Application de l'Approche Réactive
Simulations de mondes virtuels

Frédéric Boussinot, Jean-Ferdy Susini

Projet Mimosa commun avec
l'École des Mines de Paris - CMA

Travaux soutenus par France Télécom-R&D

Plan

- Comportements et simulations
- Systèmes parallèles distribués
- Jeux on-line massivement multi-joueur

Simulations virtuelles

Simulations :

collection d'entités autonomes exécutées en parallèle interagissant entre elles.

=> programmation parallèle ?

Réponse classique : non, car trop inefficace.

Réponse possible :

Oui ! programmation réactive !

Méthode classiquement mise en œuvre

Programmation objet pour représenter les différentes entités du monde simulé.

Structuration en arbre

Une notion d'instant décomposée en 3 phases :

- Gestion des interactions extérieures : Game Logic (réseau, interactions utilisateur,...)
- Calculs par nécessité des évolutions du système par remonté le long d'un arbre d'objets.
- Affichage et son : rendering

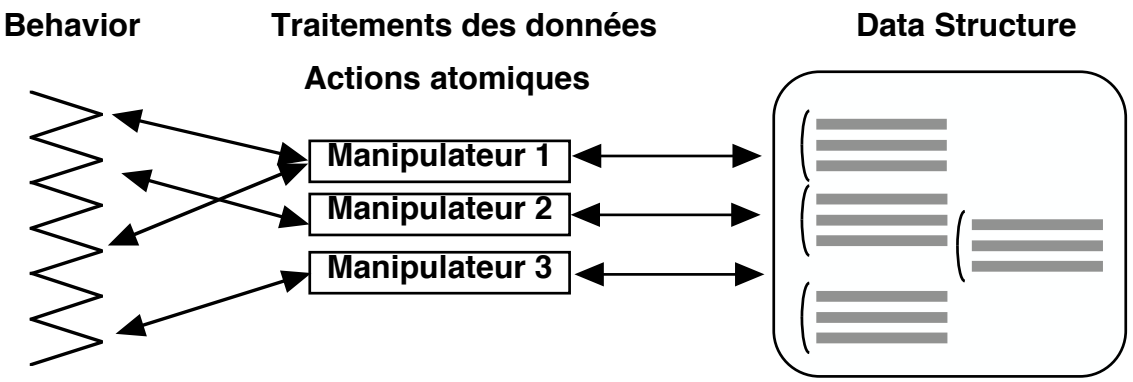
Exemple : Quake

Main Loop

```
    while(1){
// find time spent rendering last frame
        newtime = Sys_FloatTime();
        time = newtime - oldtime;
...
// decide the simulation time
    if (!Host_FilterTime (time))
        return;
       Sys_SendKeyEvents();IN_Commands();
...
       Host_ServerFrame();// SV_Physics() et SV_RunThink()
// update video
       SCR_UpdateScreen();
...
// update audio
       CDAudio_Update();
    }

-----------------------------------------------------------------
//
// treat each object in turn
//
    ent = sv.edicts;
    for (i=0 ; i<sv.num_edicts ; i++, ent = NEXT_EDICT(ent)){
...
     }
```

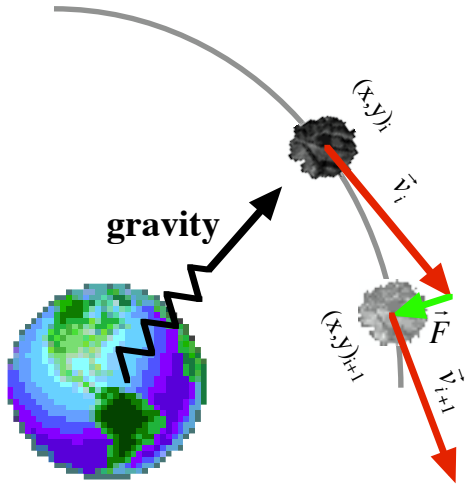
Entitée réactive



Un objet réactif dans une simulation est décrit par:

- **une structure de données (état);**
- **des opérations de traitements (manipulateurs);**
- **un comportement (ordonnancement et contrôle de tâches).**

Simulation des interactions physiques



Conserver un rendu réaliste des interactions physiques tout en conservant une programmation modulaire.

Simulation Physique

Le pendule simple :

$$\ddot{\theta} + k \cdot \sin(\theta) = 0$$

Résolution numérique : étapes successives de calculs (Runge Kutta)

```
Jr.Loop(
   Jr.Seq(
      Jr.Atom(new RKStep())
      ,Jr.Seq(
         Jr.Generate("angle", new
Position())
         ,Jr.Stop())));
```

Composition parallèle des interactions

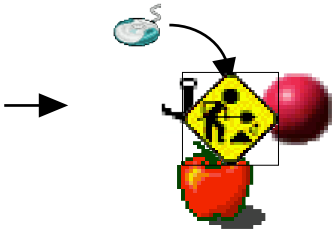
Dynamicité et modularité des systèmes simulés

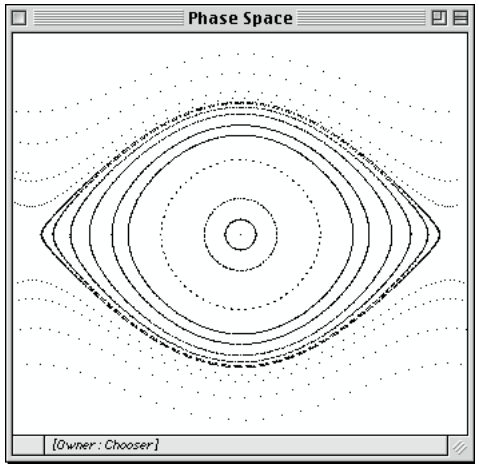
Constructeur

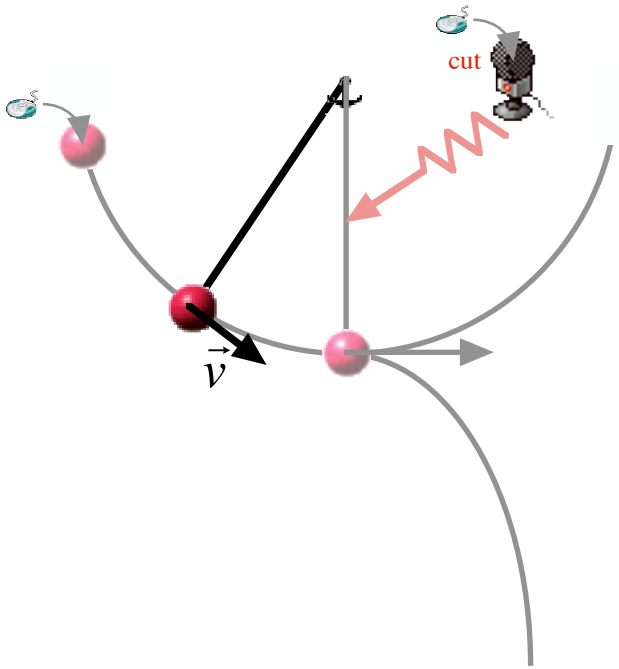
Tige rigide

Inertie

Gravité



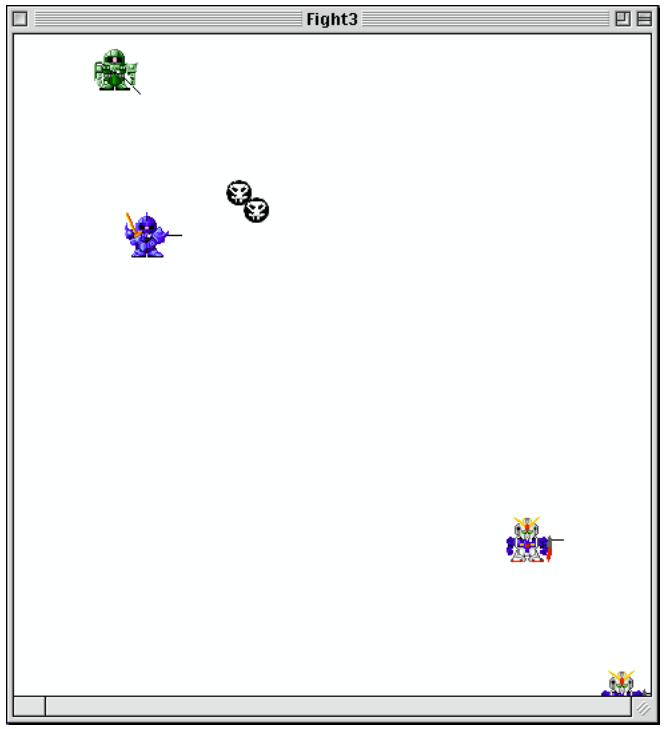


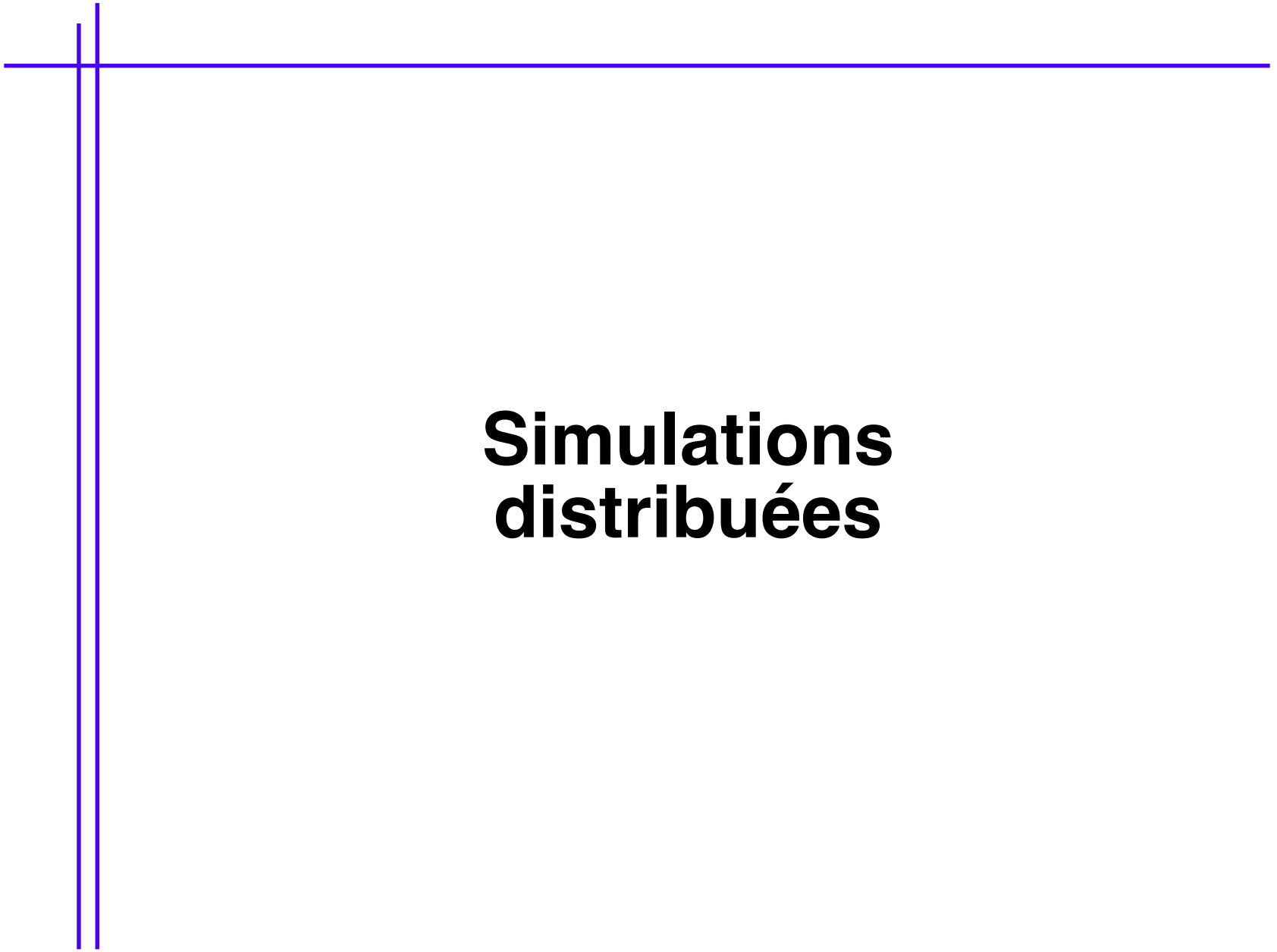


Démos

Programmation de robots évoluant dans des arènes : Fight Club.

Mélange de comportements physiques avec des comportements à événements discrets (interactions avec l'utilisateur).





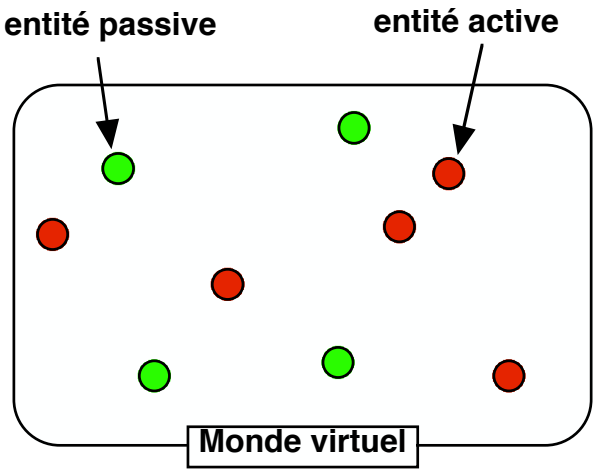
Simulations distribuées

Simulations distribuées

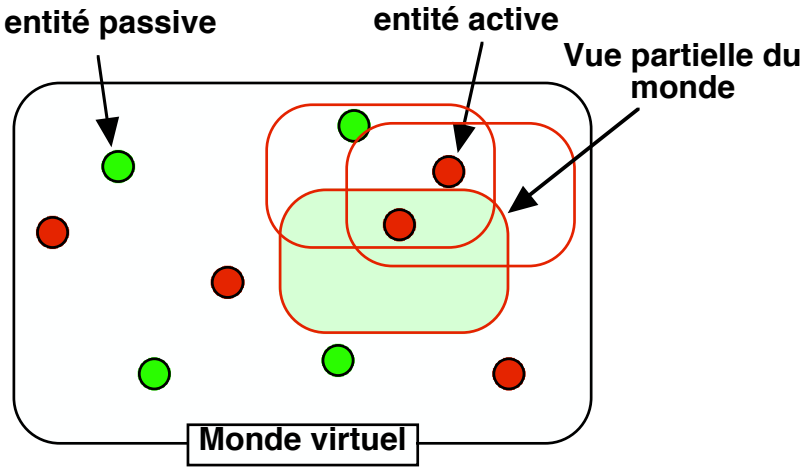
Mondes virtuels distribués :

- **être multi-utilisateurs**
- **augmenter le nombre de ressources de calculs**
- **persistance**

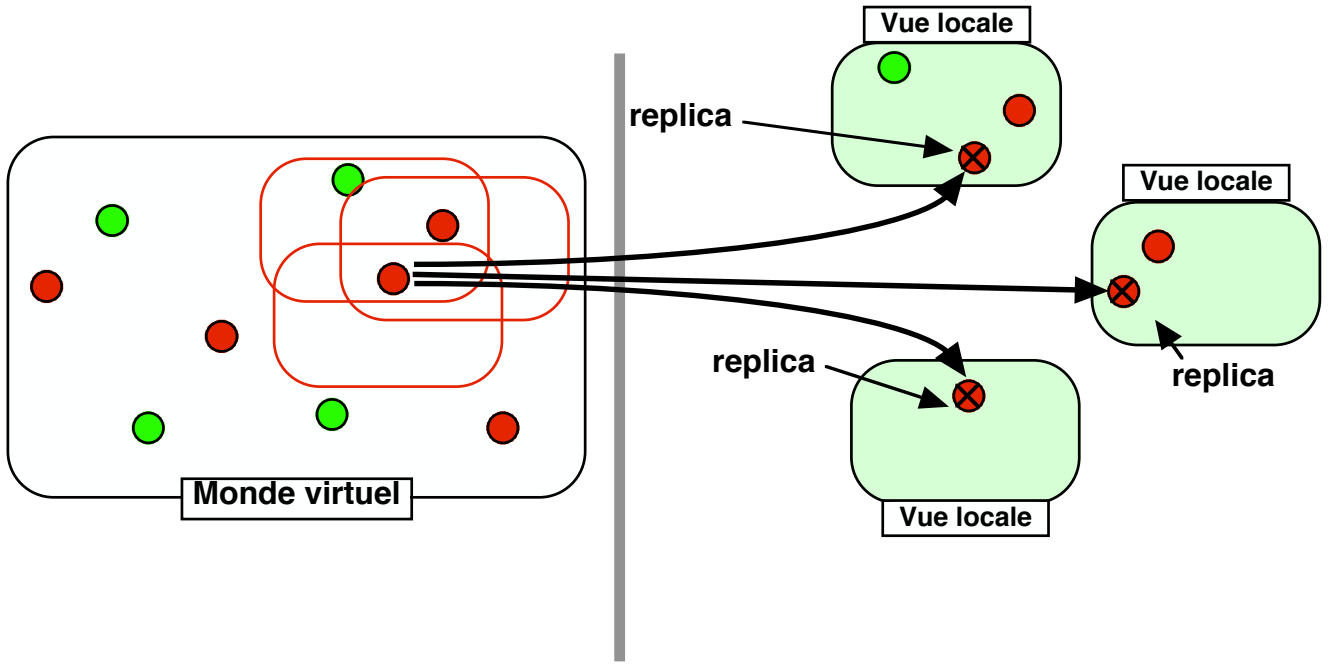
Monde virtuel



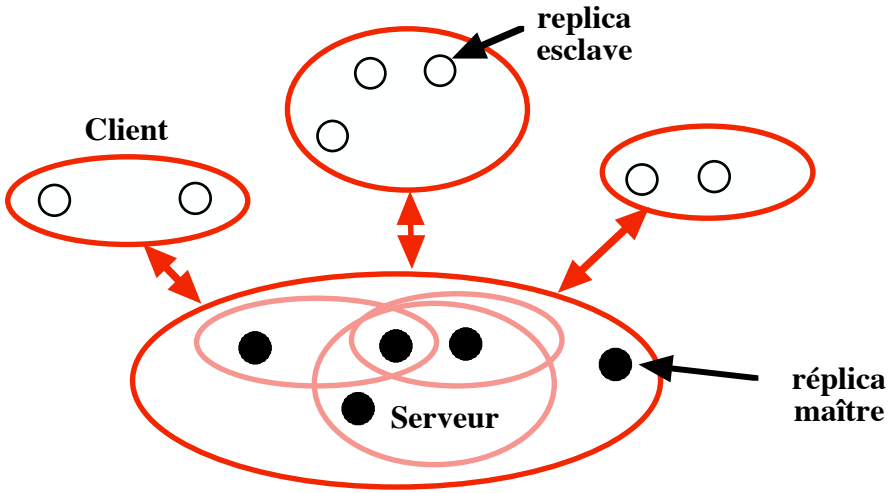
Vues partielles



Vues distribuées



Approches centralisées



Un serveur centralisé réalise l'ensemble des calculs de la simulation.

Les clients envoient au serveur les événements produits par les actions du joueur.

Les clients reçoivent régulièrement des informations du serveur et font peu d'extrapolations.

Solution centralisée

Avantages :

- **assure la cohérence de la simulation;**
- **simple à mettre en œuvre;**
- **sécurité.**

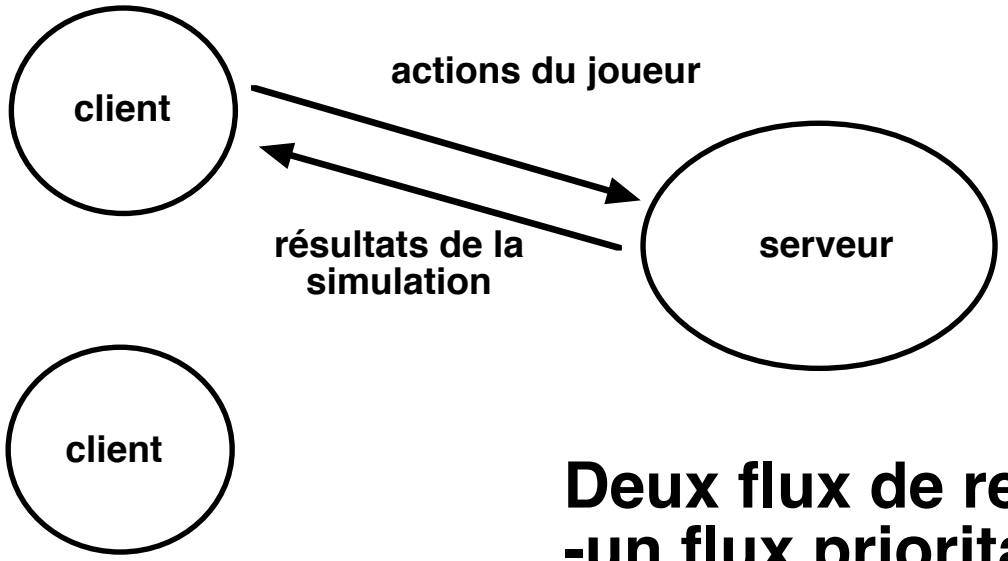
Inconvénients :

- **passage à l'échelle d'un grand nb de joueurs difficile (Charge serveur et réseau);**
- **pannes;**
- **réactivité en cas de délais réseau.**

Quake

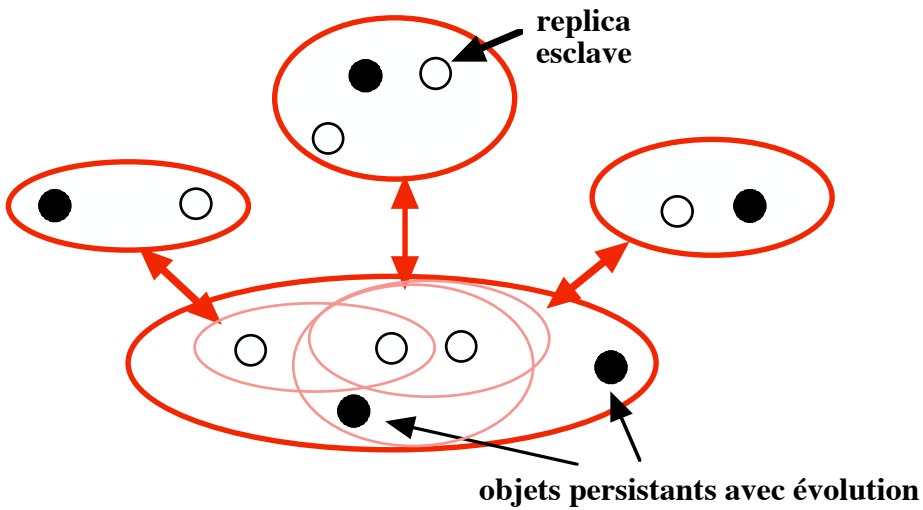
Nouvelle Frame = étape de simulation

Arborescence d'objets activés par nécessité à chaque étape (méthode `think`).



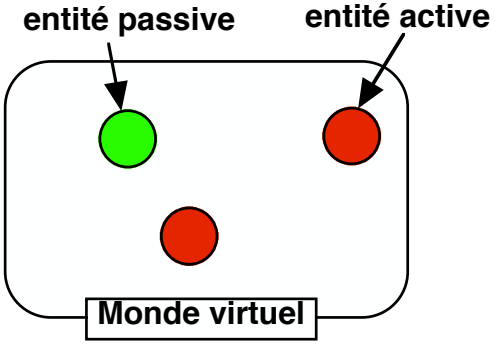
Deux flux de retour :
-un flux prioritaire
-un flux optionnel

Approche décentralisée

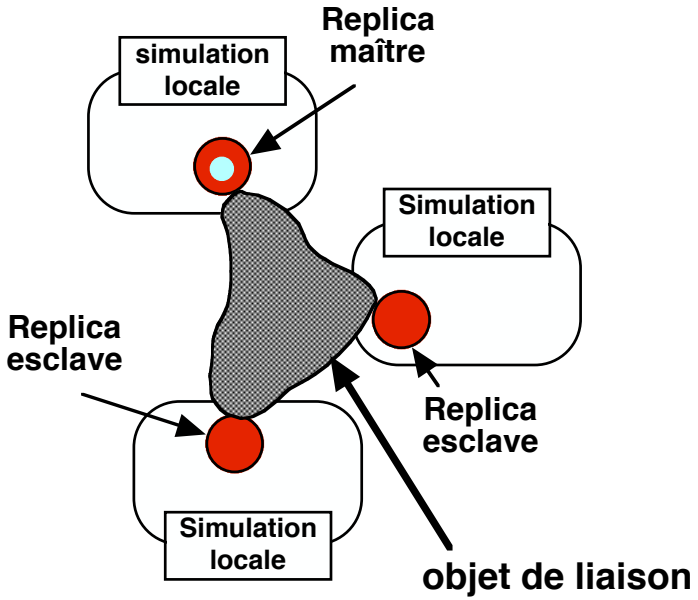


Mise à profit du max de ressources de calcul : chaque client participe aux calculs de la simulation.
Chaque simulation locale calcule l'évolution des réplicas maîtres dont elle a la charge.

Objet de liaison (binding object)



Niveau applicatif



Niveau plate-forme

Solution décentralisée

Avantages :

- **chaque objet de liaison peut optimiser les routes sur le réseau;**
- **charge de calcul répartie en fonction du nombre de participants;**
- **meilleure réactivité du système;**
- **tolérance aux pannes.**

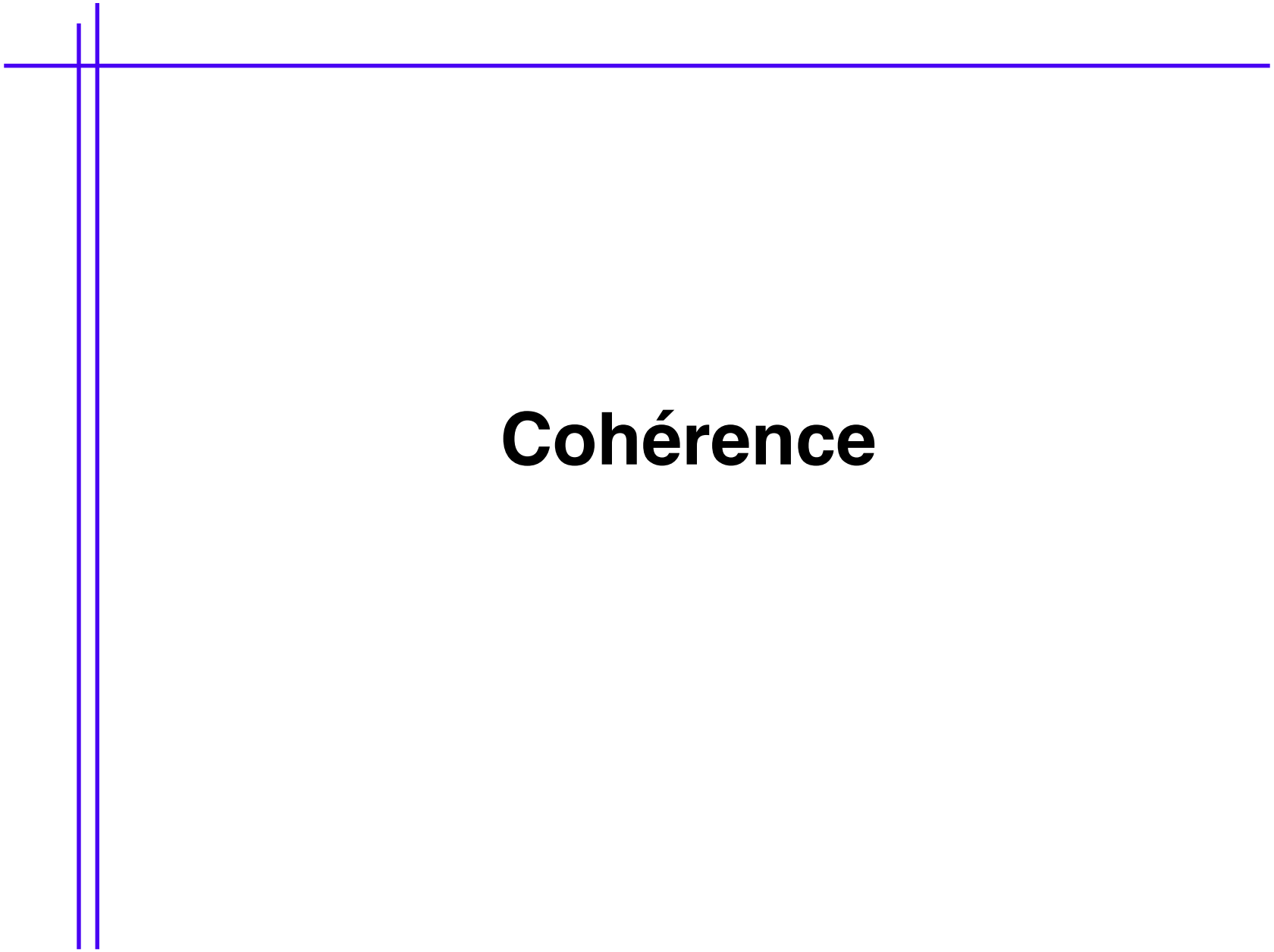
Inconvénients :

- **problème de cohérence de la simulation globale;**
- **sécurité.**

Diablo

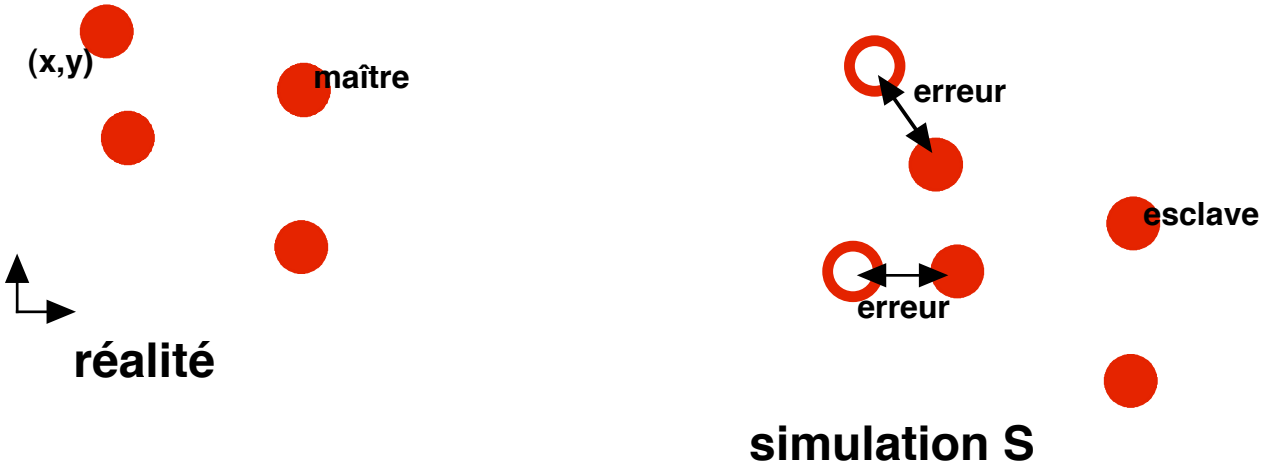
Chaque joueur simule une partie du monde.

- **Synchronisation par nécessité (manipulation d'objets,...)**
- **Le donjon existe tant qu'il reste un joueur.**
- **Pas d'évolution en dehors de la zone d'intérêt du joueur.**
- **Très nombreuses incohérences entre simulations.**



Cohérence

Notion de cohérence



distance d

incoherence(S) = $\sum$ d(o,S(o))

incoherence(S)$_t$ = $\sum$ d(o$_t$,S(o$_t$))

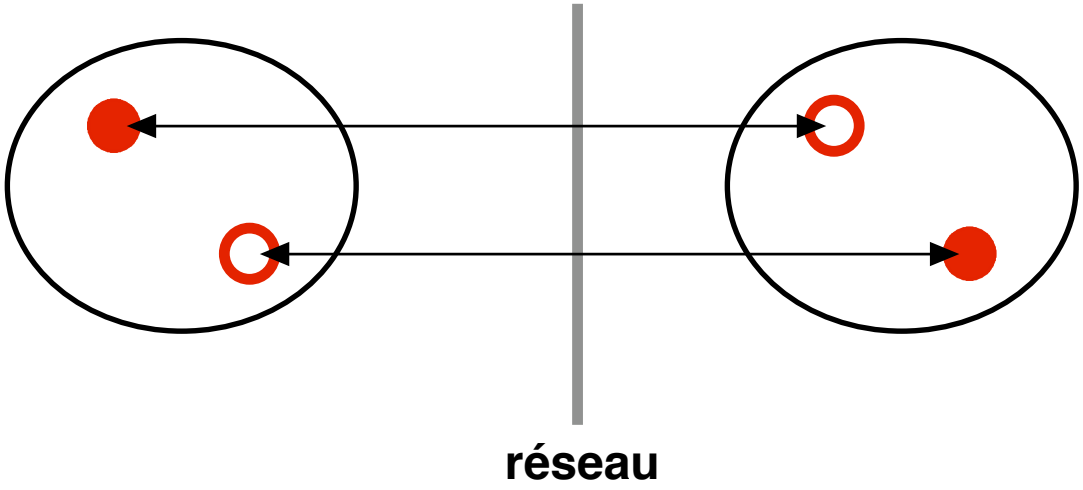
Simulation partielle

Image partielle de la réalité
Exemple : vision centrée sur un objet

Nécessité d'un mécanisme de peuplement :
création et destruction dynamique de réplicas

Maîtres distribués

- Simulations distribuées à travers le réseau
- Pas de notion de temps commun
- Maîtres gérés par les simulations réparties



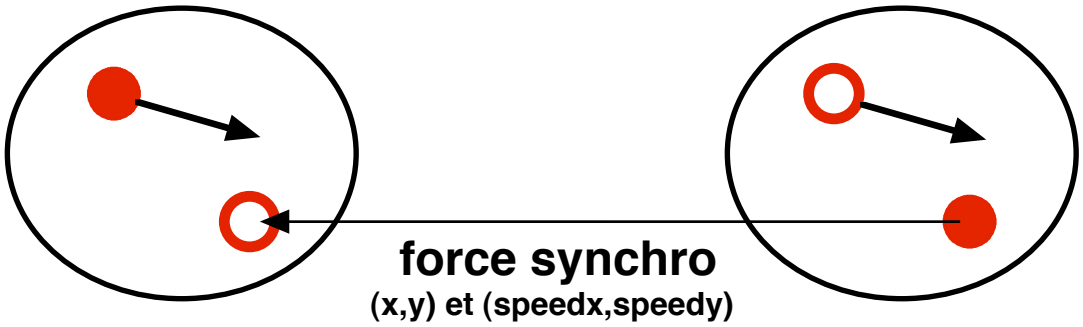
Problème central : comment maximiser la cohérence des simulations entre elles, tout en minimisant le trafic réseau

Anticipation

3 sources de non-déterminisme :

- **actions externes sur les maîtres**
- **délais de transmission**
- **charge des simulations**

Mécanismes d'anticipation (dead-reckoning) : définissent le comportement des réplicas esclaves entre deux synchronisations



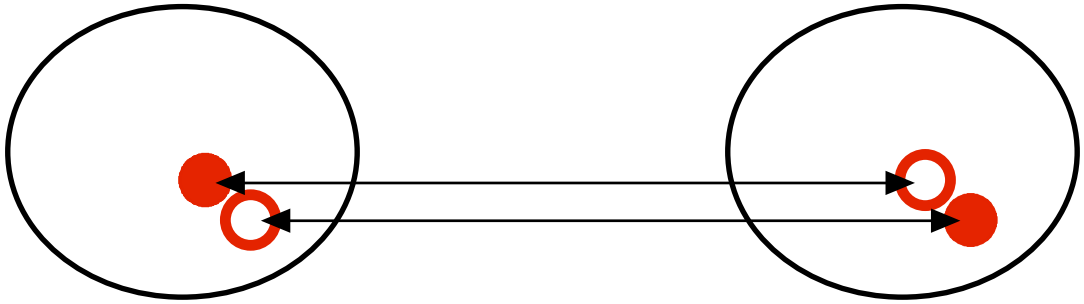
Collision

Pour minimiser la charge des simulations :

- **comportements dégradés des réplicas**
- **collisions traitées uniquement par les maîtres**

Complexité générale des collisions : N*(N-1)/2 : ~ $O(N^2)$

Chaque simulation : N-1 : ~ O(N). Complexité totale plus grande, mais parallélisme



Il y a des possibilités d'incohérences…

Architecture

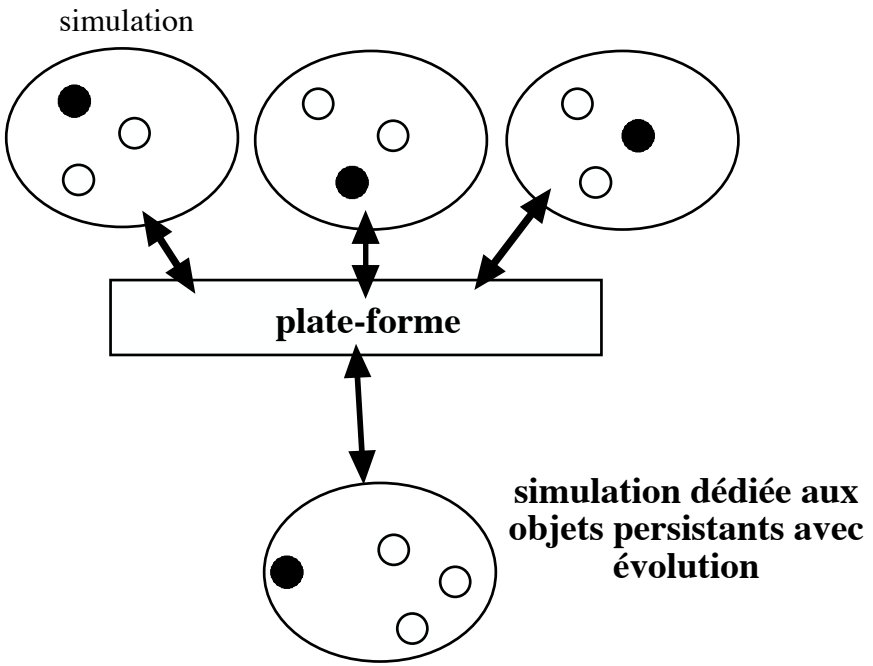
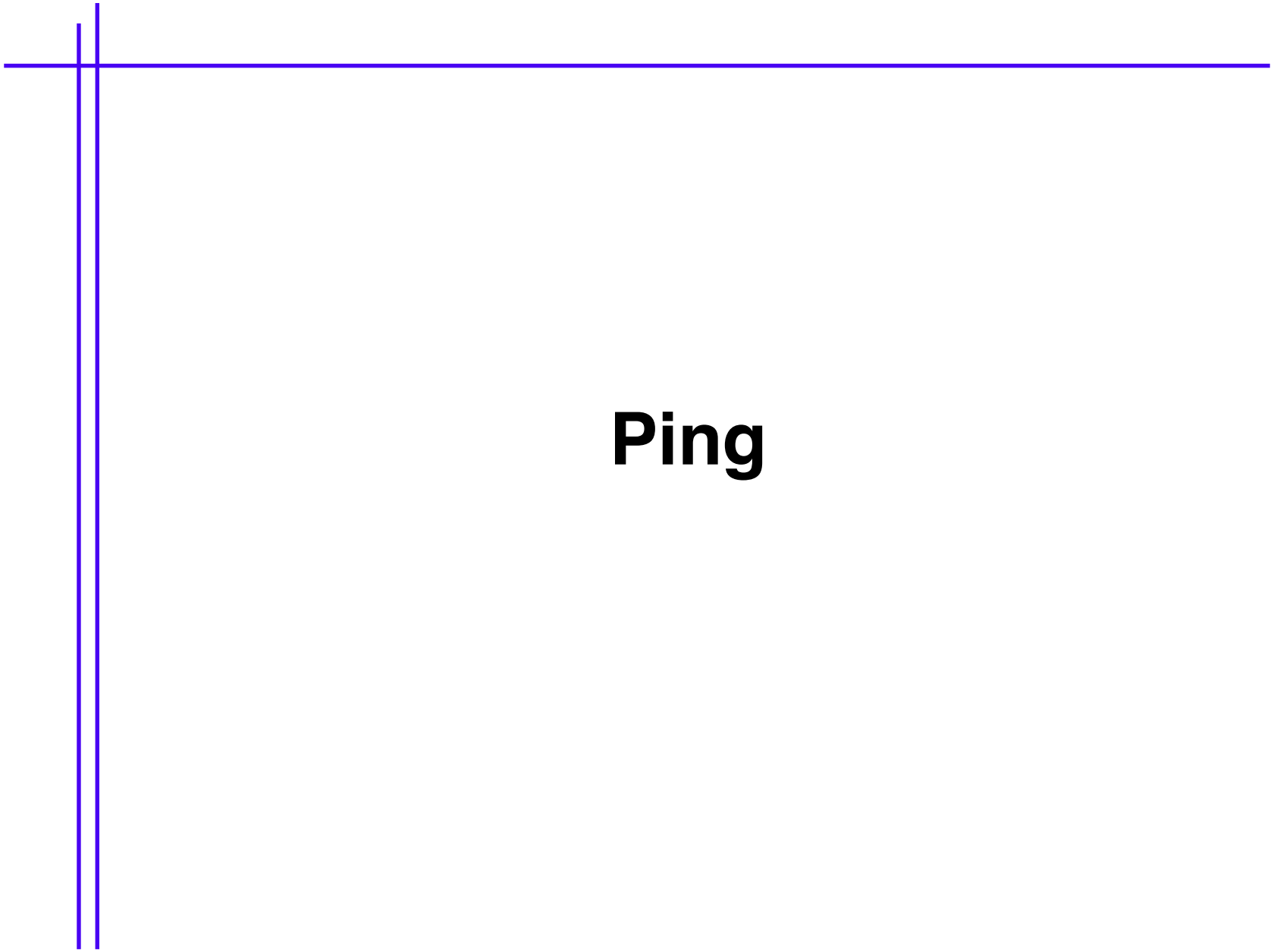


Plate-forme fournit différents services :

- **population des simulations;**
- **diffusion d'informations;**
- **persitance des données;**
- **transfert de mastership.**
- **évolution**



Ping

Ping

PING:

- Platform for Interactive Networked Games -

Objectifs : fournir une architecture software permettant le déploiement de Mondes Virtuels Massivement Multi-joueurs.

+ Midleware orienté objets (ORB).

+ Protocoles de communication efficaces et adaptables

+ Services de gestion d'objets de haut niveau (persistance, réplication, cohérence)

+ Système de partitionnement efficace compatible avec la notion de monde virtuel continu (Aura).

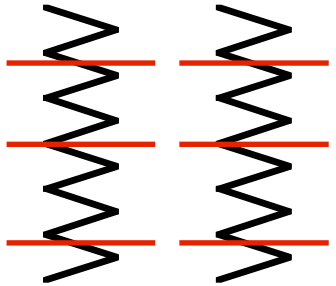
+ Utilisation de l'Approche Réactive dans la description des comportements des objets de simulation.

+ Introduction de notions de programmation au niveau du gameplay (Icobjs).

Comportements réactifs

Comportement des objets définis dans l'Approche Réactive :

- **ordonnancement des traitements de données;**
- **contrôle d'exécution;**
- **synchronisation naturelle avec les comportements des autres entités;**
- **communication entre entité par événements diffusés instantanément.**



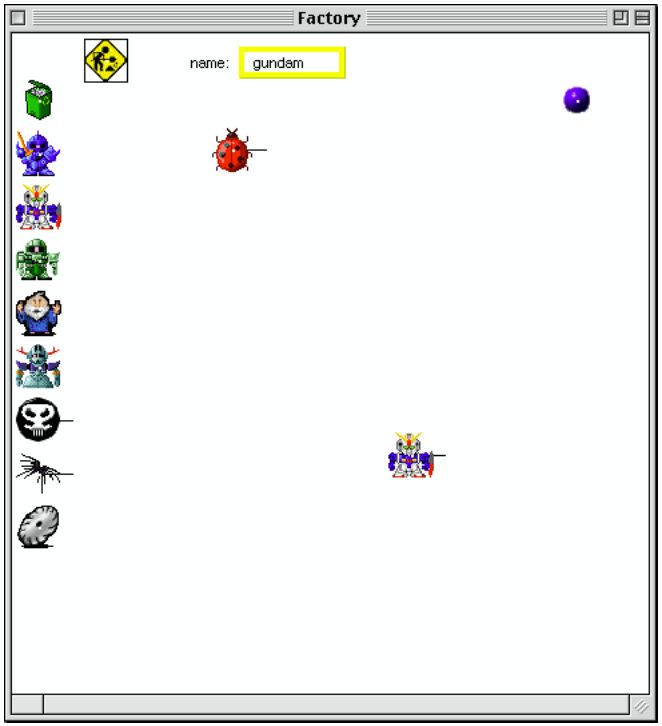
Comportements des réplicas :

- **le comportement de l'entité autonome devient le comportement du réplica maître.**

- **Les comportements des réplicas exclaves sont des versions dégénérées du comportement du maître. Ces comportements permettent d'implémenter des stratégies d'anticipation : dead-reckoning.**

Extension du gameplay

Introduction de la programmation au niveau applicatif : programmer son avatar.



Problèmes :

- **sécurité**
- **comportement des réplicas**

Icobjs Arena

- Un avatar par simulation (réplica maître)
- Simulation partielle centrée autour de l'avatar
- Les réplicas maîtres des avatars ont un comportement inertiel et traitent les collisions
- Les réplicas esclaves n'ont que le comportement inertiel
- Les avatars peuvent lancer des bombes qui explosent au bout d'un certain temps.
- Les bombes n'explosent pas en cas de collision.
- Les réplicas maîtres des bombes traitent les collisions comme n'importe quel avatar.
- Les bombes sont des objets persistants qui subsistent après déconnexion du lanceur.
- Les joueurs contrôlent la direction du canon de l'avatar avec les touches du clavier.
- Le lancement d'une bombe produit un effet de recul permettant au joueur de se déplacer.

Icobjs-Arena

IcobjsArena

Conclusion

Simulations distribuées de mondes virtuels :

- **pas de simulation centrale.**
- **charge de calcul répartie : distribution des maîtres.**
- **comportements dégradés des réplicas.**
- **méthodes d'extrapolations : dead-reckoning (minimise la charge réseau)**
- **peristance avec évolution (simulation dédiée)**

Papier : *A Reactive Behavior Framework for Dynamic Virtual Worlds*, Conf. Web3D 2001.