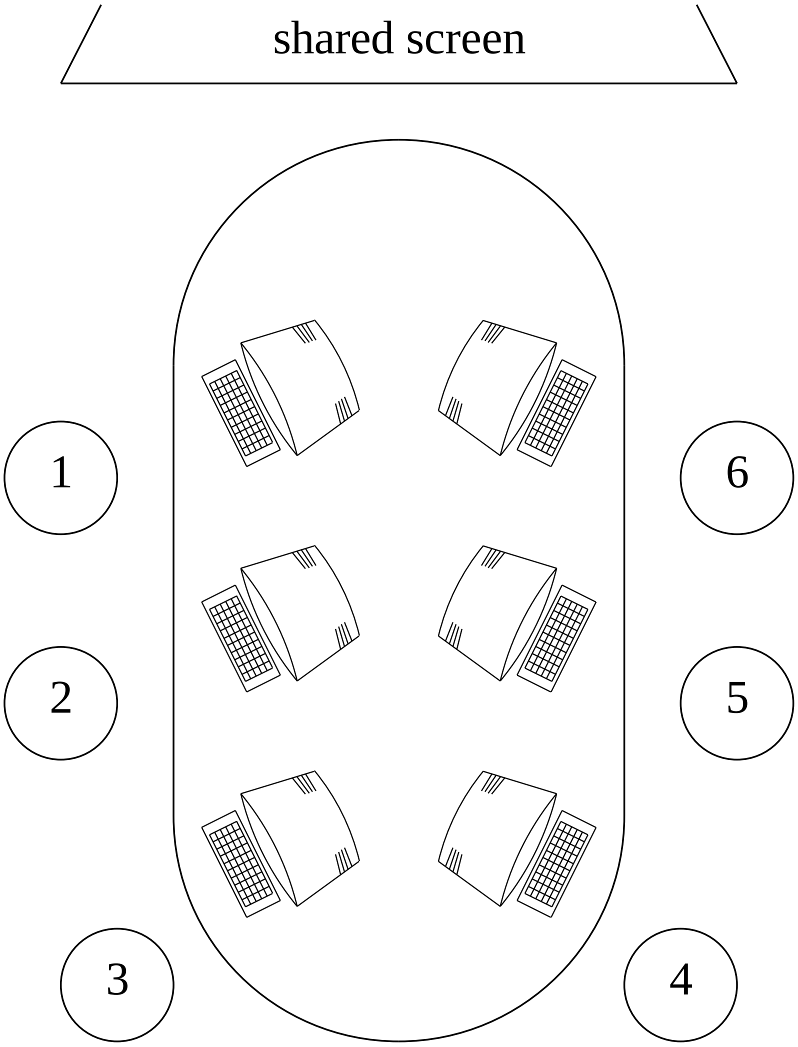
electronic support for face-to-face meetings

- individual terminals (often recessed)
- large shared screen (electronic whiteboard)
- special software
- U or C shaped seating around screen

Various modes:
brainstorming, private use, WYSIWIS

WYSIWIS — 'what you see is what I see'
all screens show same image
any participant can write/draw to screen

Typical meeting room



Issues for cooperation

Argumentation tools

concurrency control
two people access the same node
one solution is node *locking*

notification mechanisms
knowing about others' changes

Meeting rooms

floor holders one or many?

floor control policies
who can write and when?
solution: *locking* + social protocol

group pointer
for deictic reference (*this* and *that*)

Shared work surfaces

synchronous remote

At simplest, meeting rooms at a distance, but ...

- additional audio/video essential for *social protocols* and discussion
- network delays can be major problem

Additional special effects:

- participants write onto large video screen problems with *parallax*
- shadow of other participant's hands appears on screen
- electronic image integrated with video and paper images

Example: TeamWorkStation
remote teaching of Japanese calligraphy
student's strokes on paper overlaid with video of instructor's strokes

Shared Applications and Artefacts

Compare purpose of cooperation:
meeting rooms and decison support systems
— develop shared understanding
shared applications and artefacts
— work on the same objects

technology similar but primary purpose different

many different modalities (time/space matrix)

- **shared windows** — *synchronous remote/co-located*
- **shared editors** — *synchronous remote/co-located*
- **co-authoring systems** — largely *asynchronous*
- **shared diaries** — largely *asynchronous remote*
- **shared information** — any, but largely *asynchronous*

synchronous remote applications usually require additional audio/video channel

Similar – but different

Shared PCs and shared window systems

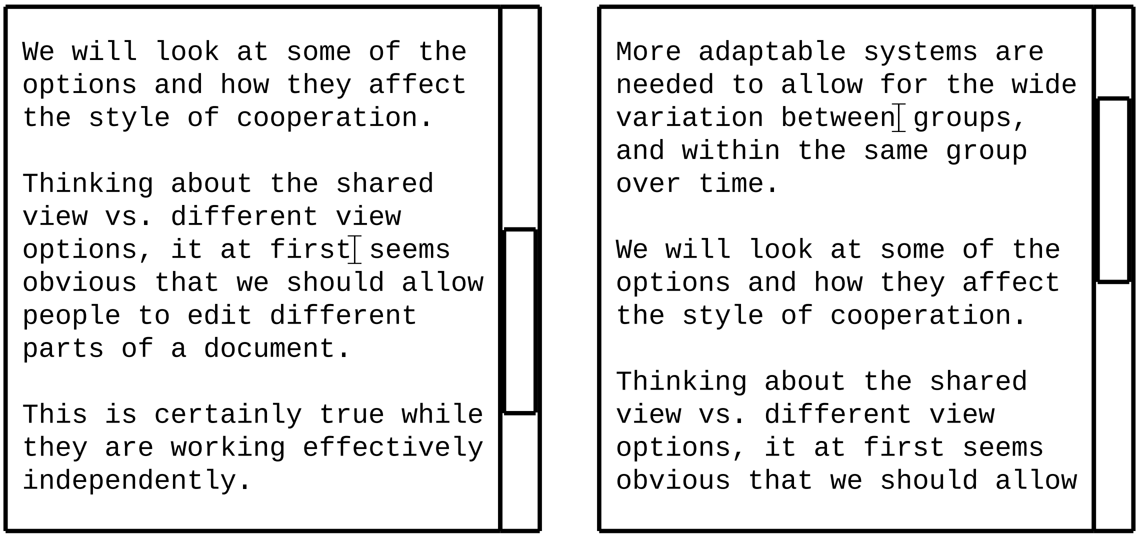
- Multiplex keyboard and screen
- Individual applications *not collaboration aware*
- Floor control problems:

user A types:	‘*interleave the*’
user B types:	‘`keystrokes`’
result:	‘*in*`key`*ter*`s`*l*`tr`*ea*`oke`*ve t*`s`*he*’

Shared editors

- An editor which is *collaboration aware*
- One document — several users
- Similar to shared screen in meeting room …
  …with similar floor control problems!
- Additional problem — multiple views

Shared editors — multiple views



your screen your colleague's screen

Options:

same view or different view
single or separate insertion points

Single view $\Longrightarrow$ scroll wars

Multiple views $\Longrightarrow$ loss of context with *indexicals*

'I don't like the line at the top'
'but I just wrote that!'

Co-authoring systems

Emphasis is on long term document production, not editing

Two levels of representation

- the document itself
- annotation and discussion

Often some form of hypertext structure used

Similar problems of *concurrency control* to argumentation systems

Sometimes include *rôles*:
  author, commentator, reader, ...
  but who decides the rôles?
  and how flexible are they?

Shared diaries

Idea:

- make diaries and calendars more easily shared
- allow automatic meeting scheduling etc.

Issues for cooperation:

privacy who can see my diary entries?

control who can write in my diary?

Similar to file sharing issues, but need to be lightwight

Many systems have failed because they ignored these issues

Communication through the artefact

When you change a shared application:

- you can see the effect — *feedback*
- your colleages can too — *feedthrough*

feedtrough enables
communication through the artefact

Not just with 'real' groupware

Shared data is pervasive:

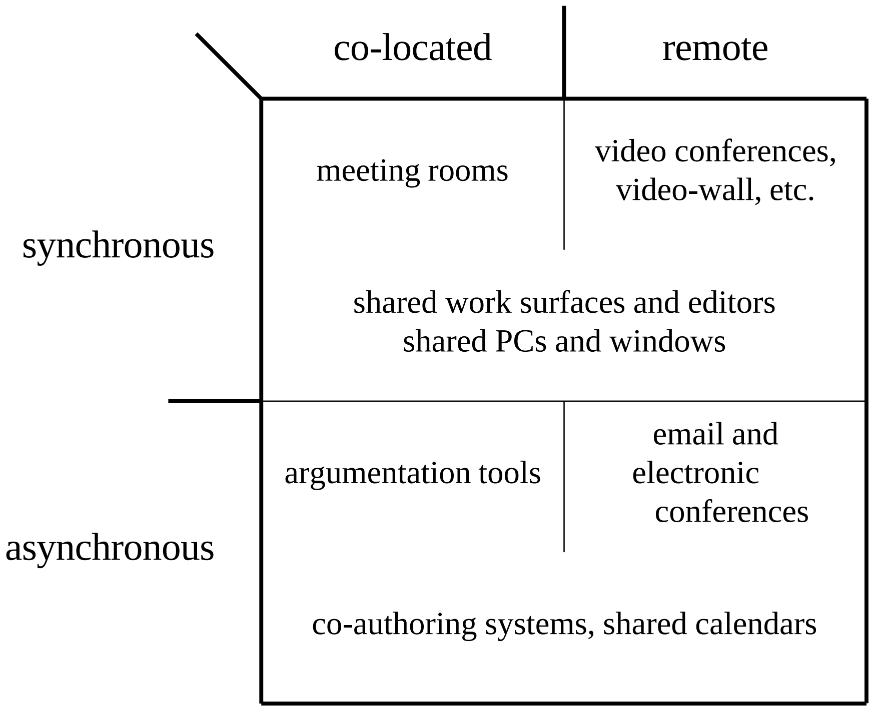
- shared files and databases
- casework files (often non-electronic)
- passing electronic copies of documents
- passing copies of spreadsheets

Often need direct communication as well, but
indirect communication *through the artefact* central

Few examples of explicit design for cooperation.

Liveware is an exception,
a database with 'merging' of copies

Time/space matrix revisited



Refined time/space matrix

<table>
<tr><th></th><th>co-located</th><th>remote</th></tr>
<tr><td rowspan="2">(a) concurrent synchronized</td><td>meeting rooms</td><td>video conferences, video-wall, etc.</td></tr>
<tr><td colspan="2">shared work surfaces and editors
shared PCs and windows</td></tr>
<tr><td>(a/b) mixed</td><td colspan="2">co-authoring systems, shared calendars</td></tr>
<tr><td>(b) serial</td><td colspan="2">argumentation tools</td></tr>
<tr><td>(c) unsynchronized</td><td colspan="2">email and structured messages
electronic conferences</td></tr>
</table>

Mobile workers and home workers have infrequent communicatio
— they require *unsynchronised* groupware

Few ‘research’ systems address this area

NO current system allows fluid movement
between synchronised/unsynchronised operation

Shared information

Granularity of sharing

chunk size

small — edit same word or sentance

large — section or whole document

update frequency

frequent — every character

infrequent — upon explicit 'send'

level of sharing

output: shared object
shared view
shared presentation

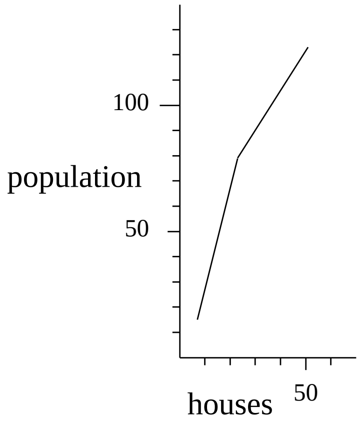
input: single insertion point
- shared virtual keyboard

multiple insertion points
- other participants visible
- group pointer
- no visibility

Levels of shared output

presentation

houses	population
7	15
23	79
51	123



view

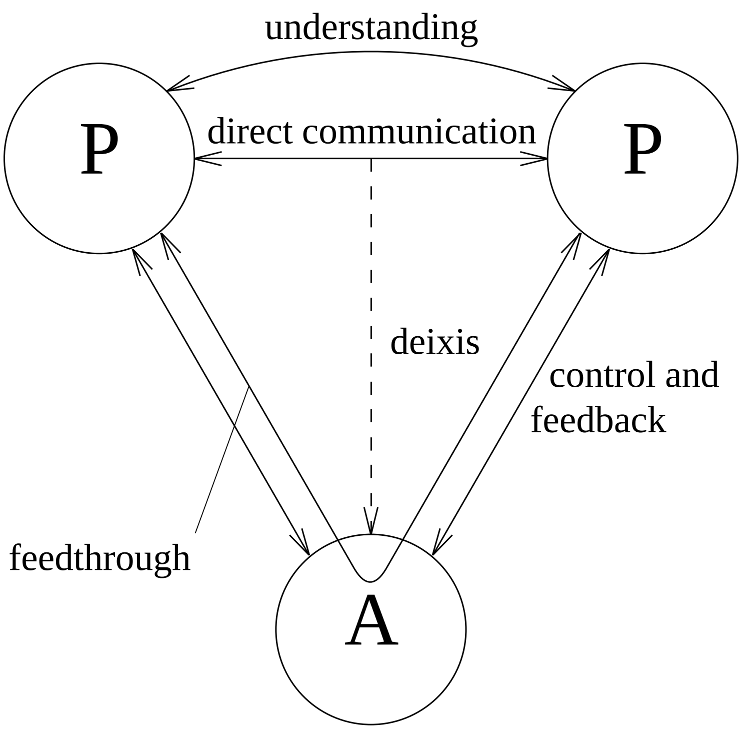
select houses, population from VILLAGE_STATS
where population < 200
sort by houses ascending

object

VILLAGE_STATS

village	houses	population
Burton	23	79
Marleigh	339	671
Westfield	7	15
Thornby	51	123

Integrating communication and work



Added: *deixis* — reference to work objects
feedthorough
— for communication through the artefact

Classified groupware by function it supported

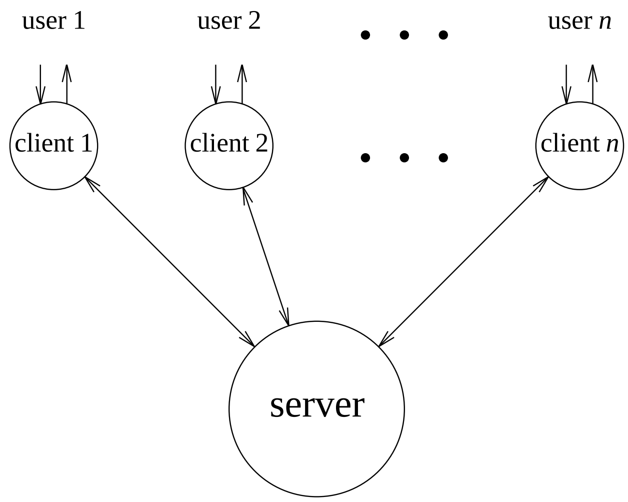
Good groupware — open to all aspects of cooperation

e.g., annotations in co-authoring systems
embedding direct communication

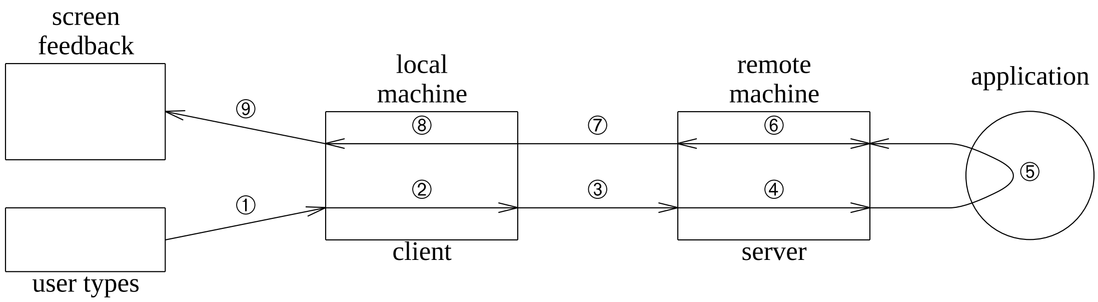
bar codes — form of deixis
aids diffuse large scale cooperation

Architectures for groupware I

Client-server architecture



Feedback and network delays



At least 2 network messages + four context switches

With protocols 4 or more network messages

Architectures for groupware II

Different architectures:

centralised — single copy of application and data

client-server — simplest case

N.B. opposite of X windows client/server

master-slave special case of client-server

N.B. server merged with one client

replicated — copy on each workstation

also called *peer-peer*

+ local feedback

− race conditions

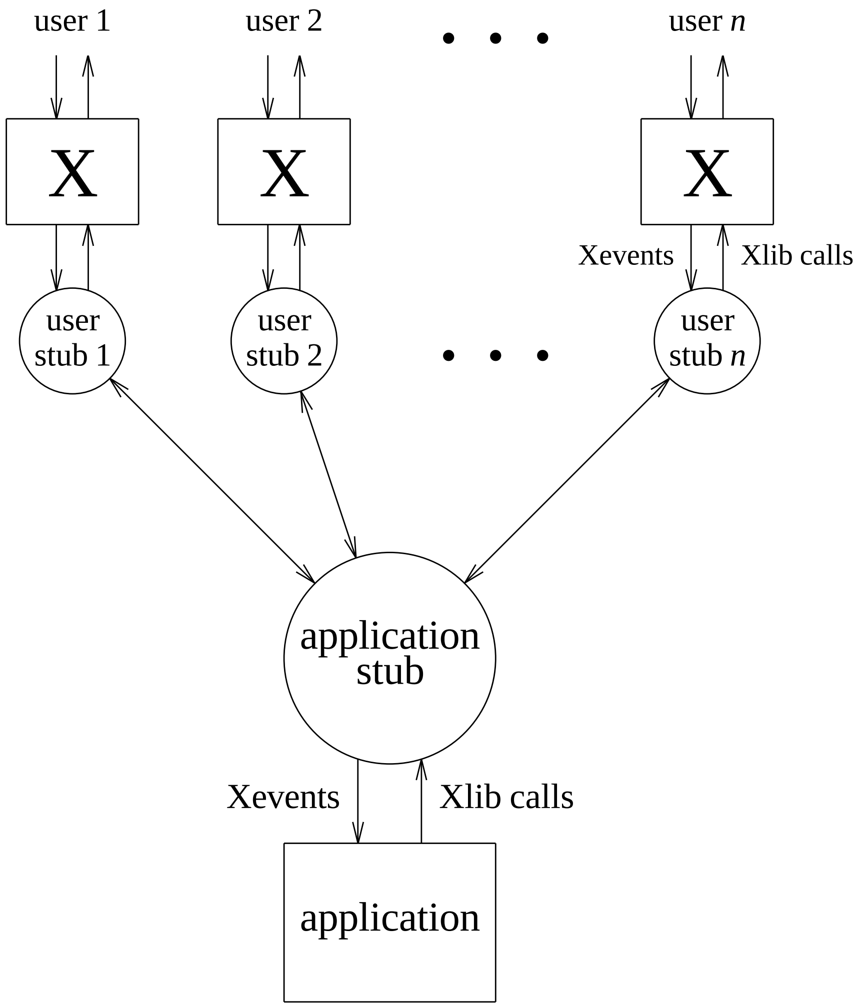
Often 'half way' archtectures:

- local copy of application
- central database
- local cache of data for feedback
- some hidden locking

Shared window architecture

- Non-collaboration aware applications
  $\Longrightarrow$ client/server approach
  corresponding feedback problems

- no 'functionality' — in the application
  but must handle floor control



Feedthrough

Need to inform all other clients of changes

Few networks support *broadcast* messages, so ...
n participants $\Longrightarrow n-1$ network messages!

Solution: increase granularity
reduce frequency of feedback
but ...

poor feedthrough $\Longrightarrow$ loss of shared context

Tradeoff: timeliness vs. network traffic

Graphical toolkits

Designed for *single* user interaction

Problems for groupware include

- pre-emptive widgets
  (e.g., pop-up menus)
- over-packaged text
  (single cursor, poor view control)

notification based toolkits with *callbacks* help (see Ch. 10)

Robustness and scaleability

crash in single-user interface — one sad user
crash in groupware — disaster !

but, groupware complex: networks, graphics etc.

- network or server fails — standard solutions
- client fails — three 'R's for server:
  robust — server should survive client crash
  reconfigure — detect and respond to failure
  resynchronise — catch up when client restarts
- errors in programming
  defensive programming
  simple algorithms
  formal methods
- unforeseen sequences of events
  deadlock — never use blocking I/O
  never assume particular orders
  network packet $\neq$ logical message

Scaling up to large numbers of users?

Testing and debugging: hard!