COAST

An Open Source Smalltalk Framework
to Build Synchronous Collaborative Applications

Jan Schümmer, Till Schümmer, Christian Schuckmann

GMD - IPSI, Darmstadt, Germany
intelligent views, Darmstadt, Germany

{jan.schuemmer|till.schuemmer}@gmd.de
christian.schuckmann@i-views.de

focus

Support the development of

- object oriented,
- synchronous,
- interactive, and
- complex
  (e.g. hypermedia applications)

groupware.

motivation

requirements

application structure

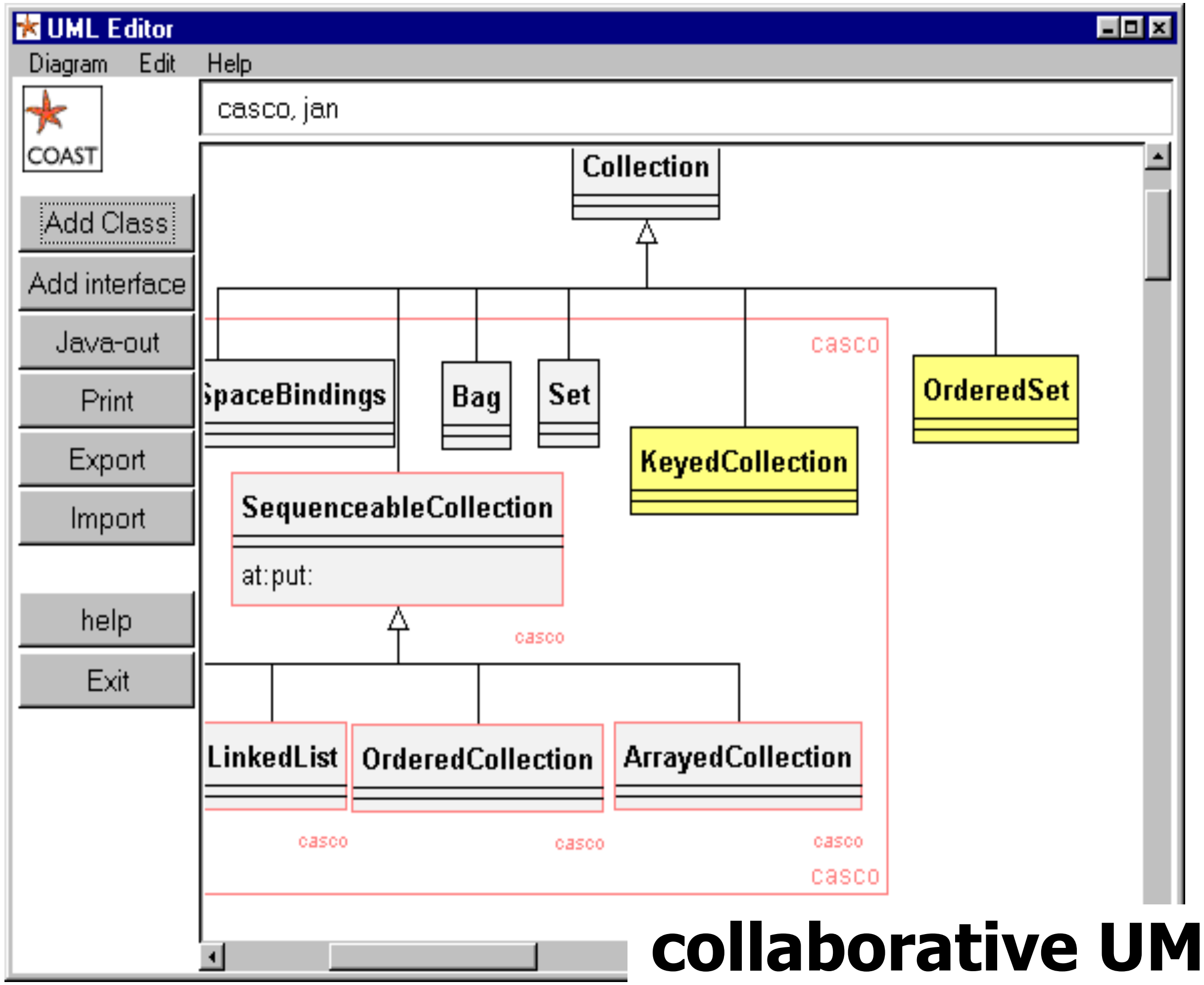
groupware model

realization

usage experiences

COAST

sample groupware application



collaborative UML editor

motivation

requirements

application structure

groupware model

realization

usage experiences

GMD IPSI

problem statement

Writing groupware is difficult.

It is different from single-user application development.

- more than one user at a time (multiple I/O)
- provision of group awareness
- support of different collaboration modes

It is error prone.

- process synchronization
- data consistency
- network (components) failure

support groupware developers

architecture

- reference architecture
- ready-to-use components
  - e.g. server component

model

- class hierarchy serves as a template for groupware applications

implementation

- do and hide as much of the 'hard and dirty work' as possible
  - e.g. synchronisation of shared objects

requirements

groupware-specific requirements

- group awareness
- coupling control
- session management
- floor control

general requirements

- ease of use => right level of abstraction
- consistency, uniform approach
- reusability

motivation

requirements

application structure

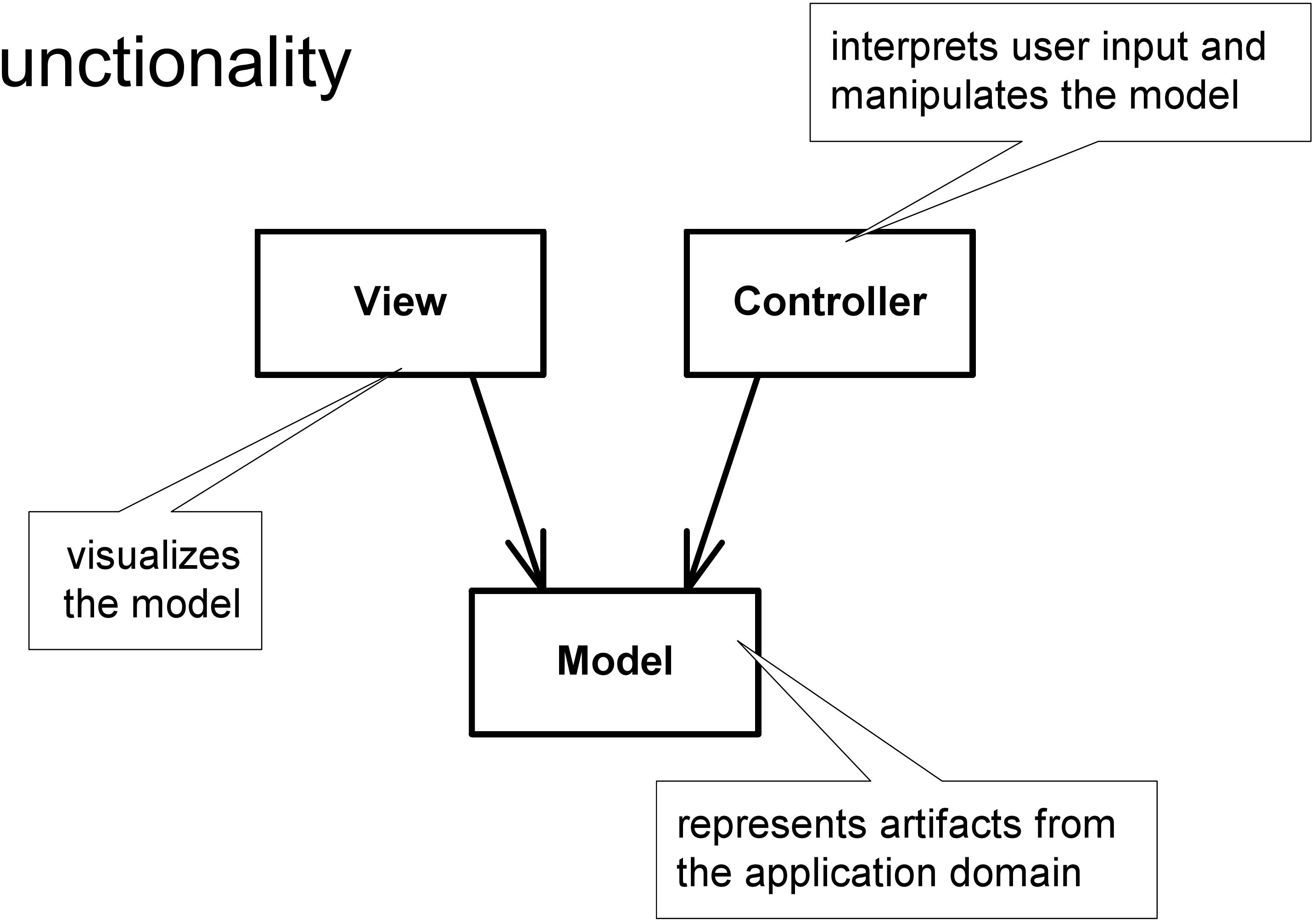
groupware model

realization

usage experiences

structure in single-user applications 1

divide functionality


interprets user input and manipulates the model
View
Controller
visualizes the model
Model
represents artifacts from the application domain

motivation

requirements

application structure

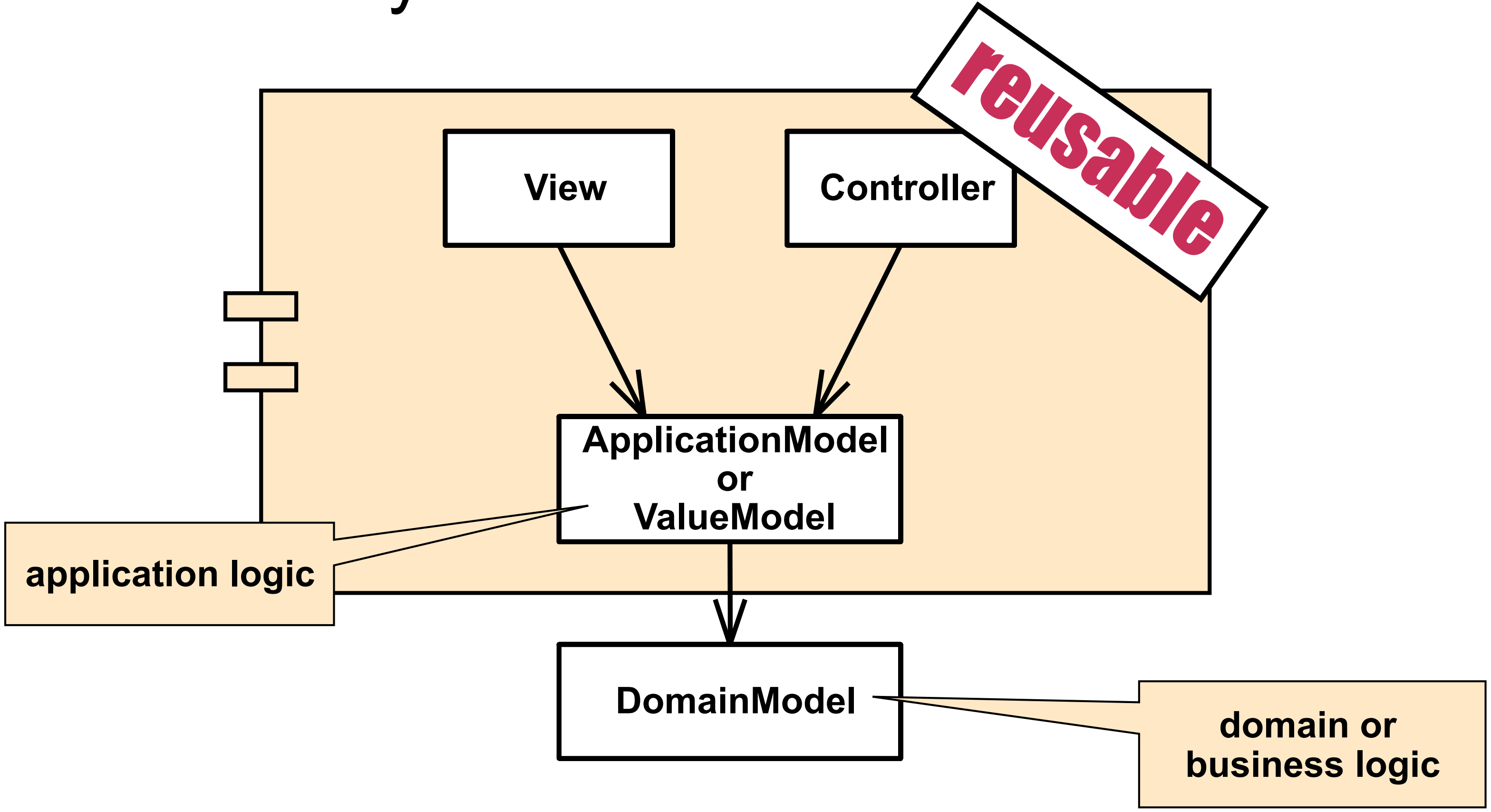
groupware model

realization

usage experiences

structure in single-user applications 2

increase reusability



application logic

domain or business logic

motivation

requirements

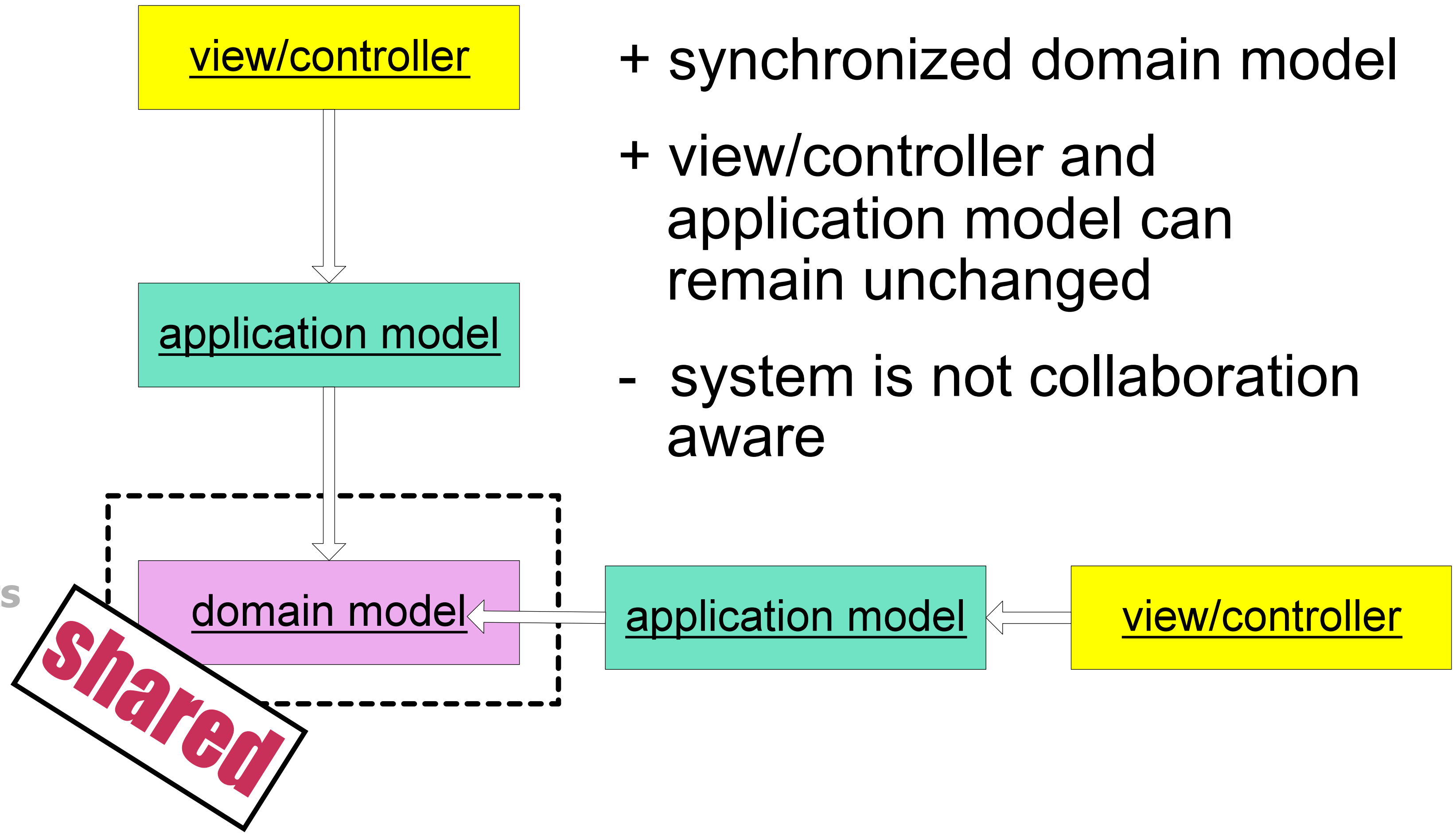
application structure

groupware model

realization

usage experiences

from single-user to multi-user: sharing the domain model



+ synchronized domain model

+ view/controller and application model can remain unchanged

- system is not collaboration aware

motivation

requirements

application structure

groupware model

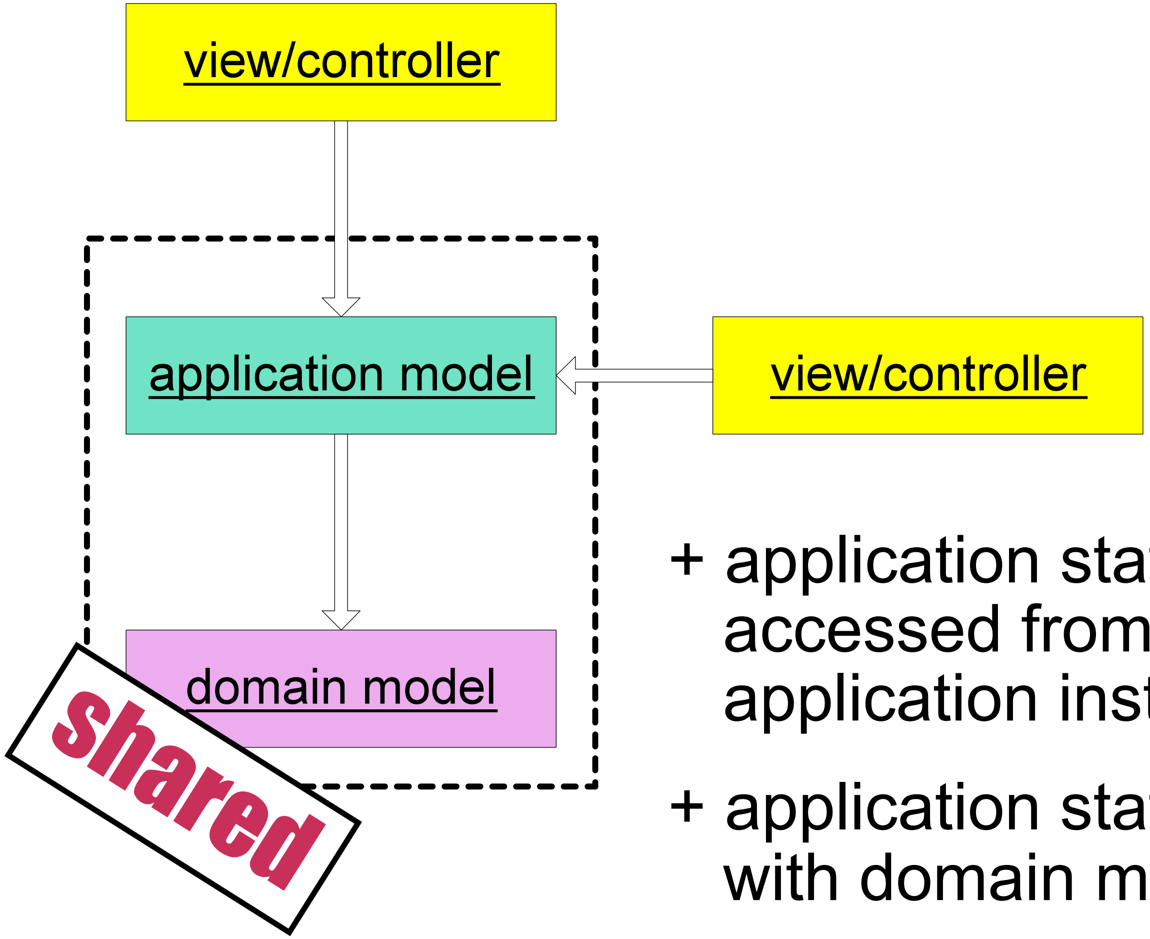
realization

usage experiences

COAST

from single-user to multi-user: sharing the application model

GMD IPSI



+ application state can be accessed from each application instance

+ application state consistent with domain model state

motivation

requirements

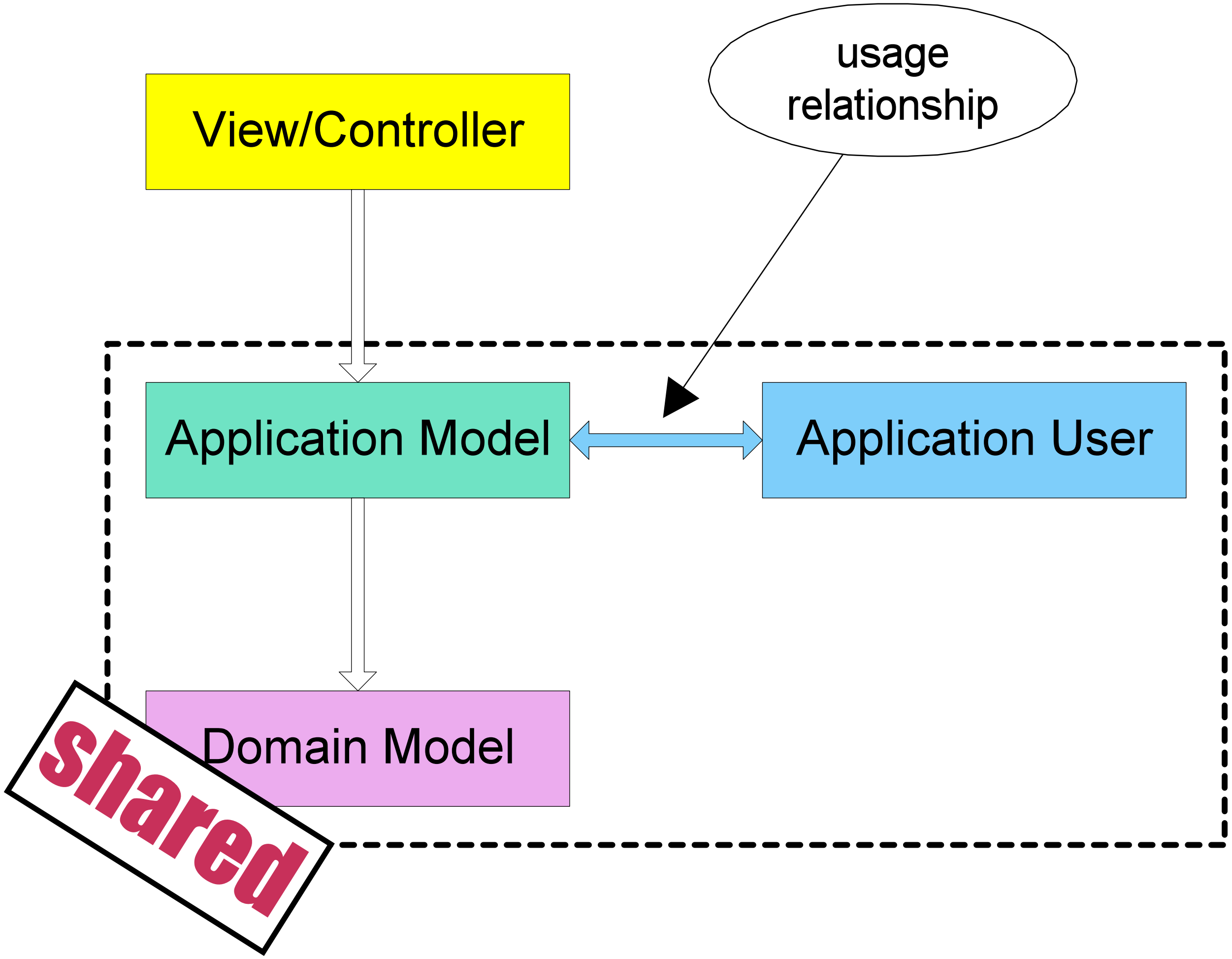
application structure

groupware model

realization

usage experiences

from single-user to multi-user: the user comes into play



motivation

requirements

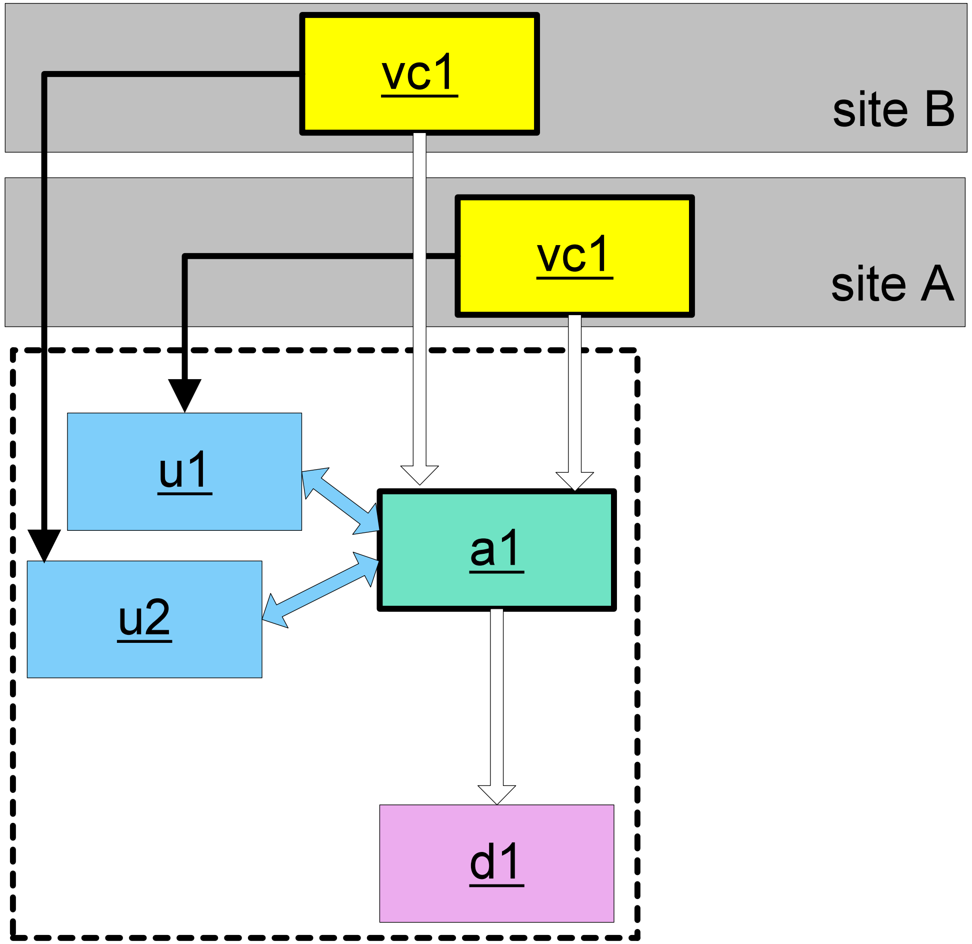
application structure

groupware model

realization

usage experiences

logical session management



motivation

requirements

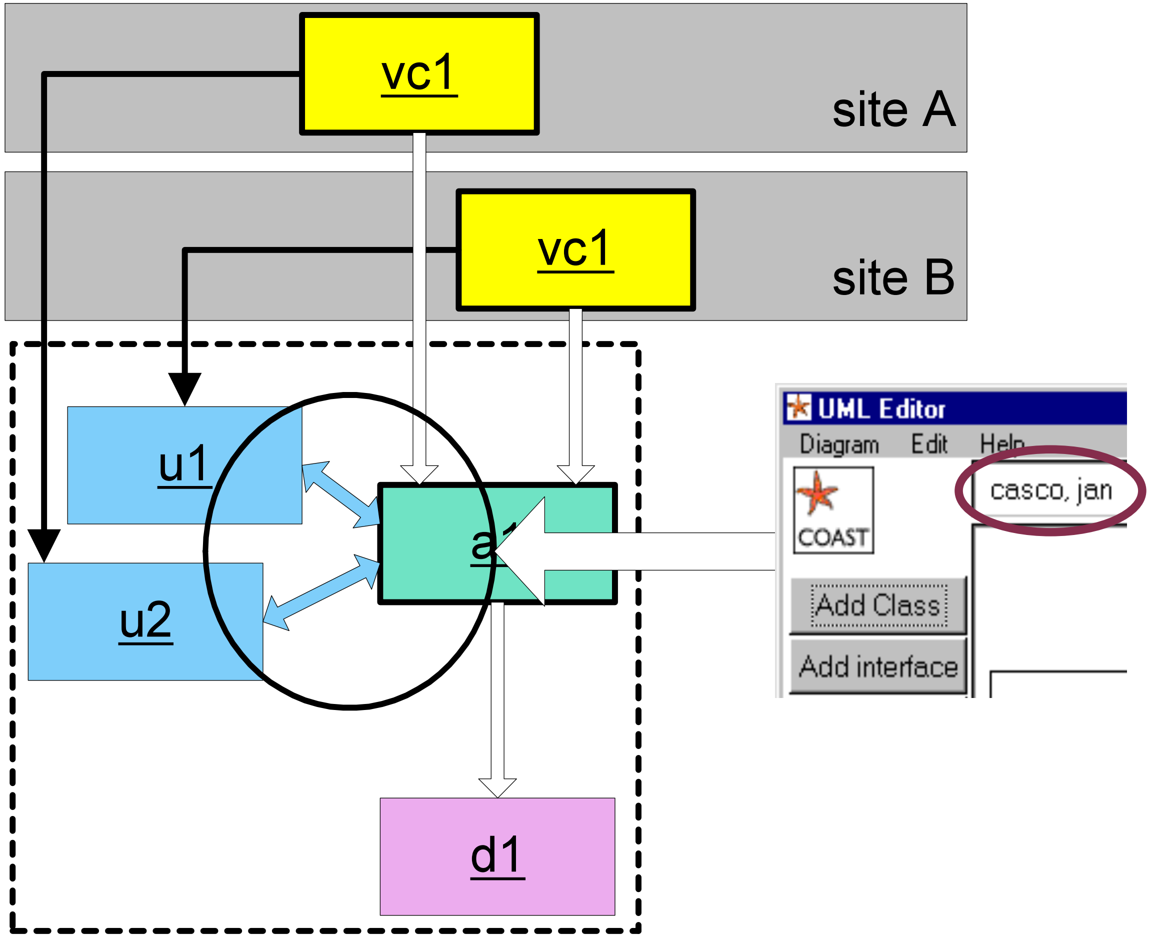
application structure

groupware model

realization

usage experiences

provision of group awareness



motivation

requirements

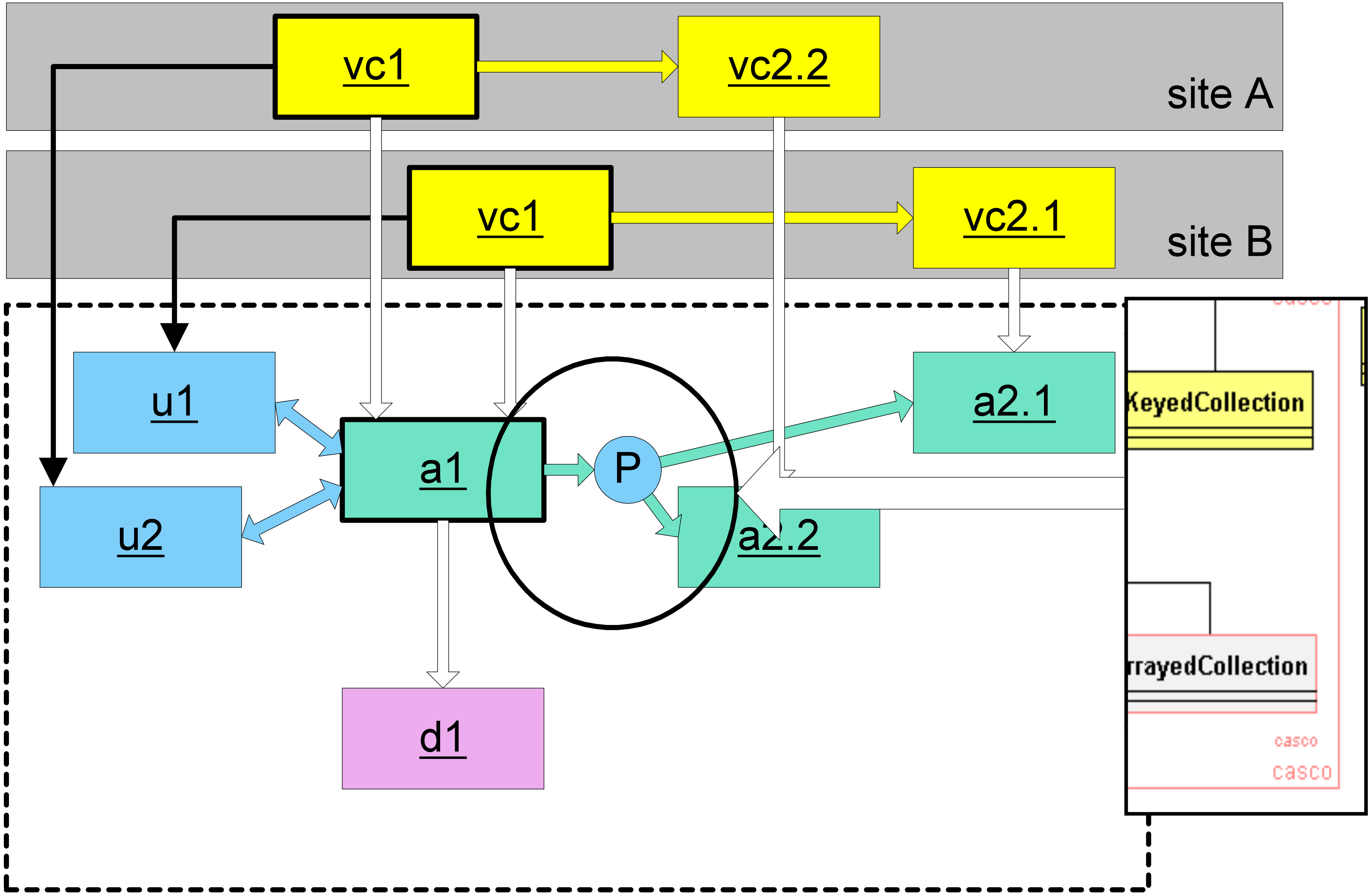
application structure

groupware model

realization

usage experiences

coupling control



motivation

requirements

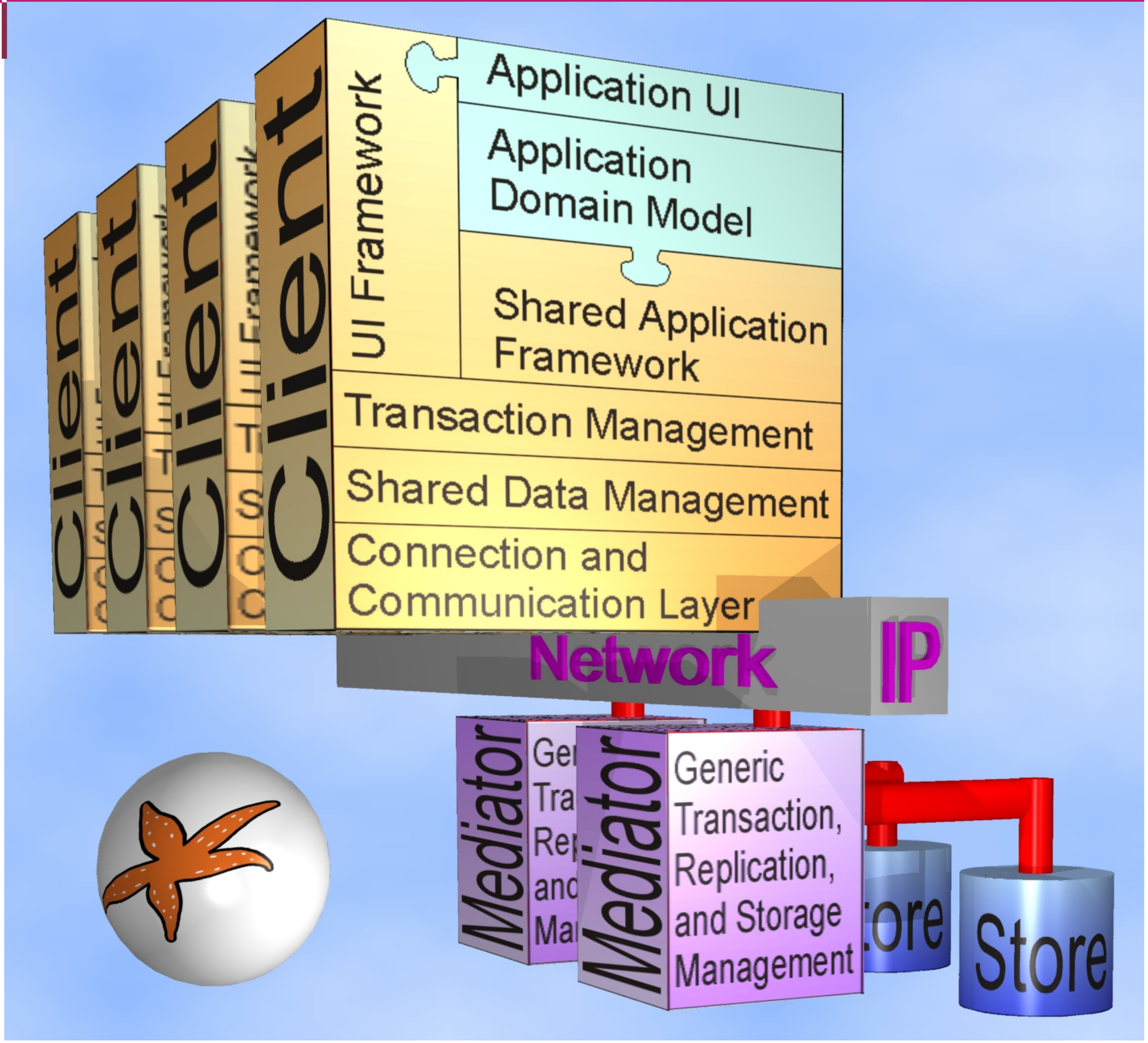
application structure

groupware model

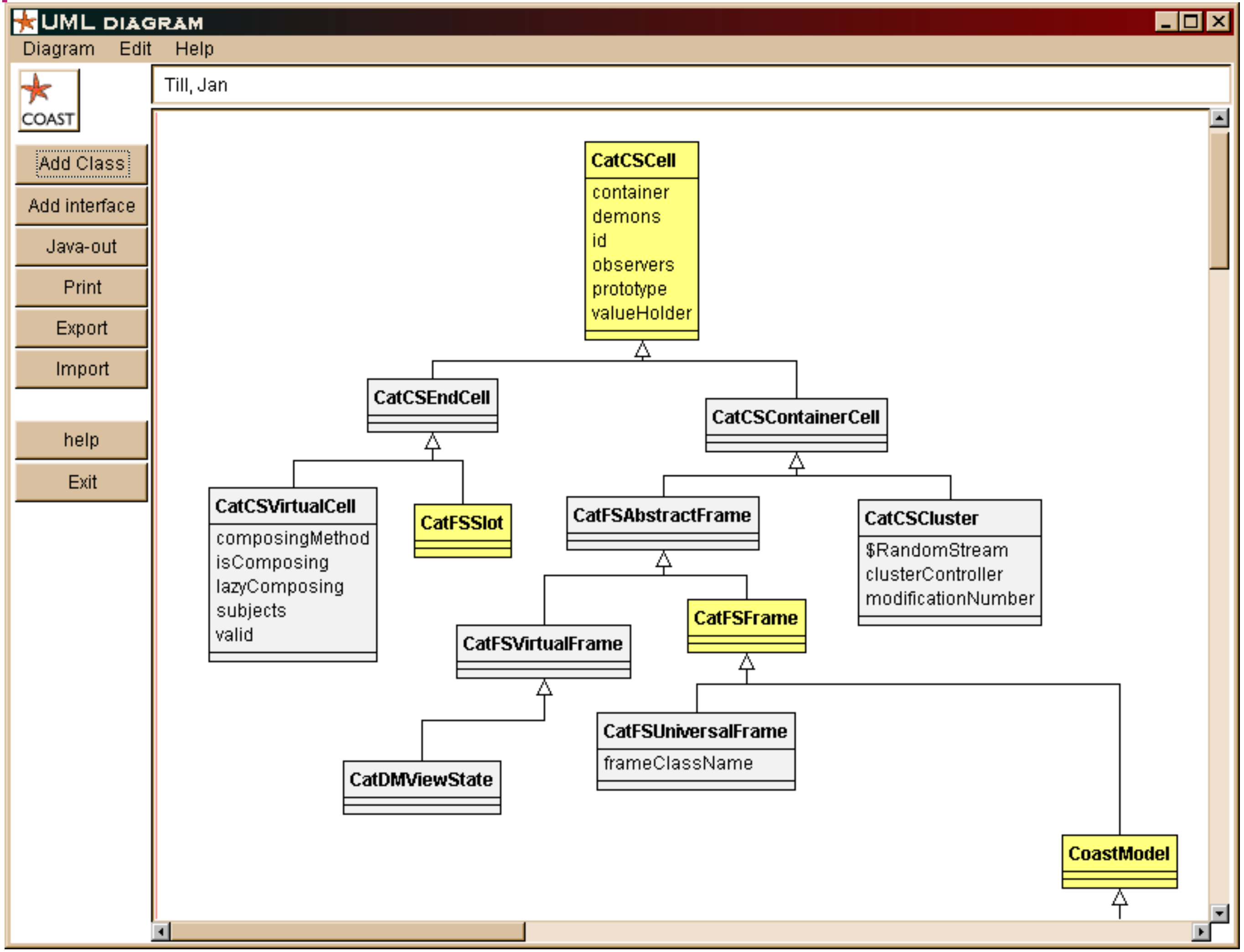
realization

usage experiences

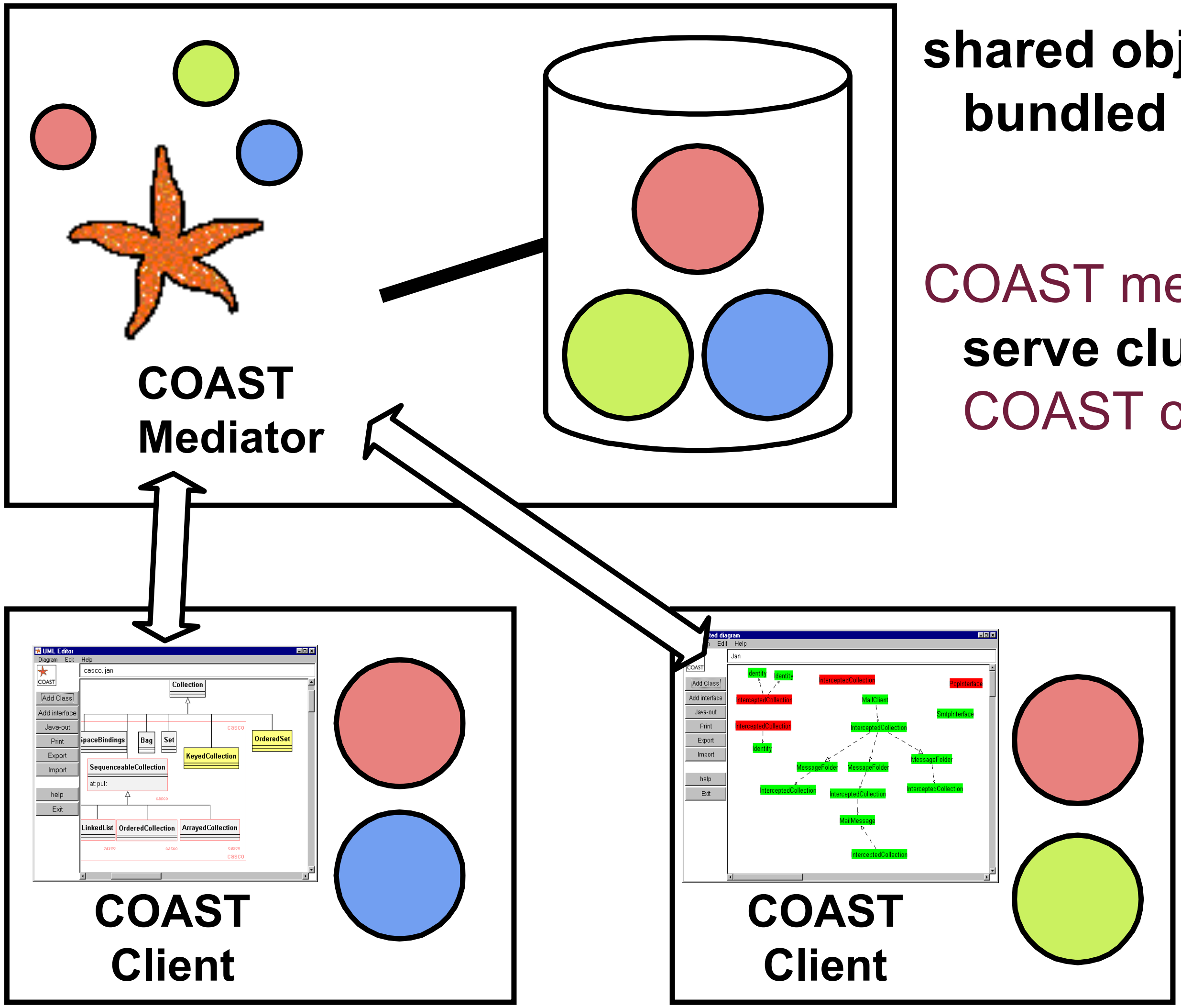
system architecture



COAST class hierarchy



shared data management



shared objects are bundled in clusters

COAST mediators **serve clusters to** COAST clients

motivation

requirements

application structure

groupware model

realization

usage experiences

transactions

shared objects are modified in transactions

- prevent inconsistencies
- short transactions
- optimistic or pessimistic
- ACID properties

transaction processing

- local execution
- local commit
- send agenda to mediator
- global commit / reject
- broadcast changes to synchronize replica

view updating

virtual slots

- cache computation results
- computation
  - on demand (lazy)
  - on invalidation (eager)
- automatic invalidation
- dependencies between model and virtual slots are detected by the framework

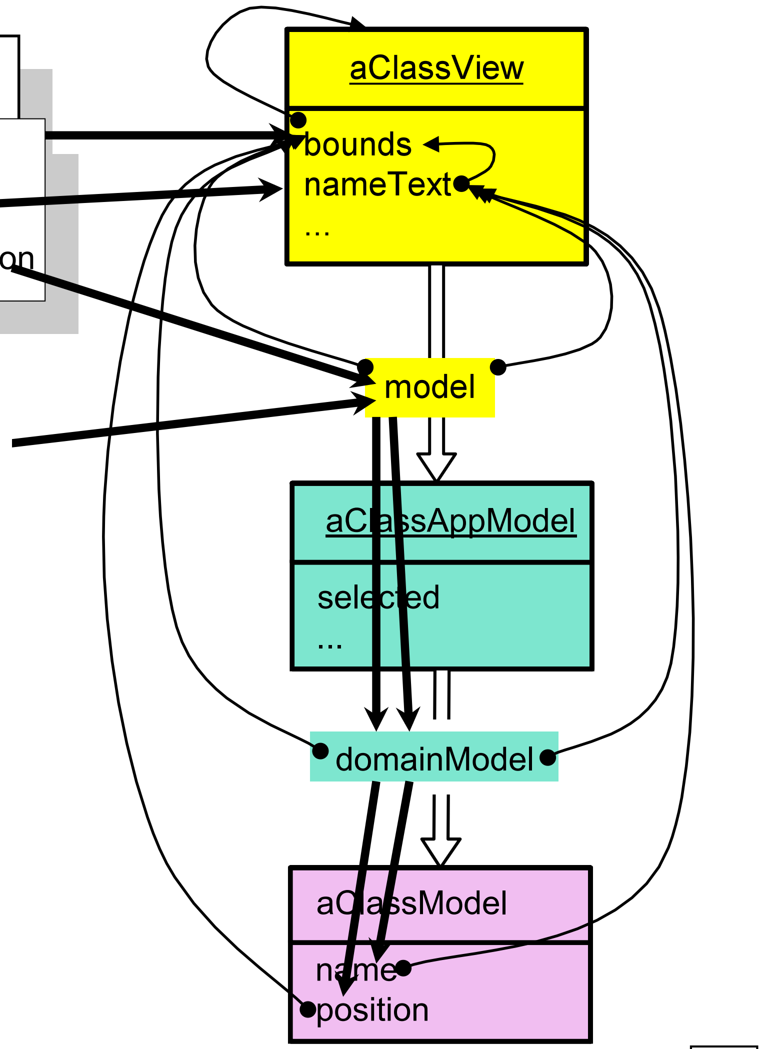
views have virtual slots that trigger redisplay

motivation
requirements
application structure
groupware model
realization
usage experiences

virtual slots: computation

```
composeDisplayOn: aGraphicsContext
    aGraphicsContext paint: ColorValue gray.
```

```
computeBounds
    | boundingRect |
    boundingRect := self nameText bounds.
    ^boundingRect translatedBy: self model domainModel position
```



motivation

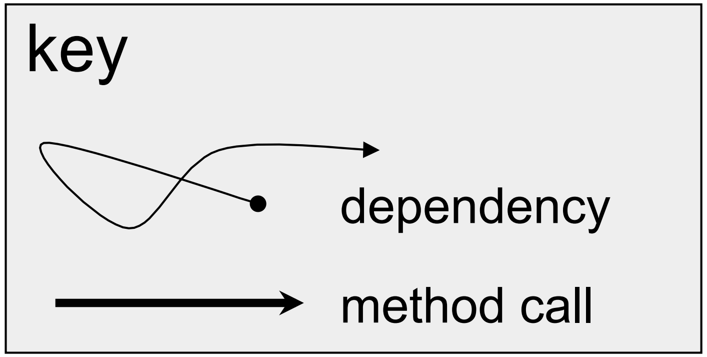
requirements

application structure

groupware model

realization

usage experiences



view updating

constraint mechanism ensures display consistency

display updating integrated into transaction scheme

- invalidation phase: accumulate display damage
- updating phase: repair display damage

motivation

requirements

application structure

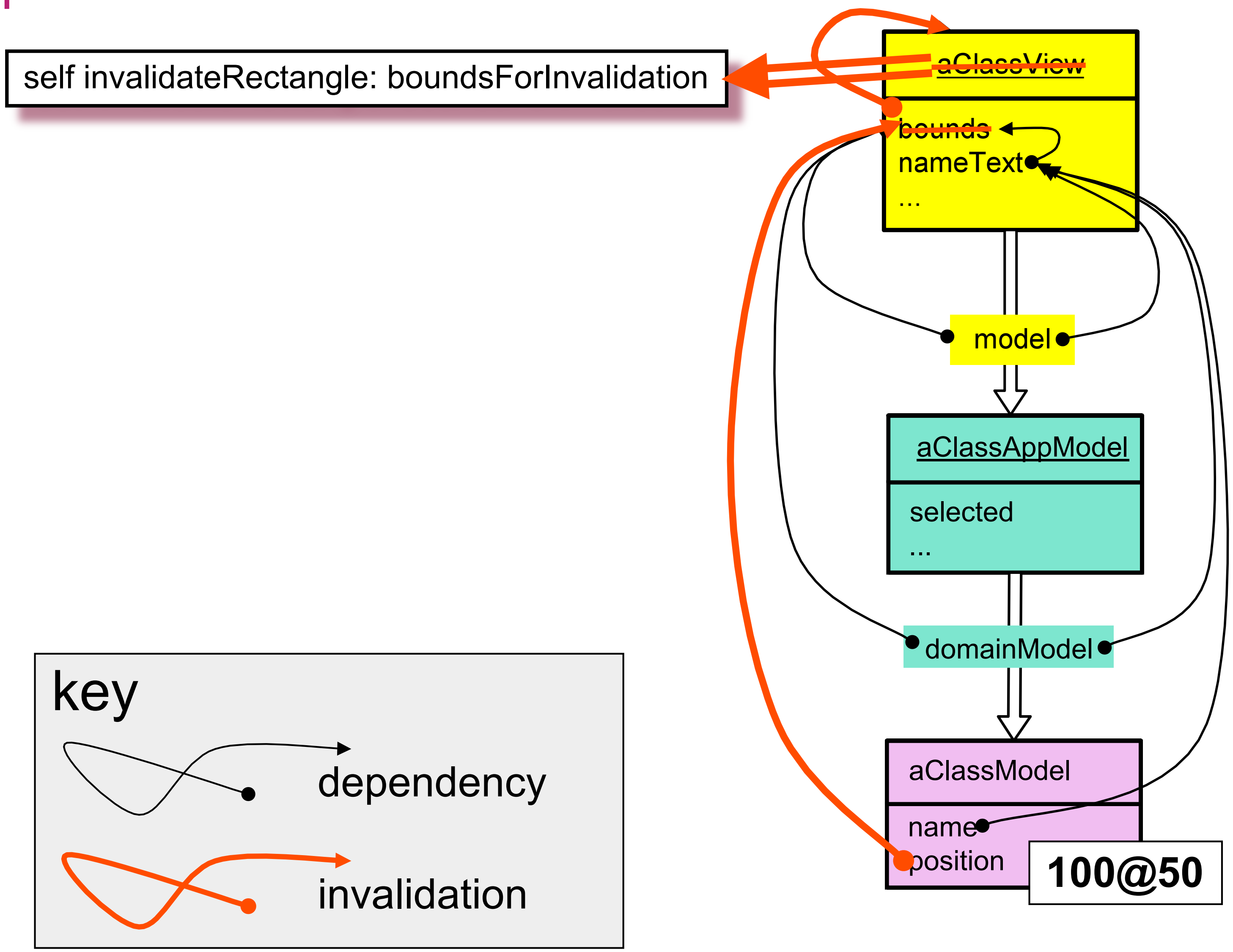
groupware model

realization

usage experiences

COAST

view updating: change notification



```
self invalidateRectangle: boundsForInvalidation
```

motivation

requirements

application structure

groupware model

realization

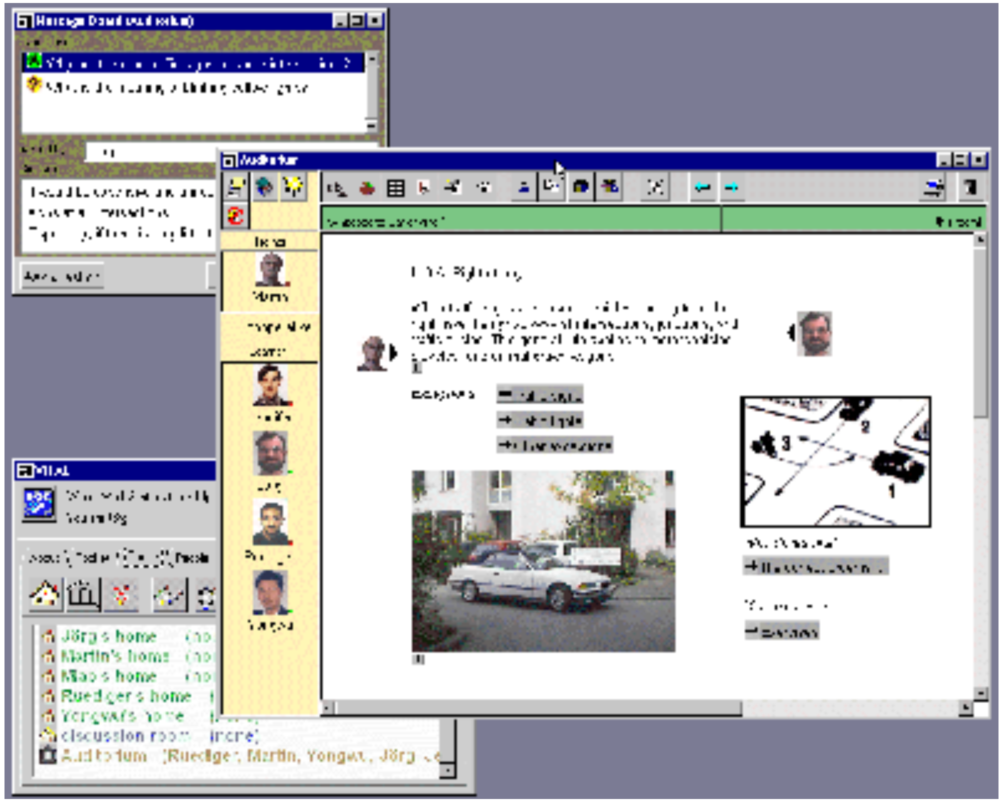
usage experiences

GMD IPSI

applications - learning

VITAL

CROCODILE



motivation

requirements

application structure

groupware model

realization

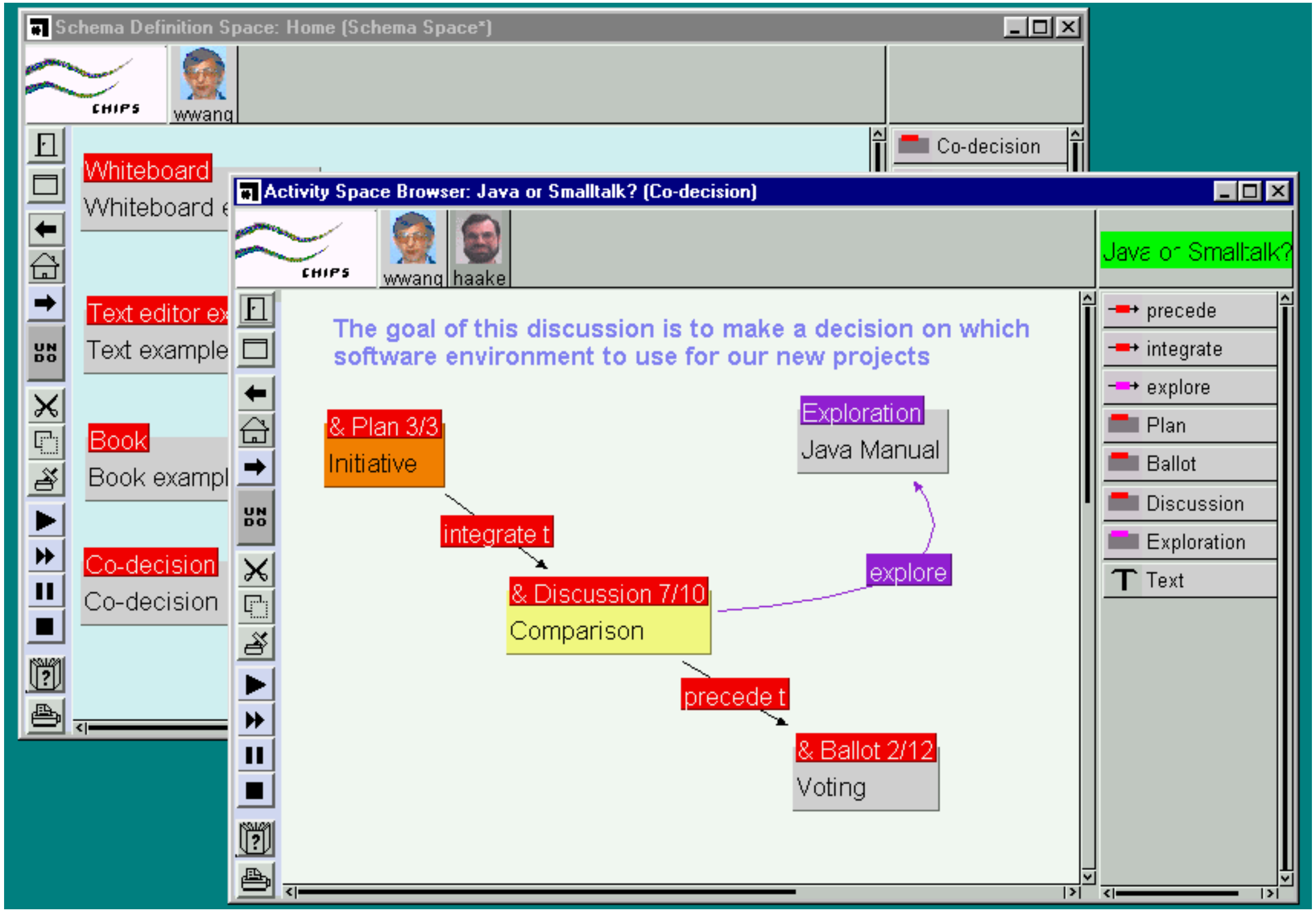
usage experiences

COAST

applications - process modelling

GMD IPSI

CHIPS



The goal of this discussion is to make a decision on which software environment to use for our new projects

motivation

requirements

application structure

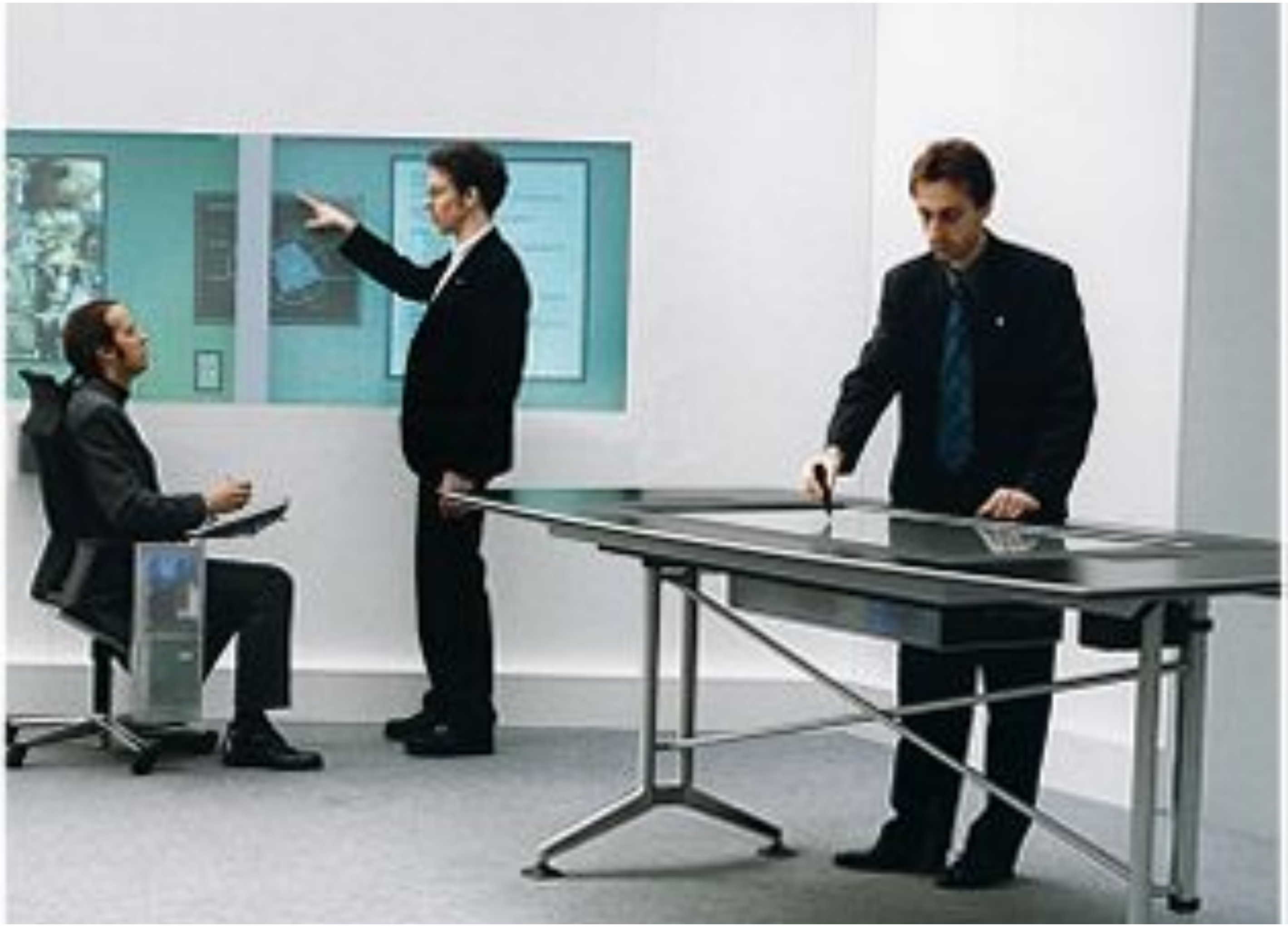
groupware model

realization

usage experiences

applications - roomware

Beach



motivation

requirements

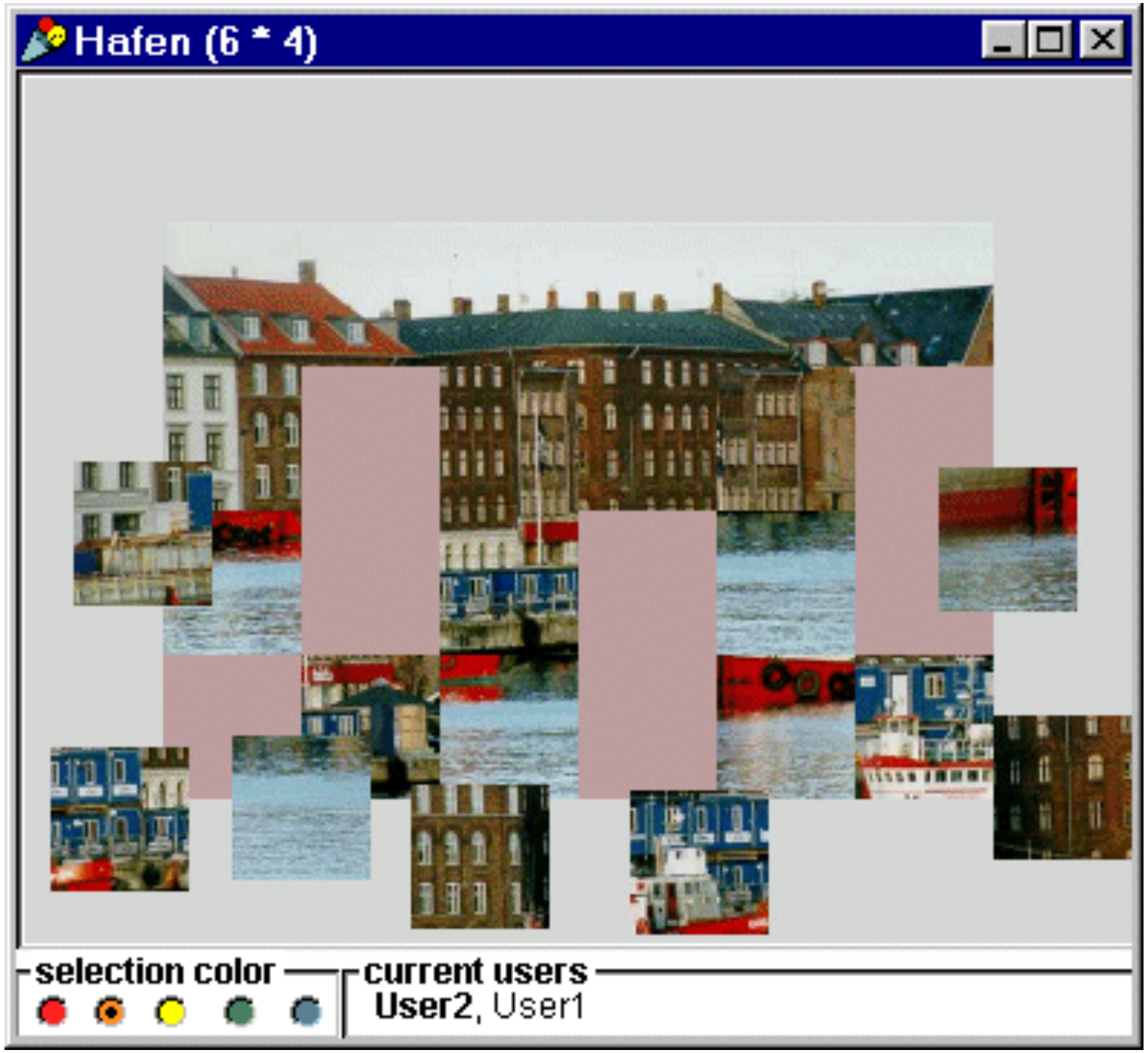
application structure

groupware model

realization

usage experiences

Co-operative Puzzle

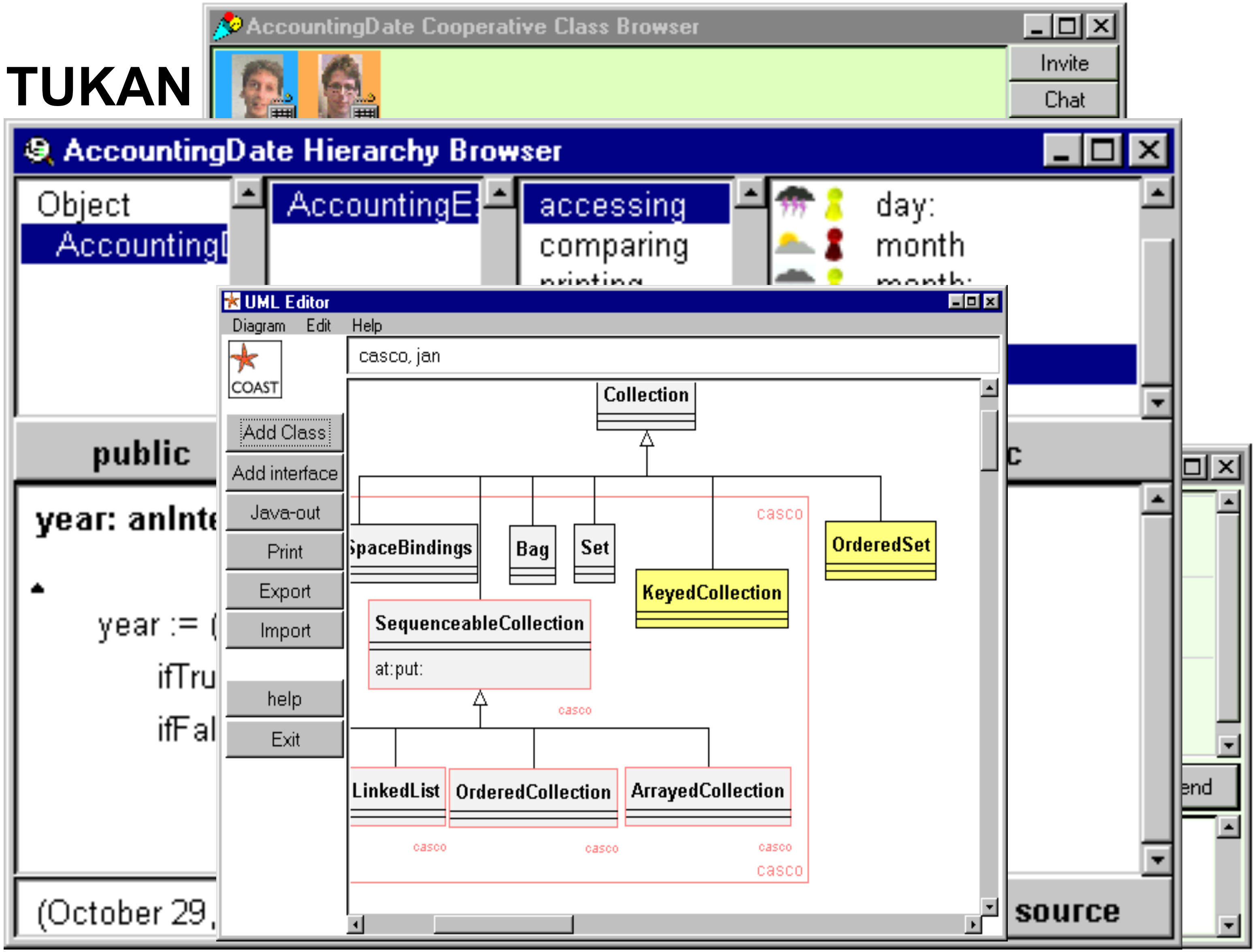


COAST

applications - software dev.

GMD IPSI

TUKAN



motivation

requirements

application
structure

groupware
model

realization

**usage
experiences**

usage experiences 1

performance

- COAST-applications are as fast as comparable single user applications

size of the shared object space

- up to now, a maximum of 30.000 was reached

number of users

- VITAL was tested with up to 12 simulanousley working users

usage experiences 2

network connection

- low bandwidth for synchronisation of replica
- initial effort for replication
- VITAL tested via 28.800 Bps modem connection
- UML-editor tested between Germany and Argentina

development effort for COAST applications

- learning effort for newbys
- experienced developers
  - one week for first version of UML editor
  - one weekend for the collaborative puzzle

motivation

requirements

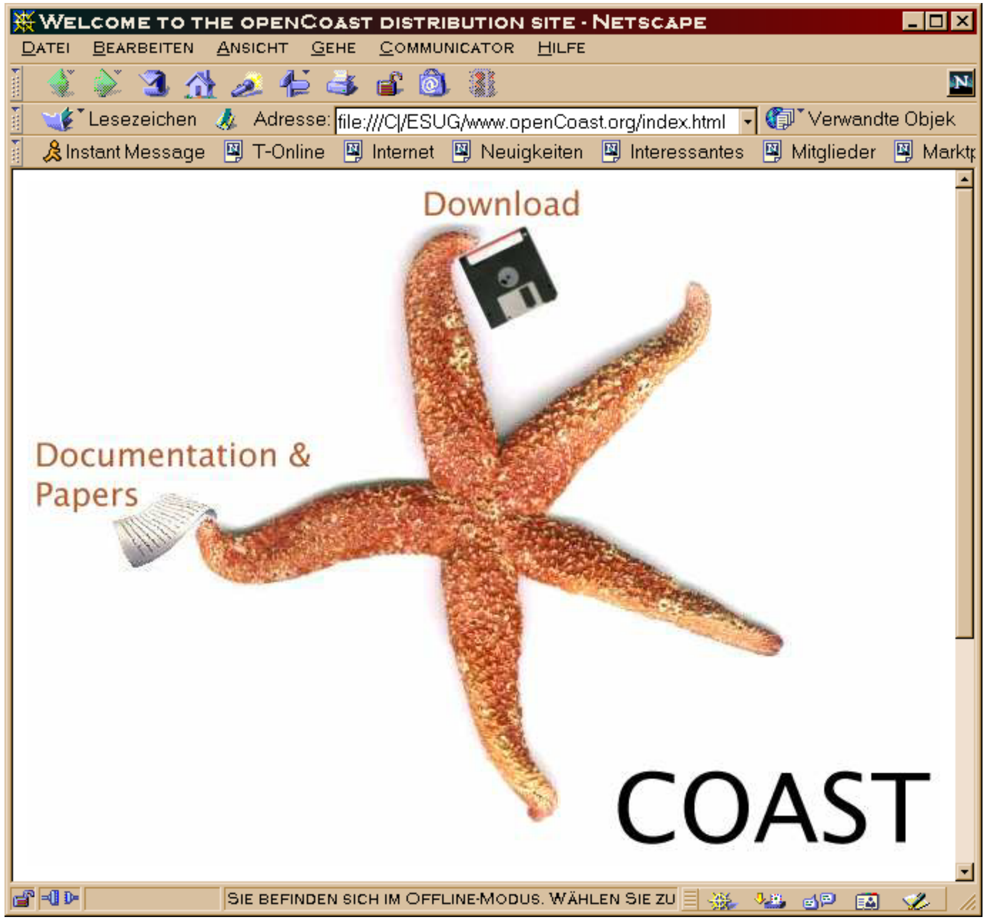
application structure

groupware model

realization

usage experiences

further info & download



Welcome to the openCoast distribution site - Netscape

Datei Bearbeiten Ansicht Gehe Communicator Hilfe
Lesezeichen
Adresse: file:///C|/ESUG/www.openCoast.org/index.html
Verwandte Objek
Instant Message
T-Online
Internet
Neuigkeiten
Interessantes
Mitglieder
Marktp
Download
Documentation & Papers
COAST
Sie befinden sich im Offline-Modus. Wählen Sie zu

www.openCoast.org