

Groupware and Collaborative Interaction
Distributed Interactive Systems

Technical aspects

M2R Interaction / Université Paris-Sud / 2019 - 2020

Cédric Fleury (cedric.fleury@lri.fr)

Introduction

Technical aspects of distributed interactive systems

Requirements redundant for all CSCW applications

Network architecture

Data distribution

Concurrency management

Collaborative virtual environment is a good example

Strong requirements

Users are interacting in real-time

Immersion requires fast multi-sensorial feedbacks

Lots of solutions to overcome the technical issues

Collaborative Virtual Environments

Enable users to work or have fun together

2 kinds of collaboration in virtual environment

Co-located collaboration

Remote collaboration

Aspects of collaboration

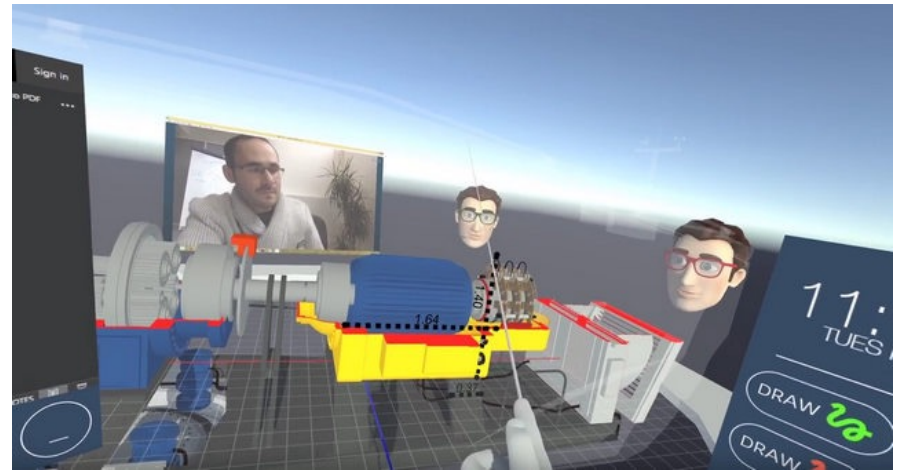
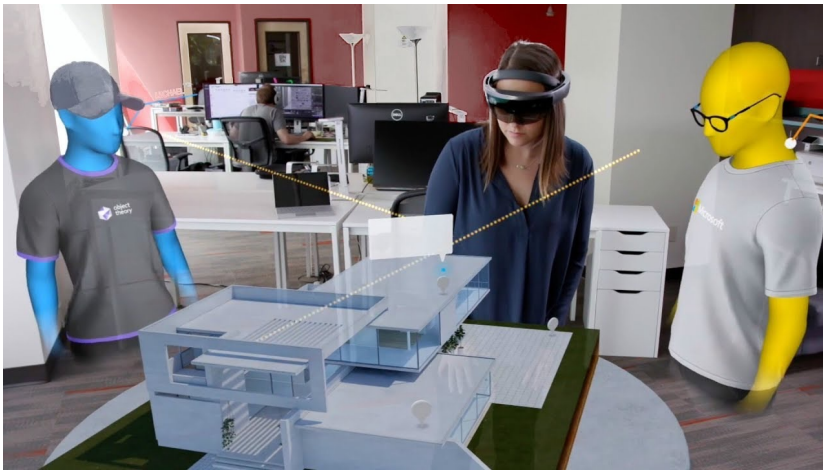
Awareness

Communications

Collaborative interaction



Collaborative Virtual Environments



Share the same virtual objects

- 3D objects (with shape, texture, color, position, etc.)

- 3D widgets (3D objects which can be used for interaction)

- Annotations

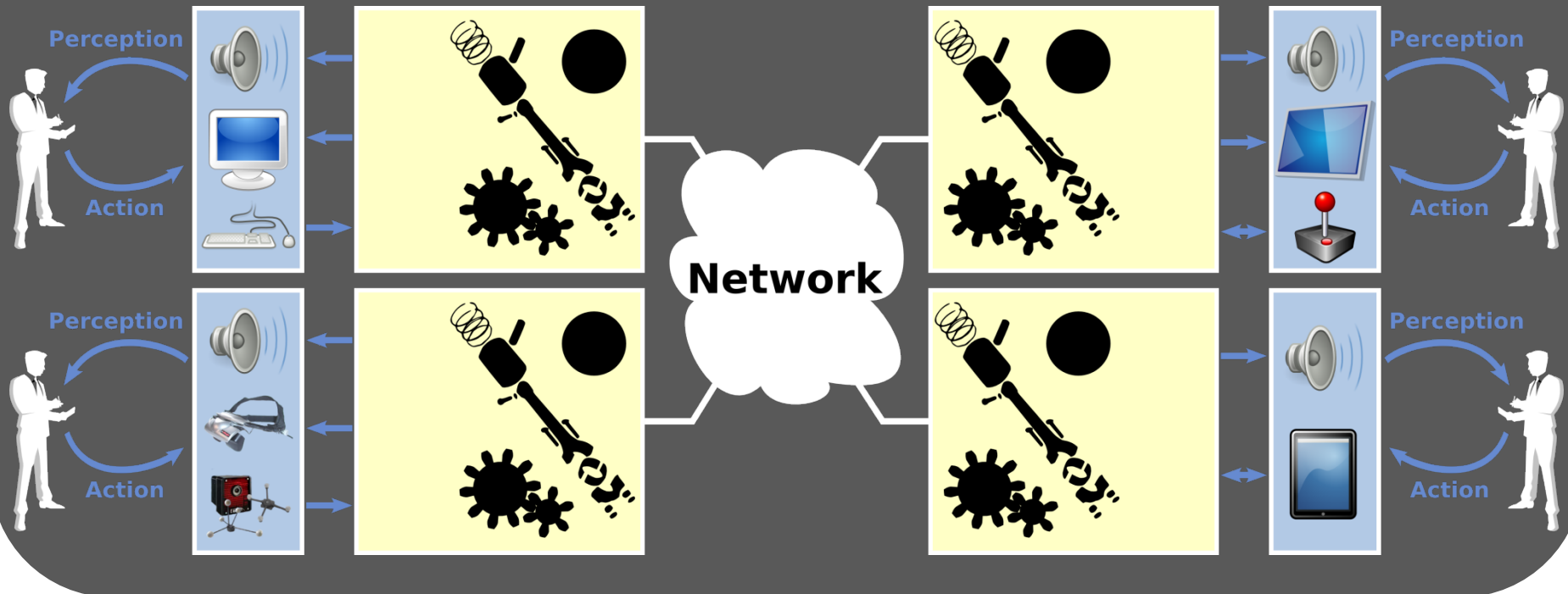
- Interaction tools (virtual ray of the others, etc.)

- Avatars

Need to interact together in real-time

Remote Collaboration

Distributed virtual environment



Collaboration requirements

For efficient collaboration, users need to:

Have the same state of the virtual environment (virtual objects) at the same time

⇒ Consistency of the VE

Modify the virtual objects in real-time

⇒ Responsiveness of the system (interactivity)

Consistency

[Delaney et al., 2006]

Distributed virtual environment

Distributed database of virtual objects with users modifying it in real-time

Manage the consistency

Ensure that the database is the same for all users

Inconsistencies due to:

Concurrent modifications

Delay to transmit modification on the network

Responsiveness

[Delaney et al., 2006]

Responsiveness of the system

Time required to respond to users' actions
(latency during users' interaction, jitter)

Due to the time required to:

Process and send users' actions

Transmit actions on the network (if mandatory)

Give a feedback to the users

Between 40ms and 300ms, under 100ms is good

Distributed Virtual Environments

Find a **good trade-off** between consistency and responsiveness (task, application, etc.)

Technical requirements

- Connect remote computers

- Distribute data

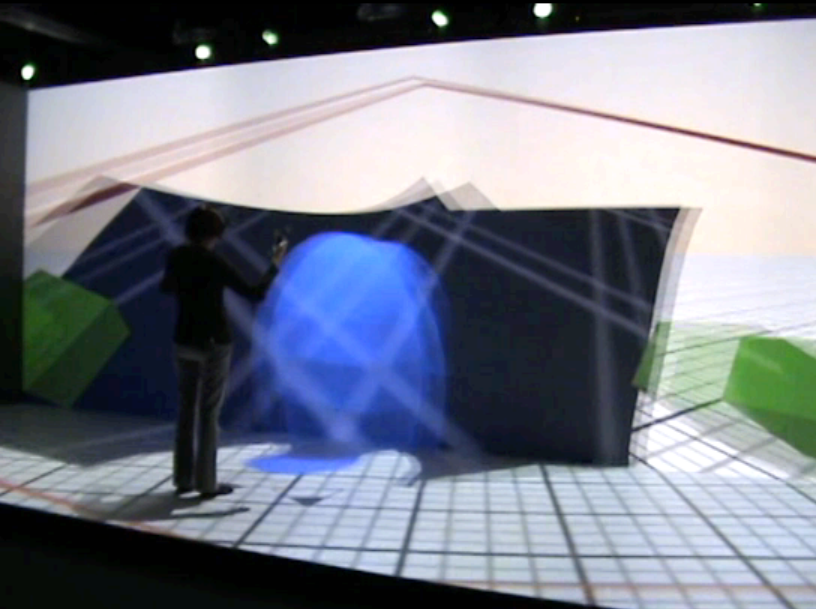
- Share information

- Manage concurrent accesses to the data

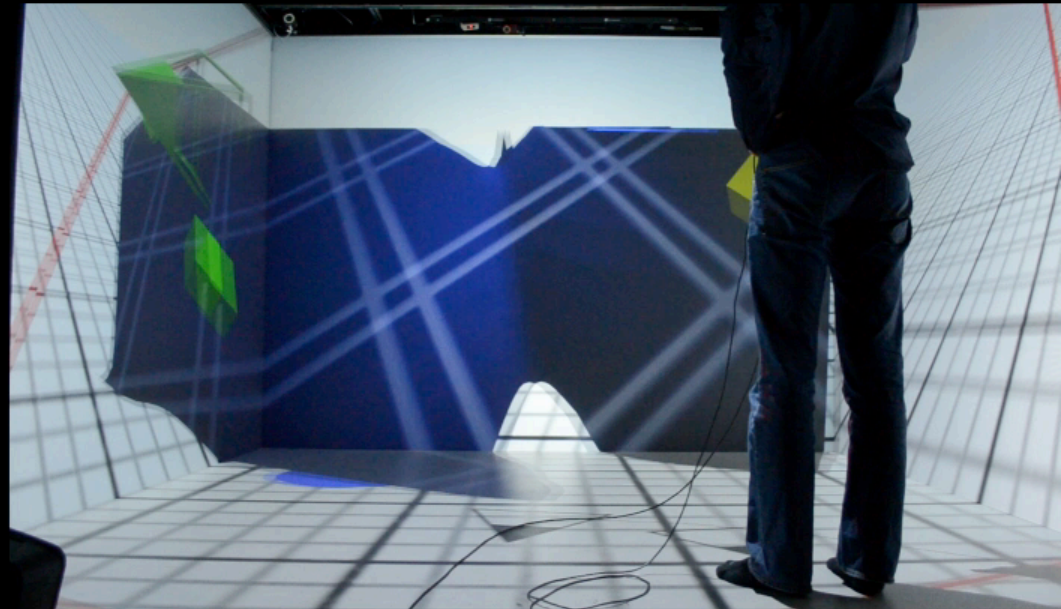
=> Each technical choice must consider both consistency and responsiveness

Example

Rennes
(INSA / IRISA)



London
(UCL)



Outline

Network Architecture

Data Distribution

Communication Protocols

Communication Reduction Mechanisms

Consistency Management Mechanisms

Software architecture

Outline

Network Architecture

Data Distribution

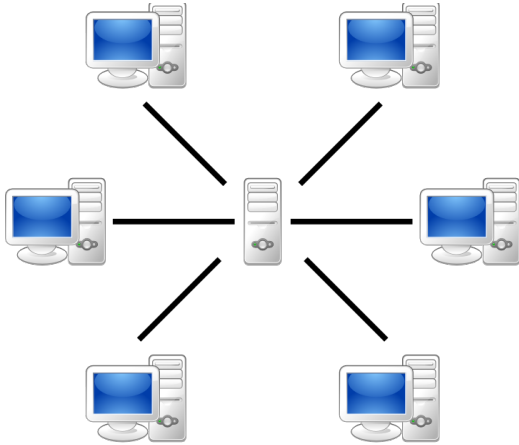
Communication Protocols

Communication Reduction Mechanisms

Consistency Management Mechanisms

Software architecture

Network Architecture



Client/server architecture

[Vistel 95, RING 95, BrickNet 95, ShareX3D 08]

All communications pass through the server

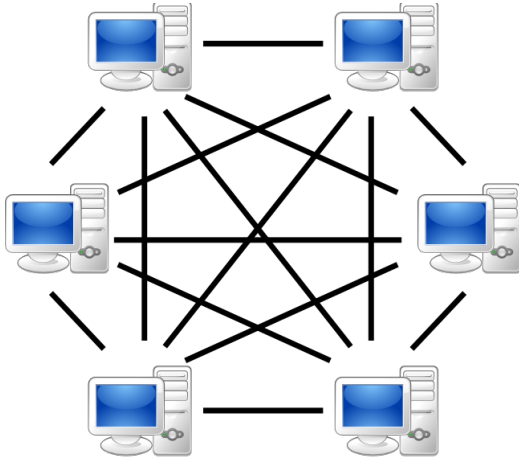
latency during interactions

All nodes can be contacted quickly

Consistency and synchronization are easy to ensure

A “bottleneck” can occur on the server

Network Architecture



Peer-to-peer architecture

[Reality Build for Two 90, MR Toolkit 93, SIMNET 93, NPSNET 94]

Fast communications between pairs of nodes

Closely coupled interactions between a few users

Difficulties to contact all nodes at the same time

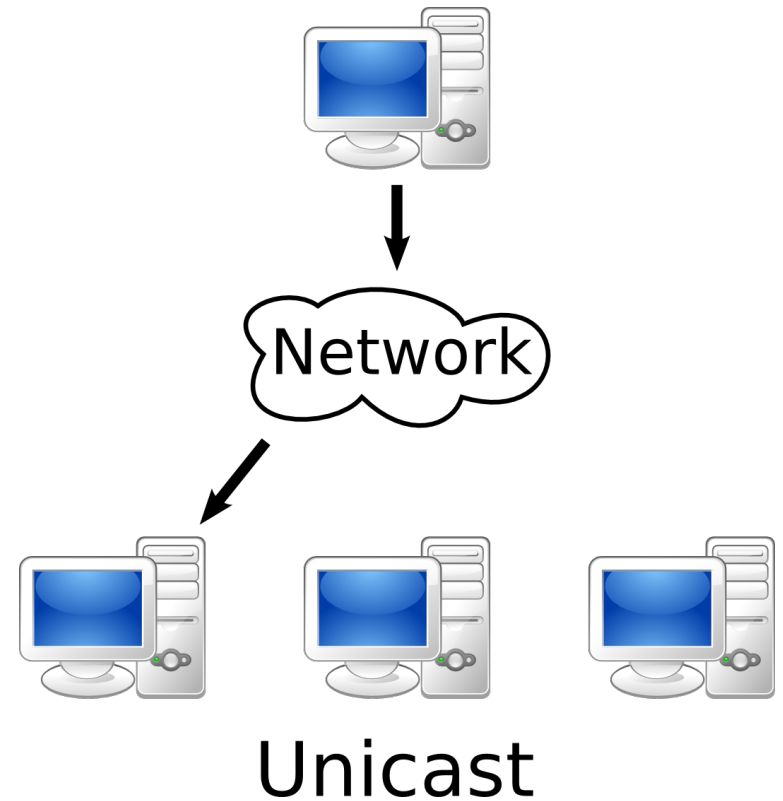
Consistency and synchronization are hard to ensure

Many messages are transmitted over the network

Network Architecture

Transmission Methods

Unicast

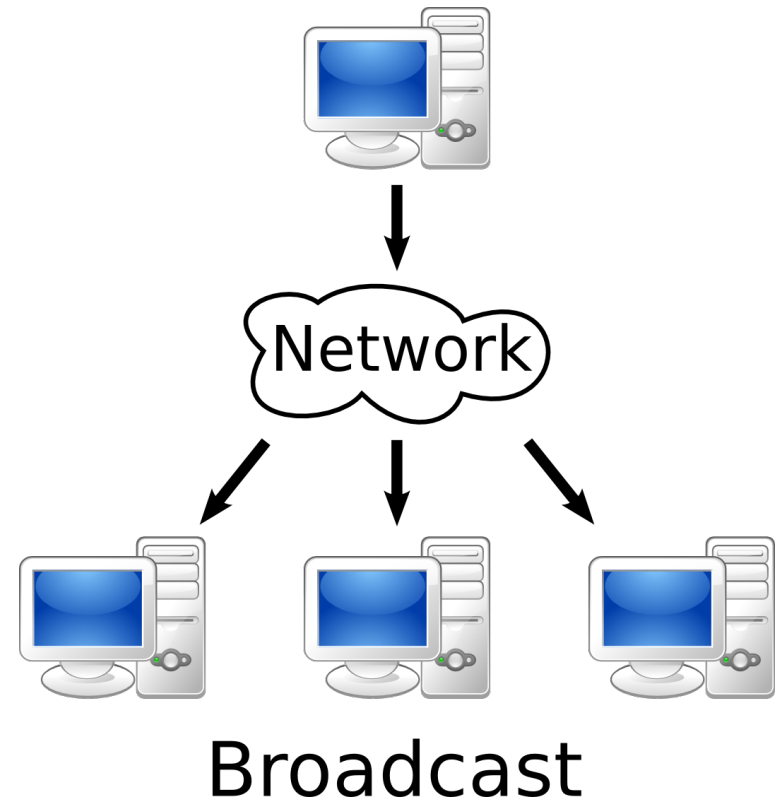


Network Architecture

Transmission Methods

Unicast

Broadcast



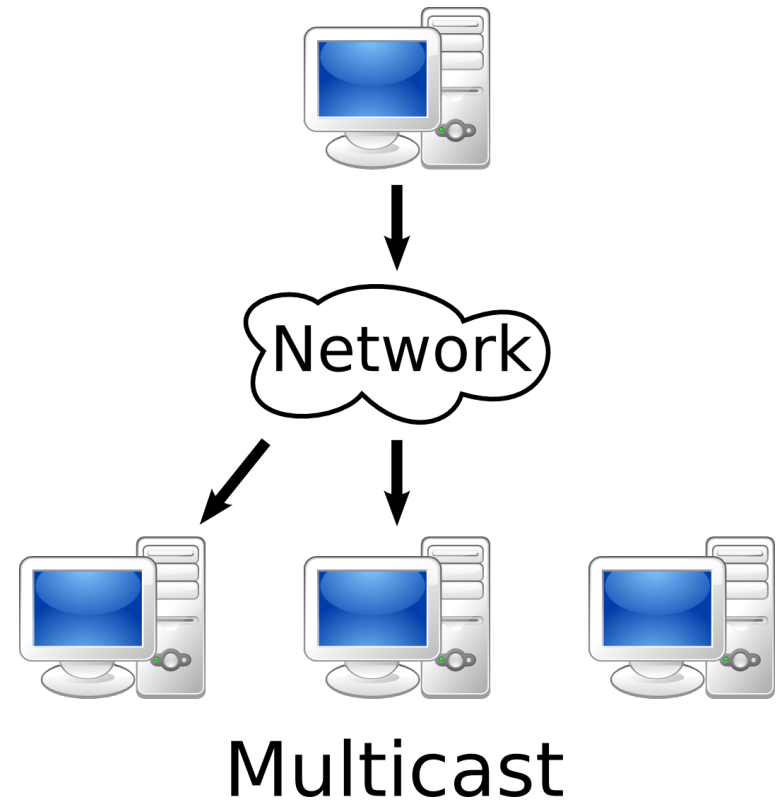
Network Architecture

Transmission Methods

Unicast

Broadcast

Multicast



Network Architecture

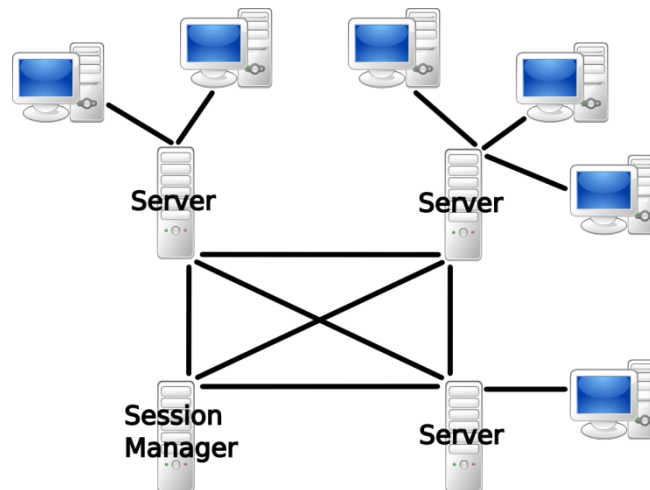
Hybrid architecture

Servers connected with peer-to-peer connections [SPLINE 97]

Avoids the “bottleneck” on a single server

Connects nodes with specific requirements

Increases system latency



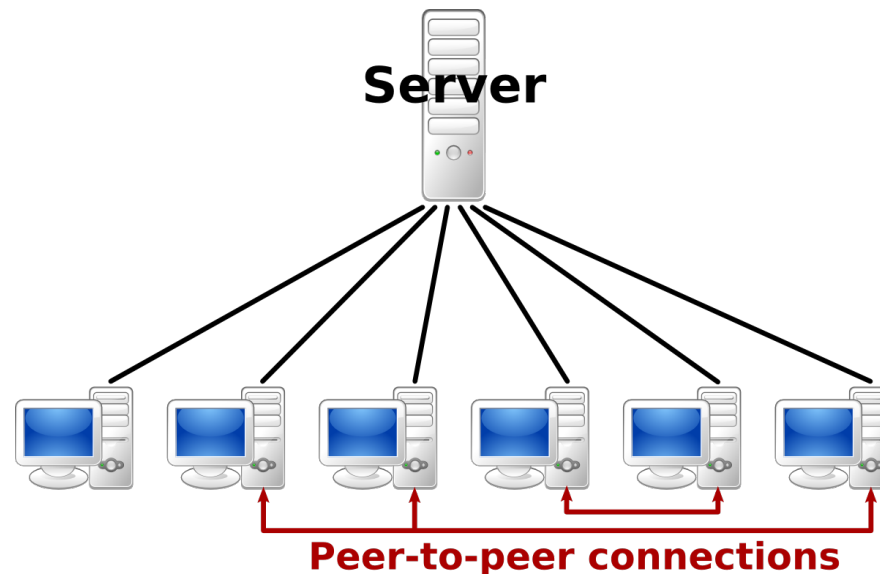
Network Architecture

Hybrid architecture

Temporary peer-to-peer connections [Anthes et al., 04]

Are established according to users' locations in the VE

Increase CVE consistency between nearby users



Outline

Network Architecture

Data Distribution

Communication Protocols

Communication Reduction Mechanisms

Consistency Management Mechanisms

Software architecture

Data Distribution

A virtual object

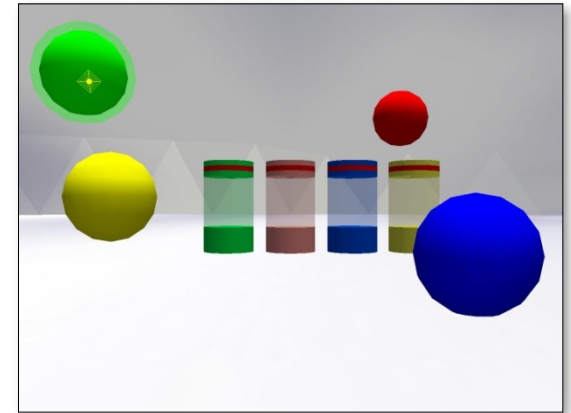
A set of parameters (data)

Identifier

Attributes (position, orientation, etc.)

User access rights

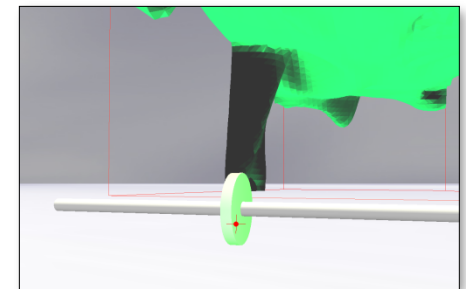
Geometry, and eventually textures



A behavior

Only reactive (responding to user actions)

Continuous (evolving in the time)

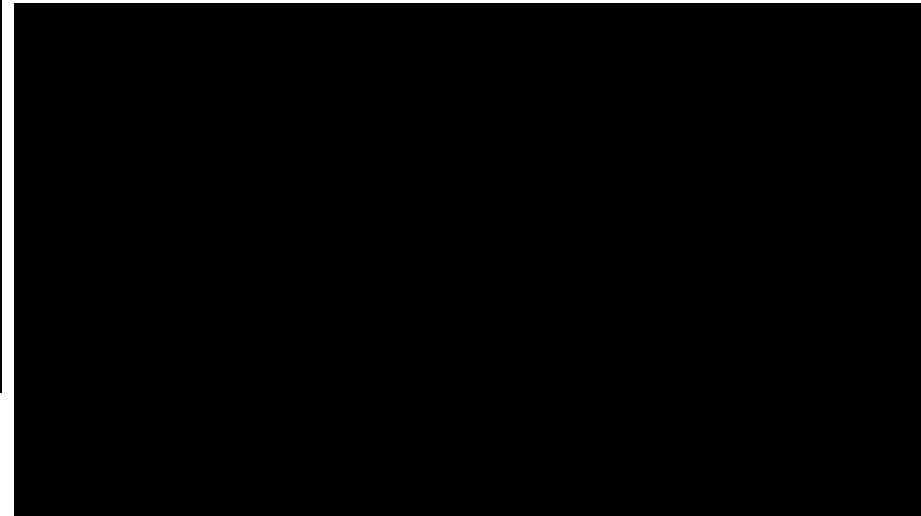


⇒ Which computers store the data?

⇒ Which computers manage the data modification?

⇒ Which computers execute the behavior?

Examples of continuous behaviors

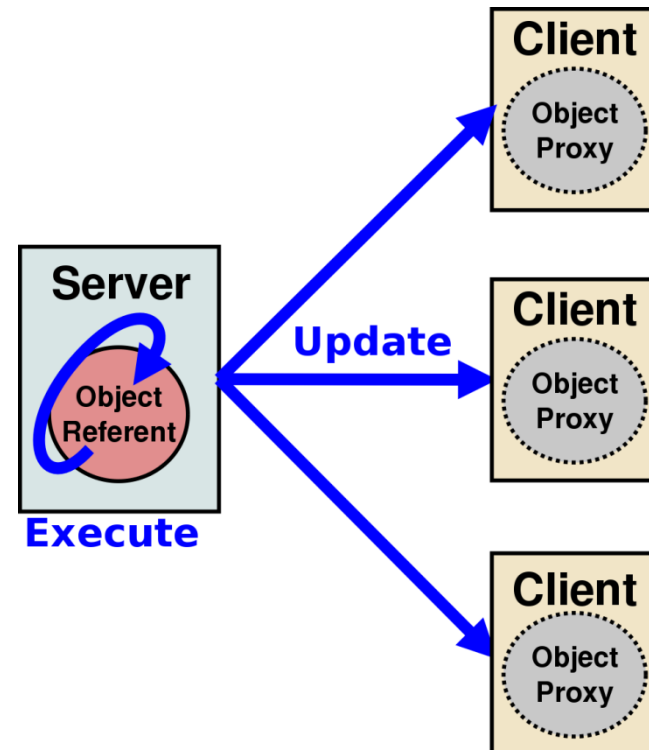


Data Distribution

Centralized [Vistel 95]

Data is stored on the server

Behaviors are executed on the server



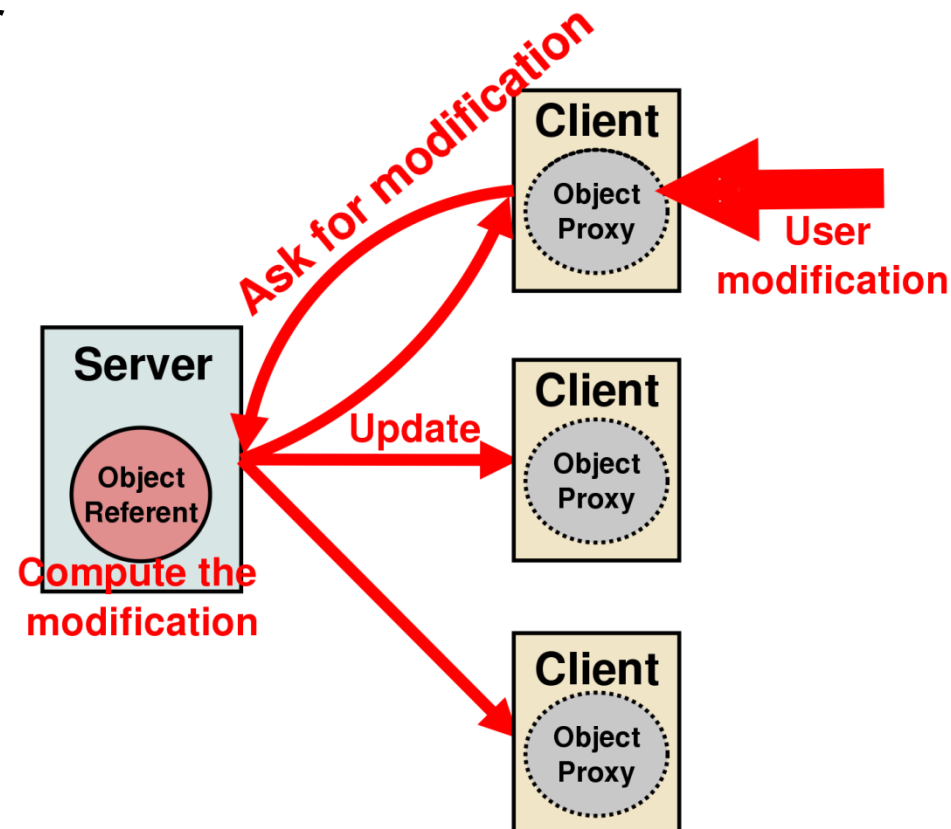
Data Distribution

Centralized [Vistel 95]

Data is stored on the server

Behaviors are executed on the server

Modification requests are processed on the server



Data Distribution

Centralized [Vistel 95]

Advantages

Ensures a global consistency

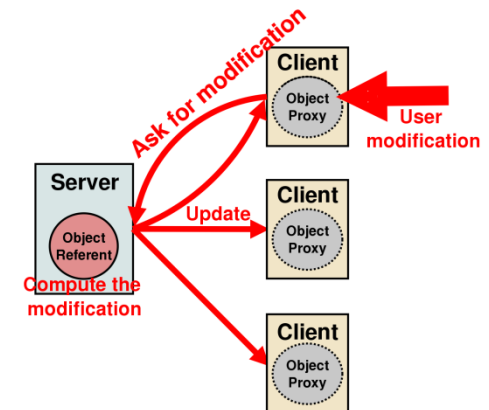
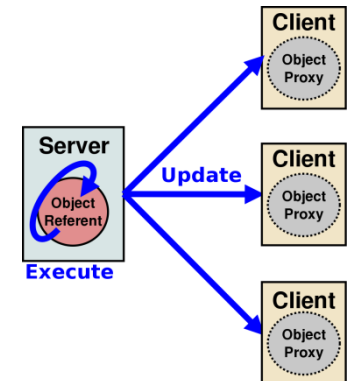
Avoids data replication

Avoids behaviors processing on the clients

Drawbacks

Introduces latency during interactions

Transmits many messages over the network



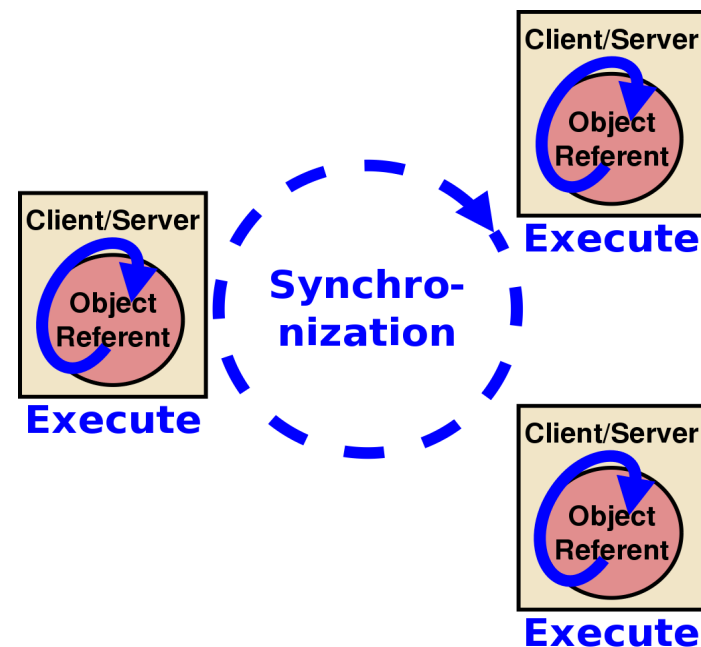
Data Distribution

Replicated [SIMNET 93, MR Toolkit 93]

Data is replicated on each node

Synchronization between nodes can be achieved

Behaviors are executed on each node



Data Distribution

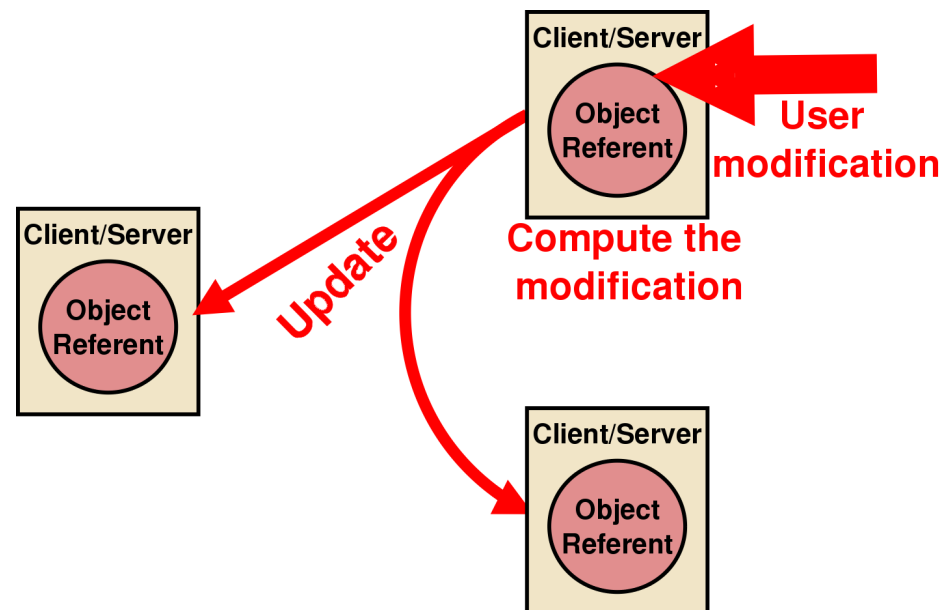
Replicated [SIMNET 93, MR Toolkit 93]

Data is replicated on each node

Synchronization between nodes can be achieved

Behaviors are executed on each node

Modification requests are processed locally



Data Distribution

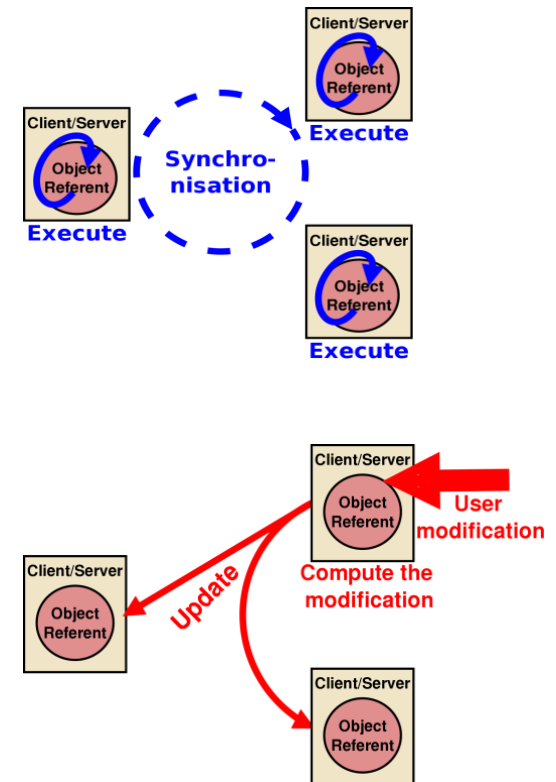
Replicated [SIMNET 93, MR Toolkit 93]

Advantages

- Low-latency interactions
- Few messages transmitted

Drawbacks

- Data replication
- Behaviors processed on each node
- Inconsistencies due to transmission delay of update messages
- Additional mechanisms for managing concurrent accesses



Data Distribution

Hybrid [DIVE 98] [BrickNet 98]

Only the necessary objects are replicated

A server saves the whole VE state

Advantages

Reduction of data replication

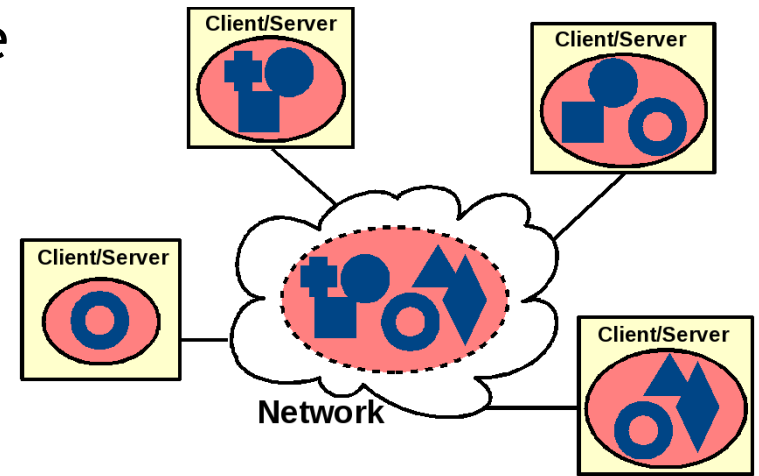
Less processing on each node

Drawbacks

Difficulties to ensure consistency and manage concurrency

Many messages transmitted over the network

Dynamic downloads of additional objects



Data Distribution

Hybrid: Referent/proxy paradigm

[OpenMASK 02][Schmalstieg et al 03][Fleury et al 10]

On a node each virtual object is represented by

A referent

Stores data

Defines behavior

Processes modification requests

or

A proxy

Receives updates from referents

Updates object representation in the CVE

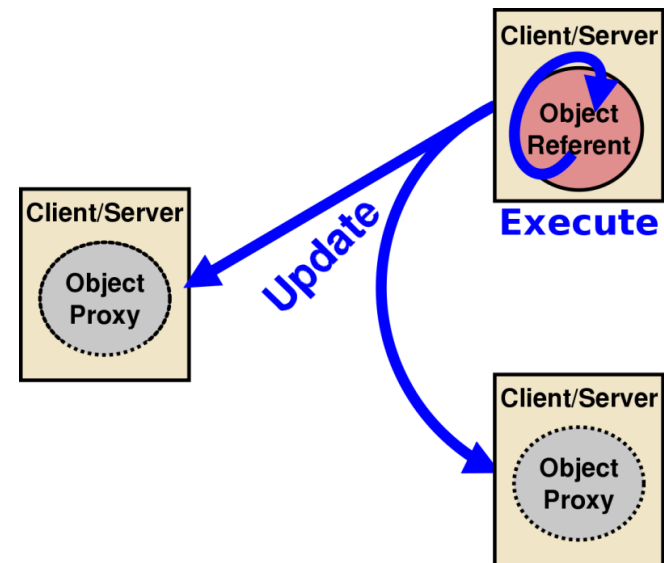
Can store copy of the data (for easy migration)

Data Distribution

Hybrid: Referent/proxy paradigm

[OpenMASK 02][Schmalstieg et al 03][Fleury et al 10]

Behaviors are executed only on one node



Data Distribution

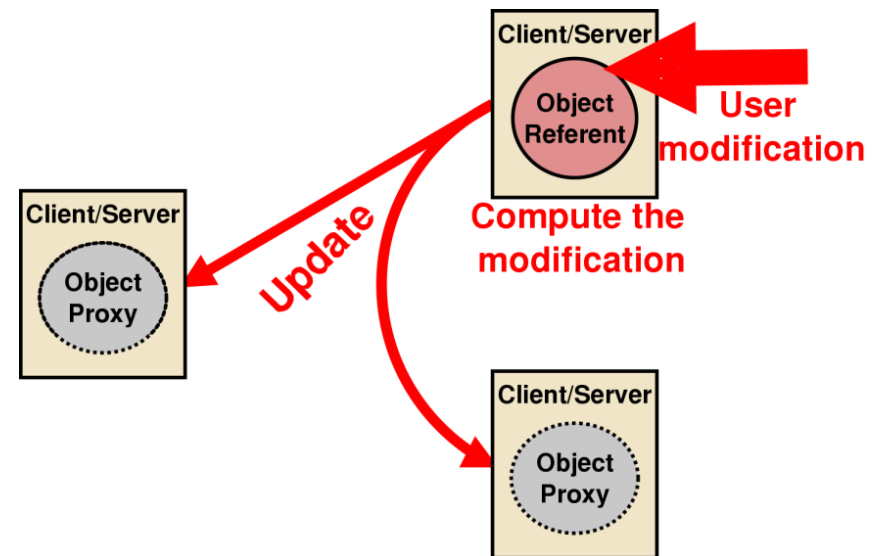
Hybrid: Referent/proxy paradigm

[OpenMASK 02][Schmalstieg et al 03][Fleury et al 10]

Behaviors are executed only on one node

Referent modification

Modification requests are processed locally



Data Distribution

Hybrid: Referent/proxy paradigm

[OpenMASK 02][Schmalstieg et al 03][Fleury et al 10]

Behaviors are executed only on one node

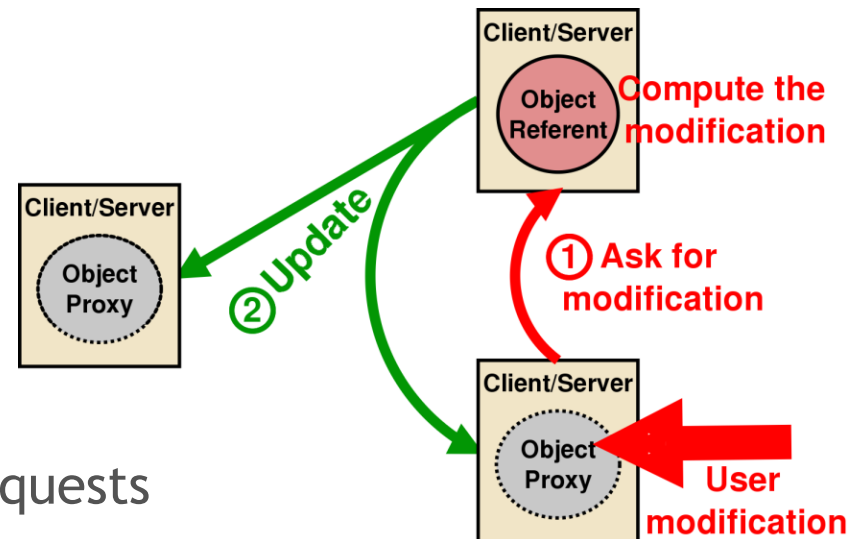
Referent modification

Modification requests are processed locally

Proxy modification

Modification requests are transmitted to the referent

The referent processes the requests



Data distribution \neq network architecture

Does not have to be the same !

Centralized data dist. requires a server

Replicated data dist. can be used with client/server arch.

Ex: video games

Centralized data dist. with peer-to-peer connection

Synthesis

Existing data distribution solutions [Fleury et al 10]

- Make a trade-off between consistency and responsiveness

- Meet particular requirements

Combine the advantages of each solution

- Dynamically adapt data distribution of each object

 - Application requirements, network capabilities

 - Tasks performed by users

 - Functions that objects fulfill in the VE

An adaptive data distribution

[Fleury et al., 2010]

Based on a referent/proxy paradigm

Three modes of data distribution

Centralized

Replicated

Hybrid

Chosen independently for each object

Changed dynamically during a working session

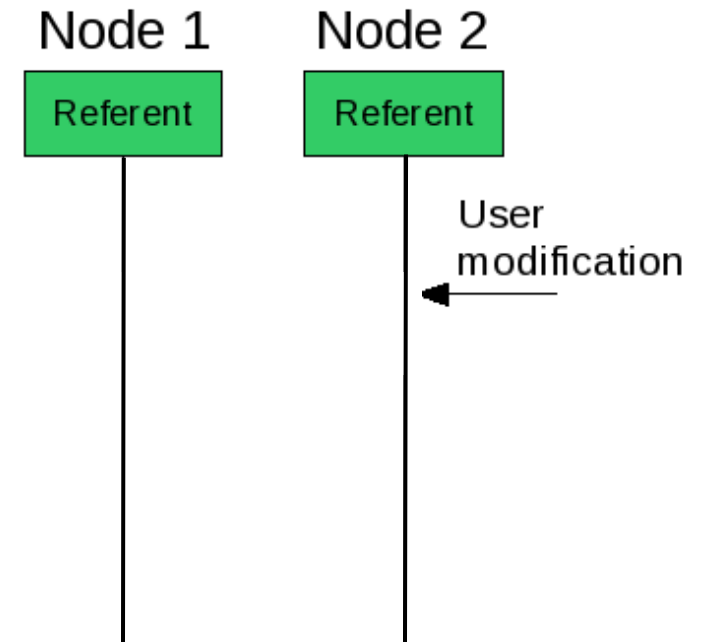
3 Modes of Distribution

Replicated Mode

Referents on all nodes

Interaction latency (IL)

Gap in consistency (GC)



⇒ Advantage: good responsiveness

3 Modes of Distribution

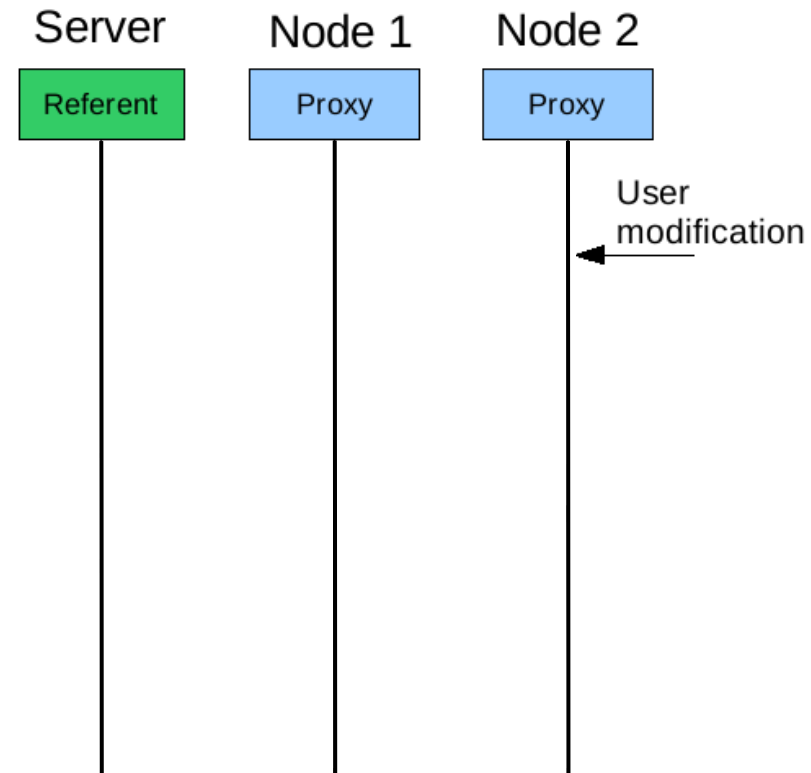
Centralized Mode

1 referent on the server

Proxies on other nodes

Interaction latency (IL)

Gap in consistency (GC)



⇒ Advantage: strong consistency

3 Modes of Distribution

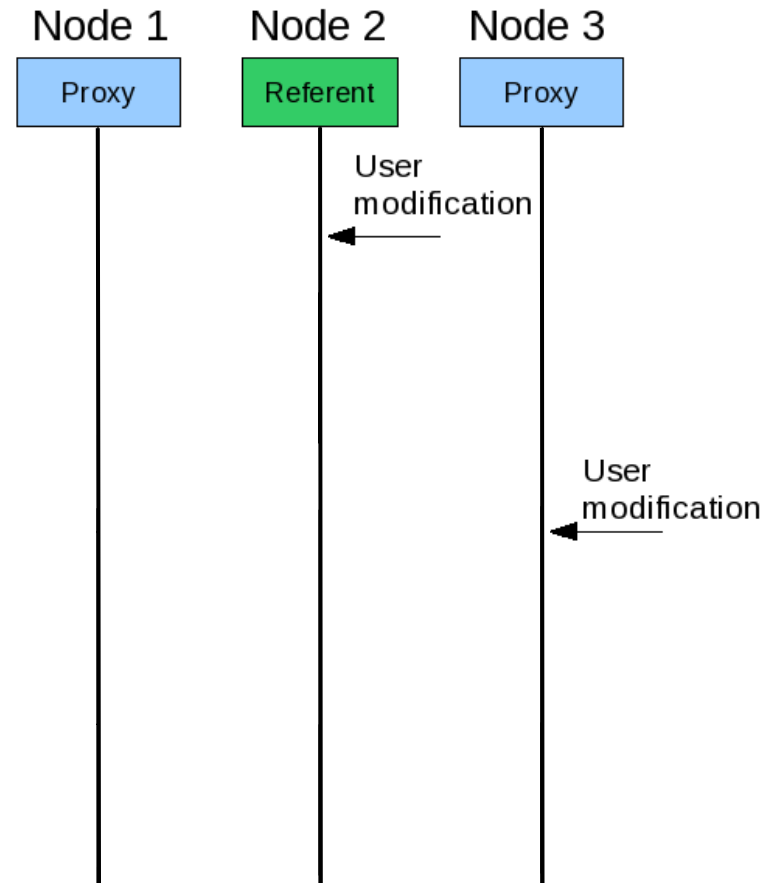
Hybrid Mode

1 referent on a node

Proxies on the other nodes

Interaction latency (IL)

Gap in consistency



⇒ Advantage: good tradeoff between responsiveness and consistency

Heterogenous data distribution

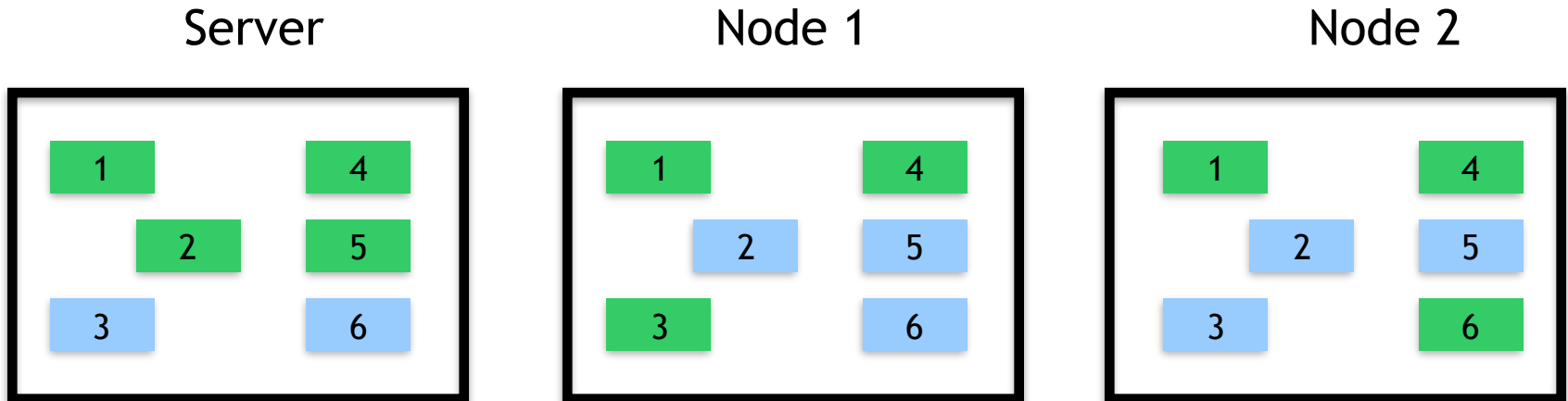


Different consistency/responsiveness trade-offs for each virtual object

Function fulfilled by objects

Precision requires to manipulate objects

Heterogenous data distribution



Choose the distribution mode at the object level

Each node can independently have

Referents for some objects

Proxies for some others

Each object can have a particular data distribution

Dynamical data distribution

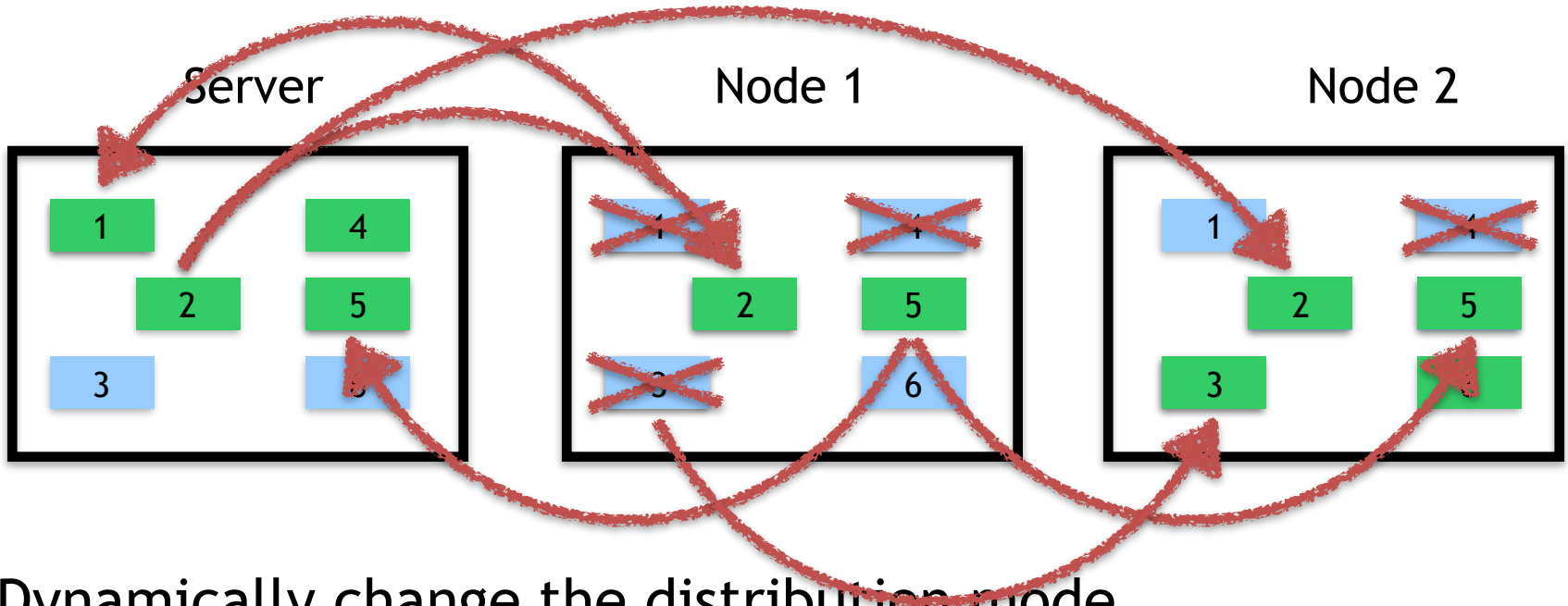


Adapt data distribution during a working session

Tasks that users perform in the VE

Network troubles

Dynamical data distribution



Dynamically change the distribution mode

Dynamically migrate the referent

Put the referent on the server (centralized mode)

Duplicate the referent on all nodes (replicated mode)

Move the referent from one node to another (hybrid mode)

Outline

Network Architecture

Data Distribution

Communication Protocols

Communication Reduction Mechanisms

Consistency Management Mechanisms

Software architecture

Communication Protocols

Classical protocols (TCP, UDP)

Multicast oriented protocols

Difficult to achieve over large network

Use additional network layers

“MBone” [DIVE 94, NPSNET 98]

Virtual Reality dedicated protocols

[RTP\I 99]: adapt RTP for interaction

[VRTP 97]: support VRML (virtual reality modeling language)

Some others [DWTP 98, DIS 93, HLA 97, ISTP 97]

Communication Protocols

Specific protocols in industrial environment

Deal with:

- Standard Internet access

- Firewalls that support only HTTP and HTTPS protocols

More generic standards start to be used

- VRPN (Virtual-Reality Peripheral Network)

- OSC (Open Sound Control)

Outline

Network Architecture

Data Distribution

Communication Protocols

Communication Reduction Mechanisms

Consistency Management Mechanisms

Software architecture

Communication Reduction Mechanisms

Avoid to overload the network

- Big number of users

- Low bandwidth network

Reduce the number of messages transmitted on the network without:

- Reducing the consistency

- Increasing the latency during interactions

Message filtering

Send only the updates to the concerned users

- Avoid overloading the network

- Reduce the processing time of the messages

Reduce the nb of shared objects between users

Filter according the area of interest of users

- Objects close to a user [Waters et al., 1997]

- Objects in the field of view of a user [Funkhouser, 1995]

Technical aspects: server and multicast

Compression & Aggregation

Compression

Not relevant for position/orientation [Joslin et al., 2004]

But when data start to be complicated

 Joints of a virtual avatar, physical simulation data

Migration

Load new virtual objects (geometry, level of details)

Aggregation

Send all the object updates in one message

Can introduce delay in message transmission

Migration

[Duval et Zammar, 2006]

Referent/proxy paradigm

Move the referent to a node to another

Goals:

Balance the processing load

Move the referent on the node of the user who interacts

Technical aspects:

Upload object data on the new referent node

Delete object data on the old referent node

Dead-Reckoning

[SIMNET 93][NPSNET 94]

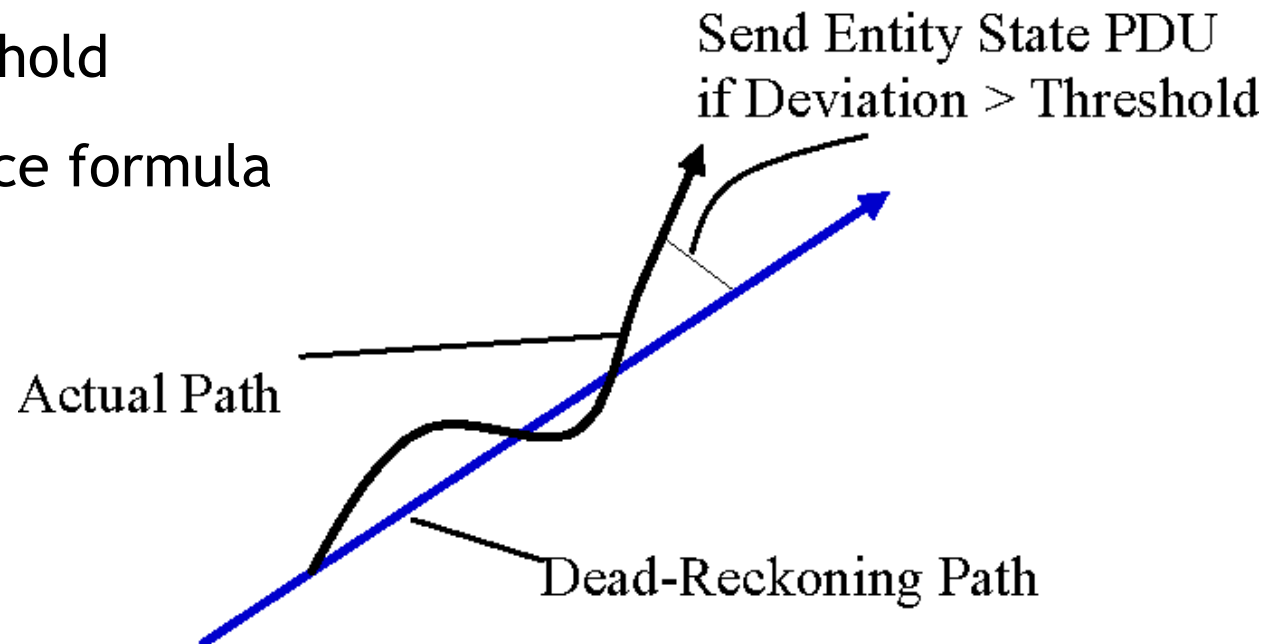
Predict object trajectories

Based on a prediction method

Prediction formula

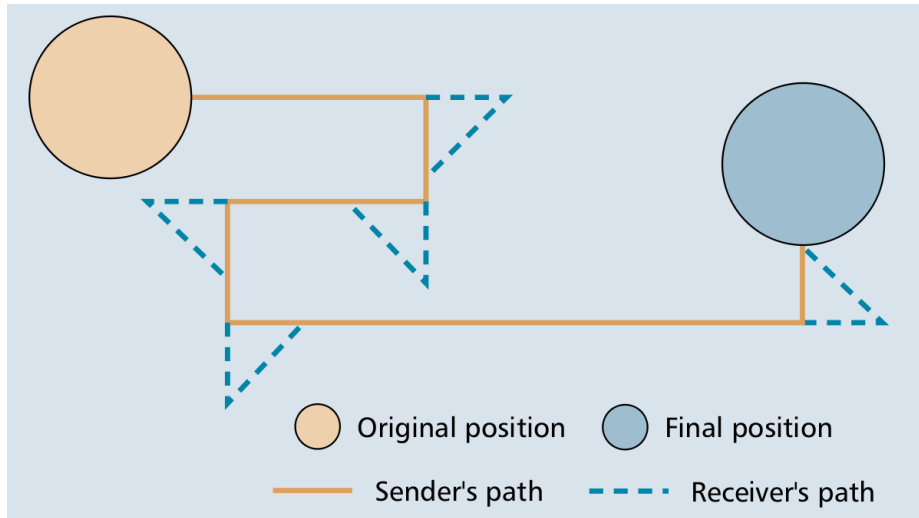
Error threshold

Convergence formula



Dead-Reckoning

[SIMNET 93][NPSNET 94]



The node in charge of the object compute

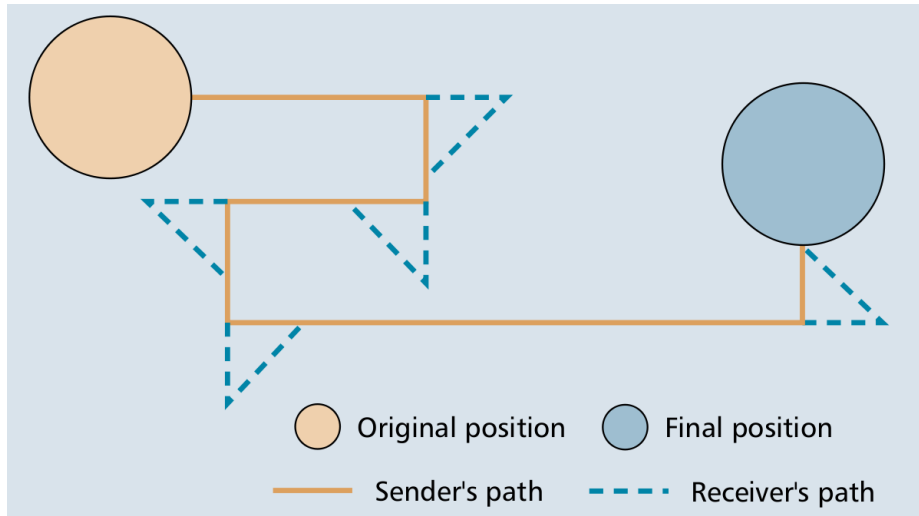
The object behavior

The prediction formula

This node does not send any update message

Dead-Reckoning

[SIMNET 93][NPSNET 94]



When the error threshold is reached

Bad prediction

Action of the user

The node send an update message

The correct state of the object is recovered using the convergence formula

Outline

Network Architecture

Data Distribution

Communication Protocols

Communication Reduction Mechanisms

Consistency Management Mechanisms

Software architecture

Consistency Management Mechanisms

Inconsistencies due to

Network delay

Concurrent modifications

2 kind of techniques

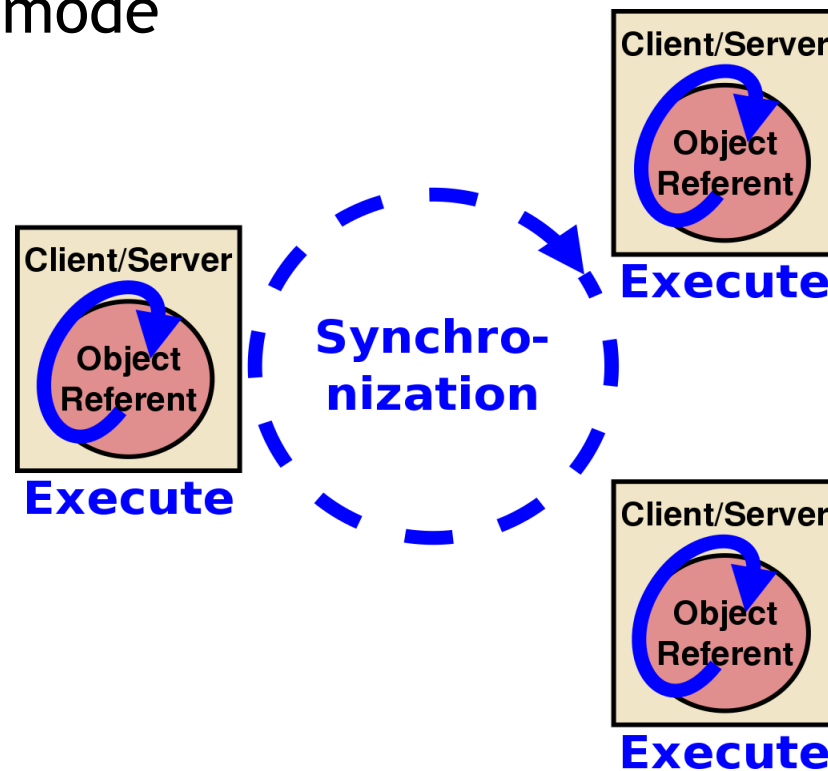
Synchronization

Concurrency control

Synchronization

Ensure that each user have the same state of the virtual environment at the same

Especially in replicated mode



Synchronization



Time is a fundamental element of CVE

Absolute time: synchronized clock (UTC)

Logical or virtual time: logical clock

Ordered sequence of events

Use timestamp

Synchronization

Lockstep synchronization [Ring 95, OpenMASK 02]

Waits all nodes before computes the next simulation step

Each node send acknowledgements to the system

Then, the system allows nodes to process the next step

Advantages

Perfect synchronization

Events are processed in the correct order

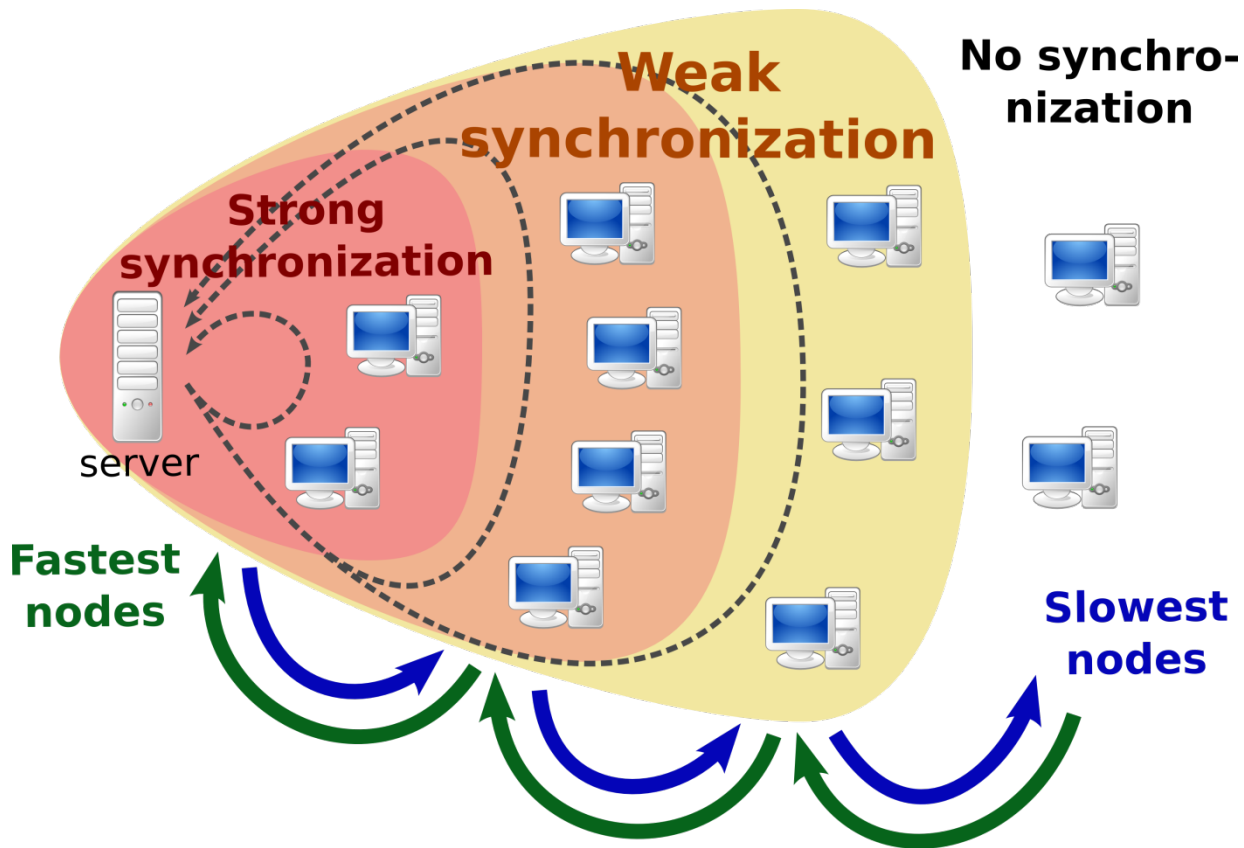
Drawbacks

Real-time is not guaranteed

One node can slow down all the others

Synchronization

Lockstep synchronization for several groups [Fleury et al 13]



Synchronization

Imposed global consistency [[Delaney et al 06](#)]

Delays the processing of local and remote events

Use a pre-defined value (max. of the network latency)

Use an absolute clock

Advantage

Strong synchronization

Drawback

Introduce latency during interactions

Synchronization

Delayed global consistency [Delaney et al 06]

Mark events with a timestamp using a logical clock

Execute events following the correct timestamp order

Advantage

Causality is ensured

Drawback

No time synchronization

Synchronization

Time warp synchronization [Jefferson 85]

Events are marked with a timestamp

Events are processed as soon as they arrive

“Rollbacks” are used to solve causality errors

Incoming event older than the event already processed

Advantage

No Latency during interactions

Drawback

“Rollbacks” are very annoying for the users (feedbacks)

“Rollback propagation”

Synchronization

Predictive Time Management [PARADE 97]

Events are predicted before they occur and send on the network

Events are sent just in time to avoid bad prediction by estimating the latency (RTT)

Advantage

Good synchronization

Drawback

Only for predictable objects (object behaviors, collision detection, etc.)

Synchronization

Server synchronization [ShareX3D 08]

Server manages a “state number” for each object

 Increments the “state number” for each modification

Server sends the last received update to nodes if they are not up-to-date

Advantages

 Ensures that nodes are up-to-date

 Reduce the number of sent messages

Drawback

 No causality and no time synchronization

Concurrency control

Centralized mode or hybrid mode (1 referent)

Server/referent can handle concurrent modification requests

Replicated mode (several referents)

Virtual objects can be modified locally on several node at the same time

Concurrency control is required

Concurrency Control

3 main modes of concurrency control

Pessimistic mode [[BrickNet 98](#)]

Only one user can modify an object at the same time

Optimistic mode [[Delaney et al 06](#)]

No concurrency control during interactions

A correction is necessary when conflicts occur

Prediction based mode [[PARADE 97](#), [ATLAS 07](#)]

Predict which users will probably modify an object

Give priority to the users according to the prediction

Users' Access Rights

Give different access rights to users

Protect virtual objects (confidential data, no modifiable objects, etc.)

Assign some role to users

3 criteria

Right to see an object

Right to modify its parameters

Right to create/delete objects

Use a scale of access level from 0 to N
(0 is the most restrictive)

Outline

Network Architecture

Data Distribution

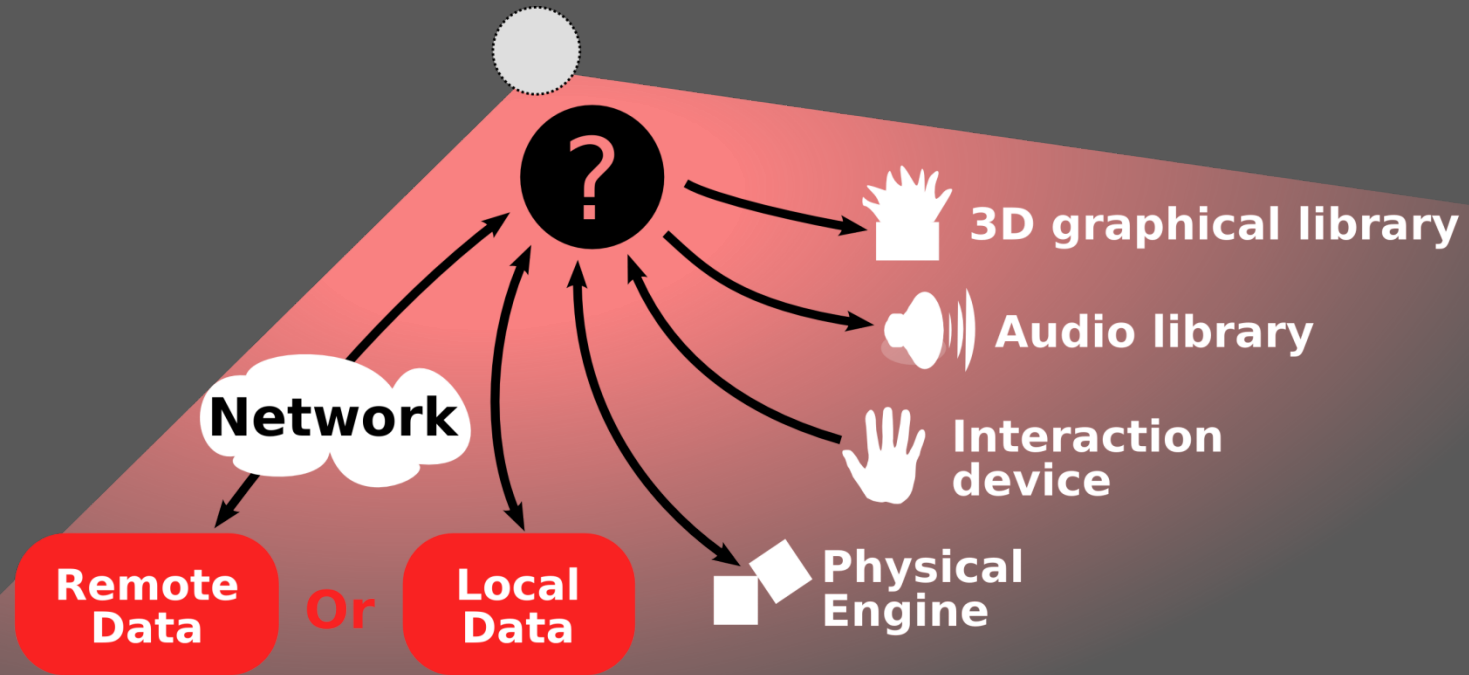
Communication Protocols

Communication Reduction Mechanisms

Consistency Management Mechanisms

Software architecture

Software Architecture



⇒ How to design virtual objects in order to insure a good separation between data distribution and multiple representations?

Models for Interactive System

Application can be decomposed in 3 parts

Core component

- Store data

- Execute behavior

- Process users' modification requests

Interface component

- Make the link with the users

 - Display the object

 - Register the action of the users

A link between the Two components

Models for Interactive System

Existing models

Functional decomposition

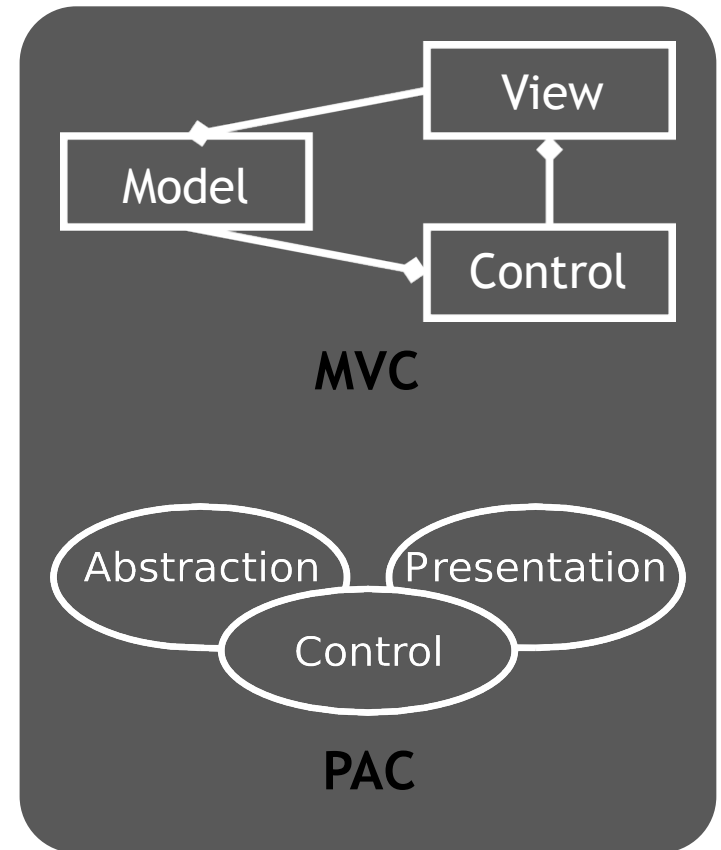
Arch [UIMS 92]

Multi-agents

MVC [Reenshaug 79][Eckstein 07]

PAC [Coutaz 87][Duval et Tarby 06]

Hybrid [Nigay et Coutaz 91]



Models for Collaborative System

Distributed data on remote computer

Manage communications

Existing models

Abstraction layers [Dewan 99]

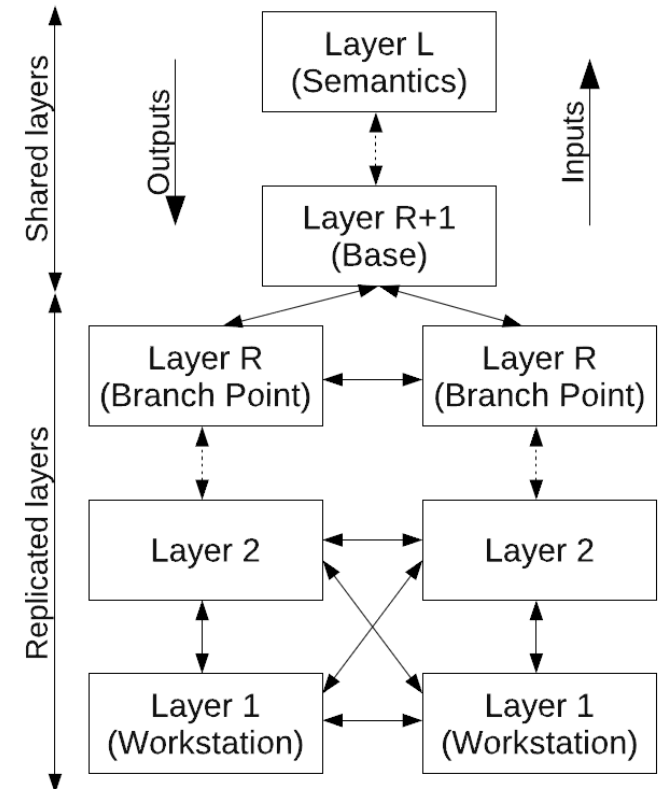
Multi-agents

ALV [Hill 92]: shared abstraction

CoPAC [Salber 95]: Additional communication component

Functional description of collaboration

[Calvary et al. 97][Laurillau et Nigay 02]



Dewan

Synthesis

Multi-agents models are well adapted for VE

A virtual object = an agent

Existing models do not deal with data distribution

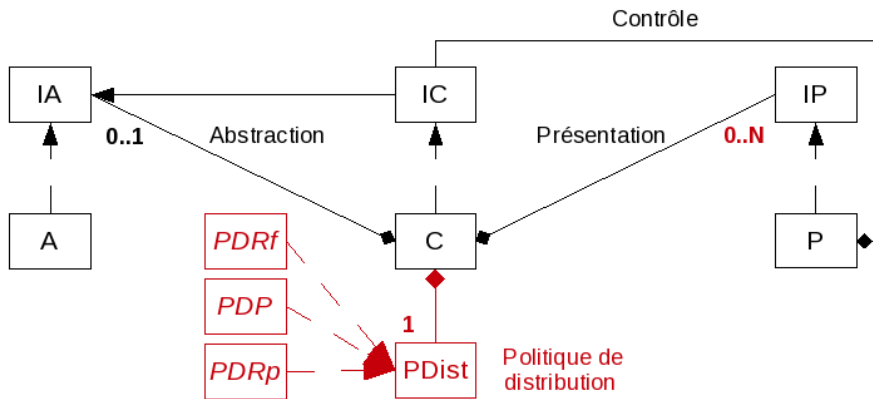
ALV proposes only a centralized data distribution

CoPAC does not specify the data distribution

⇒ We need a model which manage data distributions

PAC-C3D Model

[Duval et Fleury, 2011]



Extend the PAC model to the CVE

Each virtual object is modeled by a PAC agent on each node

The Control manages the network distribution

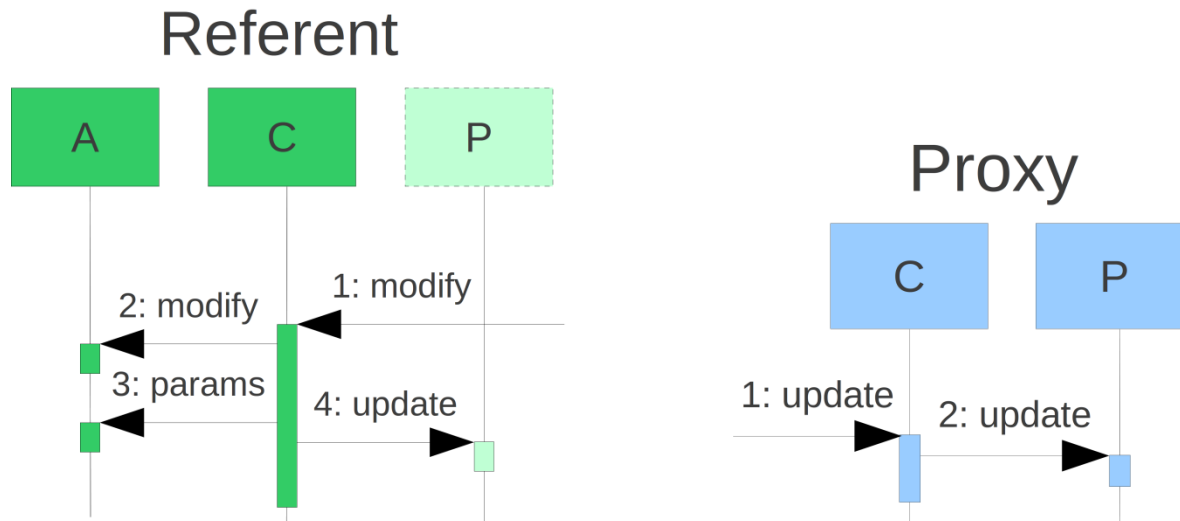
Maintains the consistency between all the nodes

Several distribution policy (one for each data distribution mode)

Provides generalized interface to access to the object

Multiple Presentations of a same virtual object

Data Distribution



Easy implementation of referent/proxy paradigm

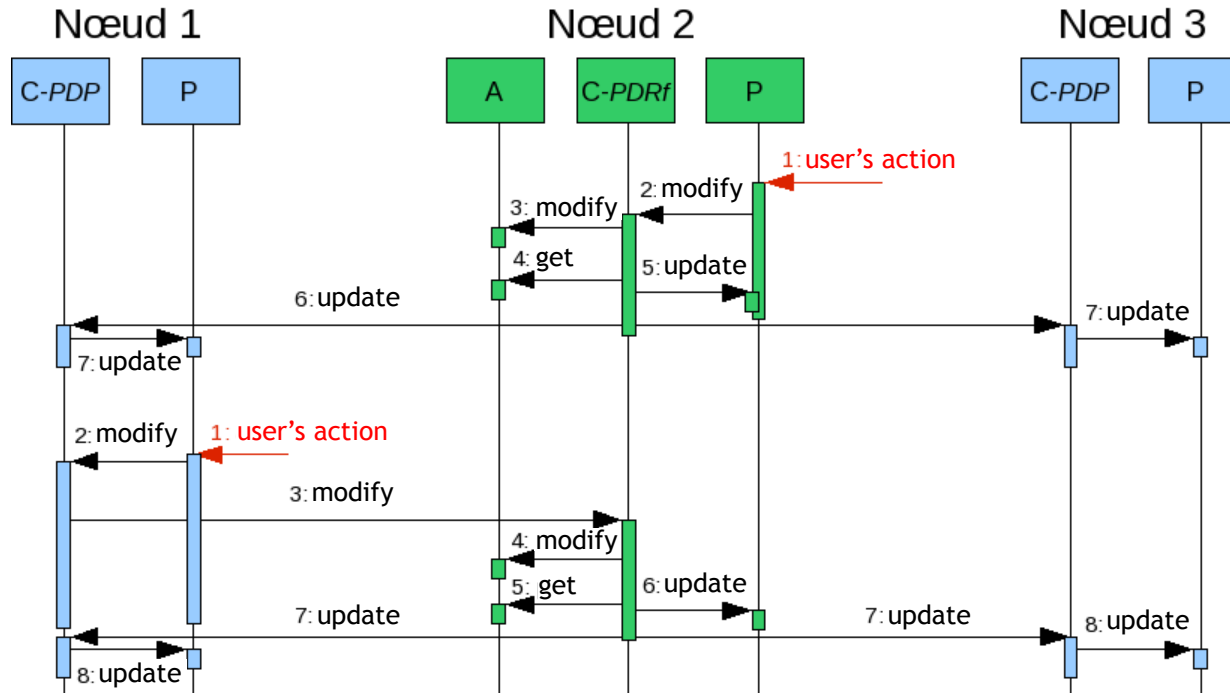
Interoperability between virtual objects

(even if they don't use the same data distribution mode)

All accesses to objects are managed by the Control

Dynamic migration of the referent

Example for the hybrid mode



All modification requests are sent to the Control

The Control:

- Chooses where the requests should be processed

- Manages updates of the remote versions of the objects

Advantages for data distribution

« Interoperability » between objects using different data distribution modes on the network

All the accesses go through the Control

Easy migration of the referent

Change the distribution policy of the Controls

Create an updated Abstraction for the new referent

Delete the Abstraction of the old referent

Developer do not have to deal with data distribution

They just have to heritage from basically components

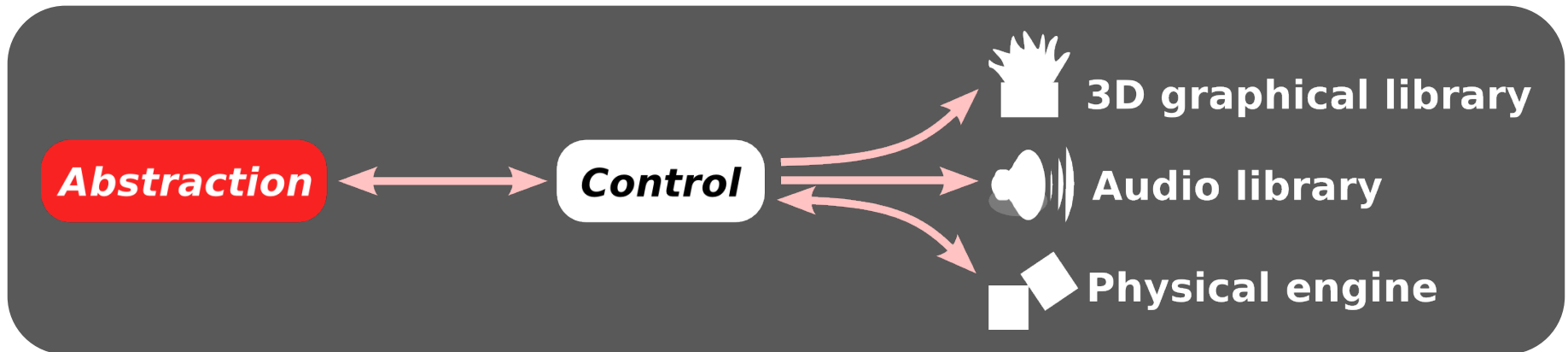
Multiple representations

Several Presentations of an object on the same node

Multi-sensorial representation of the object

Add of some “active” Presentations

Ex: physical instance of objects in a physical engine

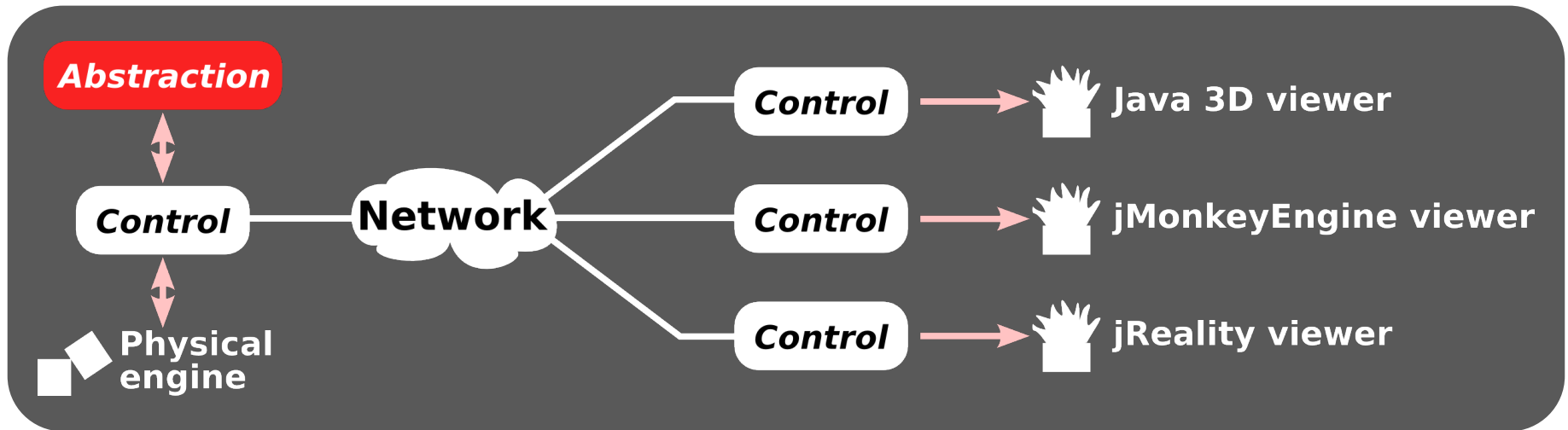


Multiple representations

Several Presentations of an object on multiple nodes

No duplication of data and behavior processing in each software libraries

Interoperability between several software libraries



Conclusion

Common issues of CSCW applications

Trade-off between consistency and responsiveness

Network architecture and data distribution

Consistency management mechanisms

⇒ No solution which fits all application requirements, so an adaptive solution might be a good solution

Software architecture has to deal with

Data distribution over the network

Various software libraries and materiel devices

⇒ Make a clear separation between core application part, data distribution part, interface with the users