Cohérence de données pour la gestion de déconnexion

Sophie Chabridon et Lydialle Chateigner

get . . . INT

GDRI3/GTMOB, CNAM, Paris

13/01/2005

Plan de la présentation

1 Introduction

- Dans le contexte des environnements mobiles, besoin de répliquer les données
  - Pour améliorer les performances et augmenter la disponibilité des données
  - A la demande des applications (travail collaboratif)
  - Pour la continuité de service (travail en mode déconnecté)
- Solutions envisageables pour la gestion de données
  - Ecrivain unique / lecteurs multiples : réplication pessimiste avec verrouillage
  - Ecrivains multiples / lecteurs multiples
    - Possible avec un seul utilisateur : copies sur différents terminaux, travail déconnecté
    - Travail collaboratif : mise à jour locale puis propagation aux autres membres du groupe
    - Réplication optimiste (pour permettre le travail déconnecté)
    - Transfert d'état vs transfert d'opérations (simplicité vs facilité de réconcilier)

2 Bref état de l'art

- Point de départ : AS19 du CNRS-STIC "Mobilité et accès aux données" [Pucheral, P. et al., 2004]
  - Travaux du LIRMM sur la convergence des copies en environnement collaboratif distribué et temps réel
  - Travaux de l'équipe ECOO du LORIA sur le travail collaboratif multi-synchrone
- Accent sur les solutions multi-écrivains avec transfert d'opérations [Saito, Y. and Shapiro, M.,]
- Travaux importants du domaine :
  - Bayou [Terry et al., 1995]
  - IceCube [Kermarrec et al., 2001]

2.1 Bayou

- Projet de Xerox PARC (1993 - 1997)
- Applications visées : Bases de données pour applications collaboratives mobiles
- Critères syntaxiques (basés sur les estampilles) pour déterminer l'ordre d'exécution des opérations
- Pour chaque opération, définir un test de vérification des dépendances (détection de conflit) et une procédure de fusion
- Chaque écriture est estampillée (accept-stamp) et marquée à l'aide de l'id de l'écrivain
- Chaque site maintient un journal des écritures et une copie de la base de données
- Protocole Primary Commit :
  - Estampille de validation ($\inf$ tant que l'opération n'est pas validée)
  - Serveur primaire estampille les opérations (séquenceur)
- Algorithme anti-entropique épidémique
- Retour arrière possible

2.2 IceCube

- Travaux démarrés en 2000 par M. Shapiro chez Microsoft Research Laboratory (UK)
- Basé sur une exploration heuristique des ordonnancements possibles
- Pendant la phase de travail isolé, chaque copie génère un journal local
- Définition très détaillée de contraintes
  - Contraintes statiques, indépendantes de l'état de l'objet : log, object
  - Contraintes dynamiques (assertion, détection de conflit à la Bayou ...)
- Gestion de plusieurs versions de l'état répliqué des données partagées item

Réconciliation des logs de deux copies en 3 étapes :

1. Génération d'ordonnacements en combinant les opérations des 2 logs en respectant les contraintes statiques
2. Simulation : abandon d'un ordonnancement si non satisfaction des contraintes dynamiques
3. Sélection d'une solution optimale

- Proposition d'un modèle de cohérence
- Applications visées : Applications collaboratives, mobiles

2.3 Transformées opérationnelles

- Permet de transformer une opération (changement de paramètres...) afin de sérialiser des opérations concurrentes (potentiellement non-commutatives) pour assurer la convergence des copies
- Preuves théoriques associées ([Vidot, 2002])
- Plusieurs types de transformées
  - Transformée en avant : tient compte de l'effet d'une opération concurrente
  - Transformée en arrière : change l'ordre d'exécution de deux opérations
  - Transformée inverse
- Contribution de l'INT : Adaptation des solutions pour le travail collaboratif multi-synchrone aux environnements mobiles

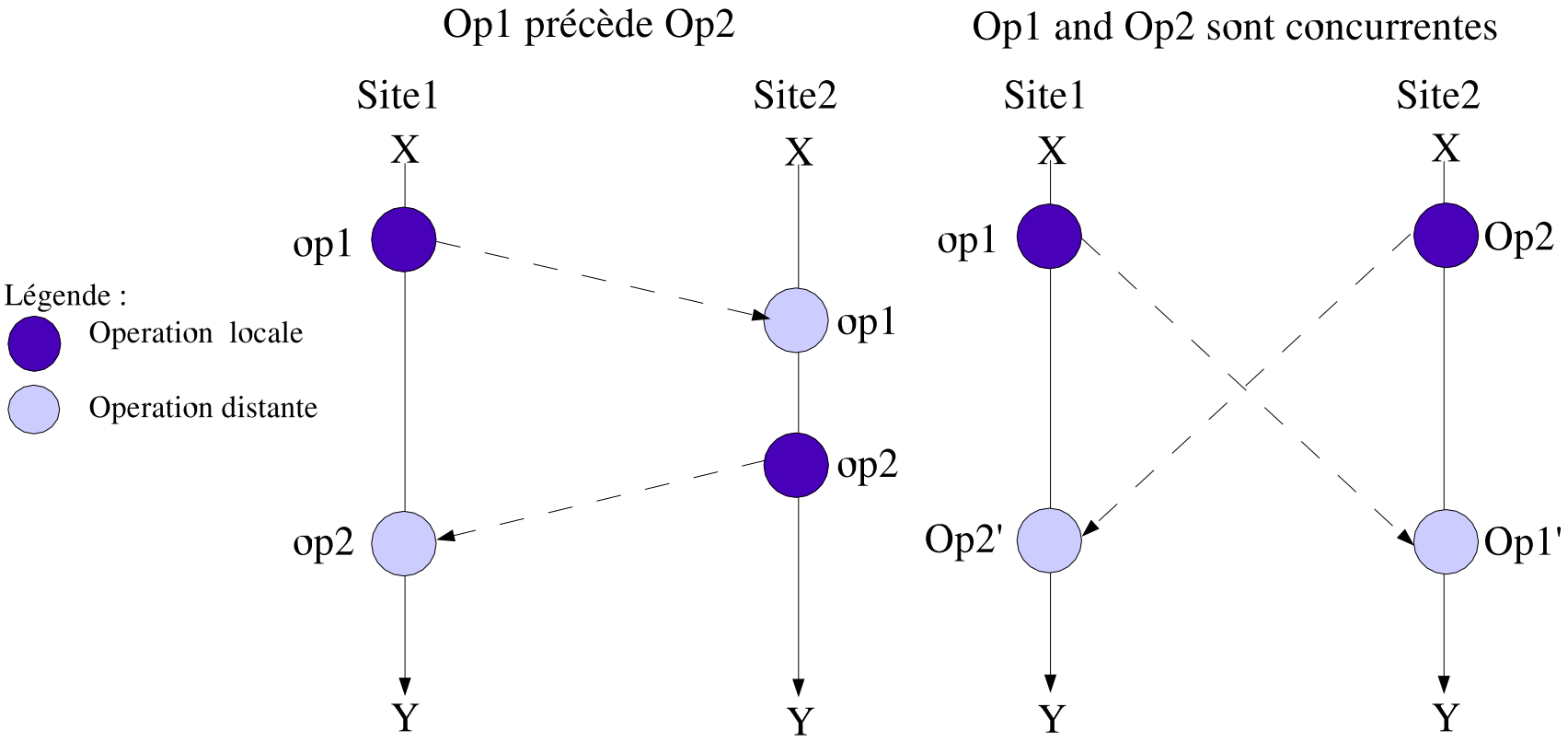
3 Base des travaux : Algorithme de réconciliation SOCT4

SOCT4 : Sérialisation des opérations concurrentes par transposition [Vidot, 2002]

- Conçu pour le travail collaboratif à contraintes de temps
- Exploitation des propriétés sémantiques des opérations

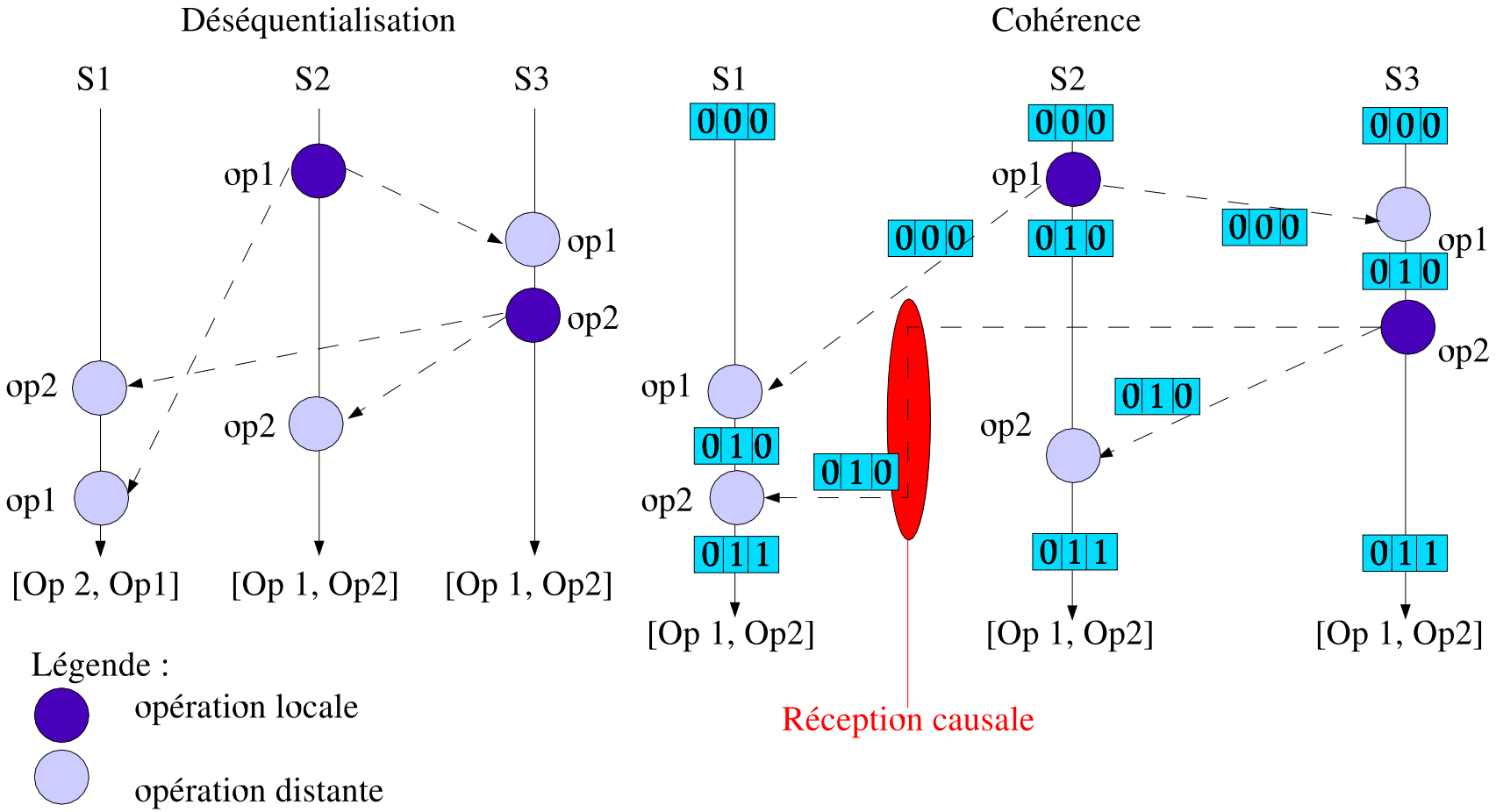
- Utilise la transformée en avant uniquement
- Ordre global continu (besoin d'un séquenceur)
- Une seule condition (C1) à vérifier pour converger

3.1 Précédence causale vs concurrence



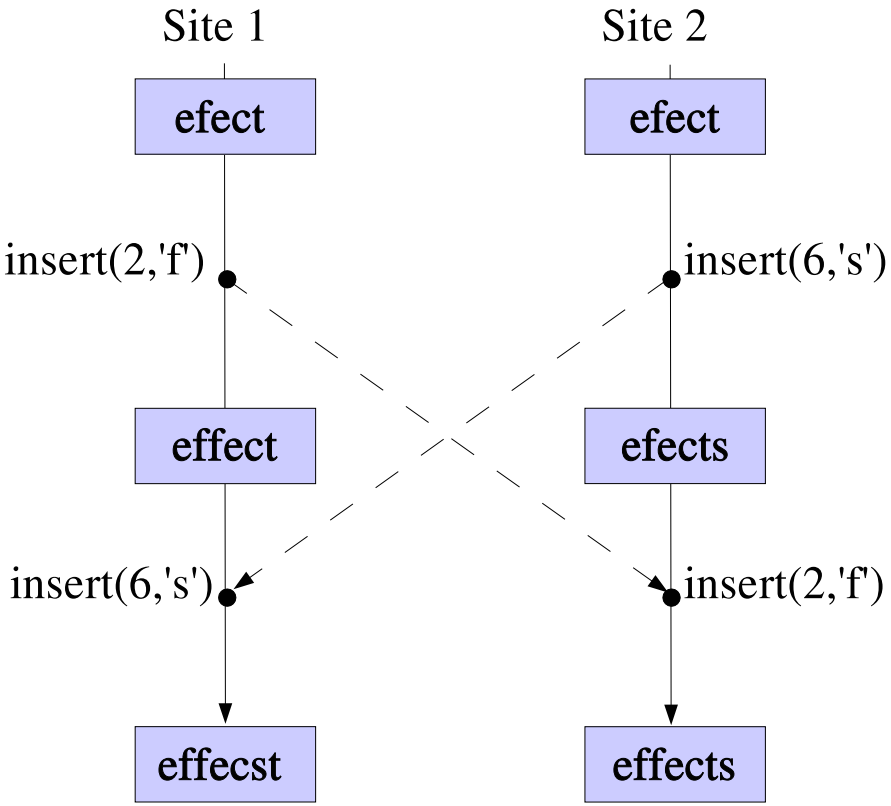
3.2 3 Critères pour le maintien de la cohérence

- 1) Préservation de la causalité :
  - Ordre d'exécution identique sur tous les sites
  - Livraison causale des opérations à l'aide de vecteurs d'état [Mattern, 1989]

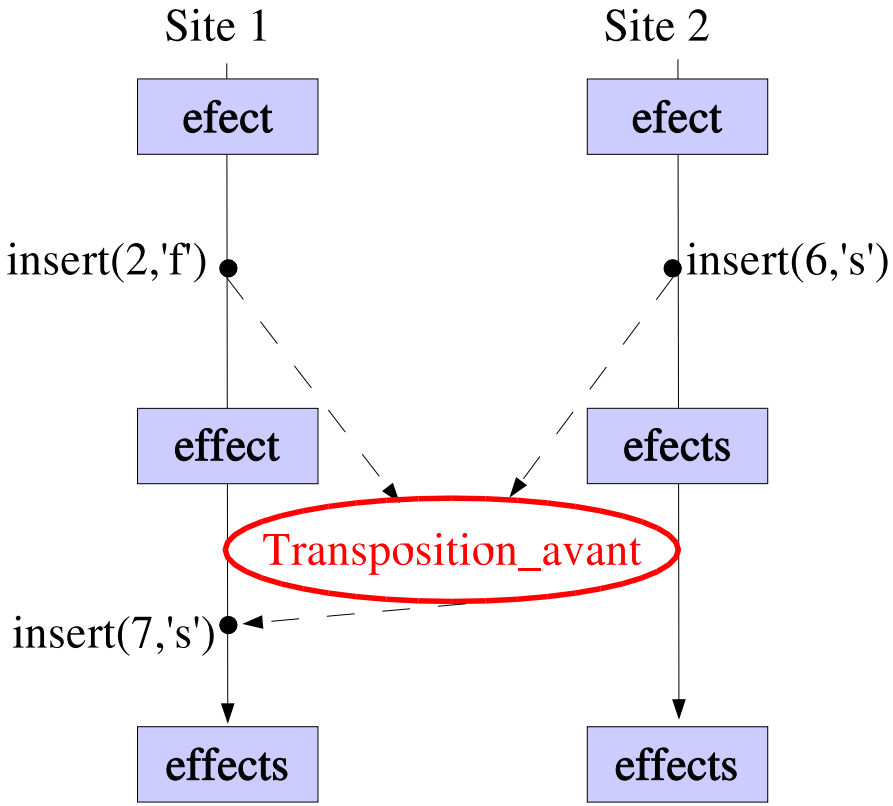


2) Préservation de l'intention de l'utilisateur :

- Gestion des opérations concurrentes (-> transposition avant)



INCOHERENCE!!!



COHERENCE!!!

3) Convergence des copies :

La fonction de transposition avant doit vérifier :

$$\text{C1} : op1.op2^{op1} \equiv op2.op1^{op2}$$

- Résolution des conflits
  - Au moment de l'écriture des transformées
  - Choix arbitraire si possible en garantissant que le même choix sera fait sur tous les sites
  - Sinon, définition d'un conflit avec réconciliation par les utilisateurs

3.3 Exemple : Transformée avant pour un éditeur de texte

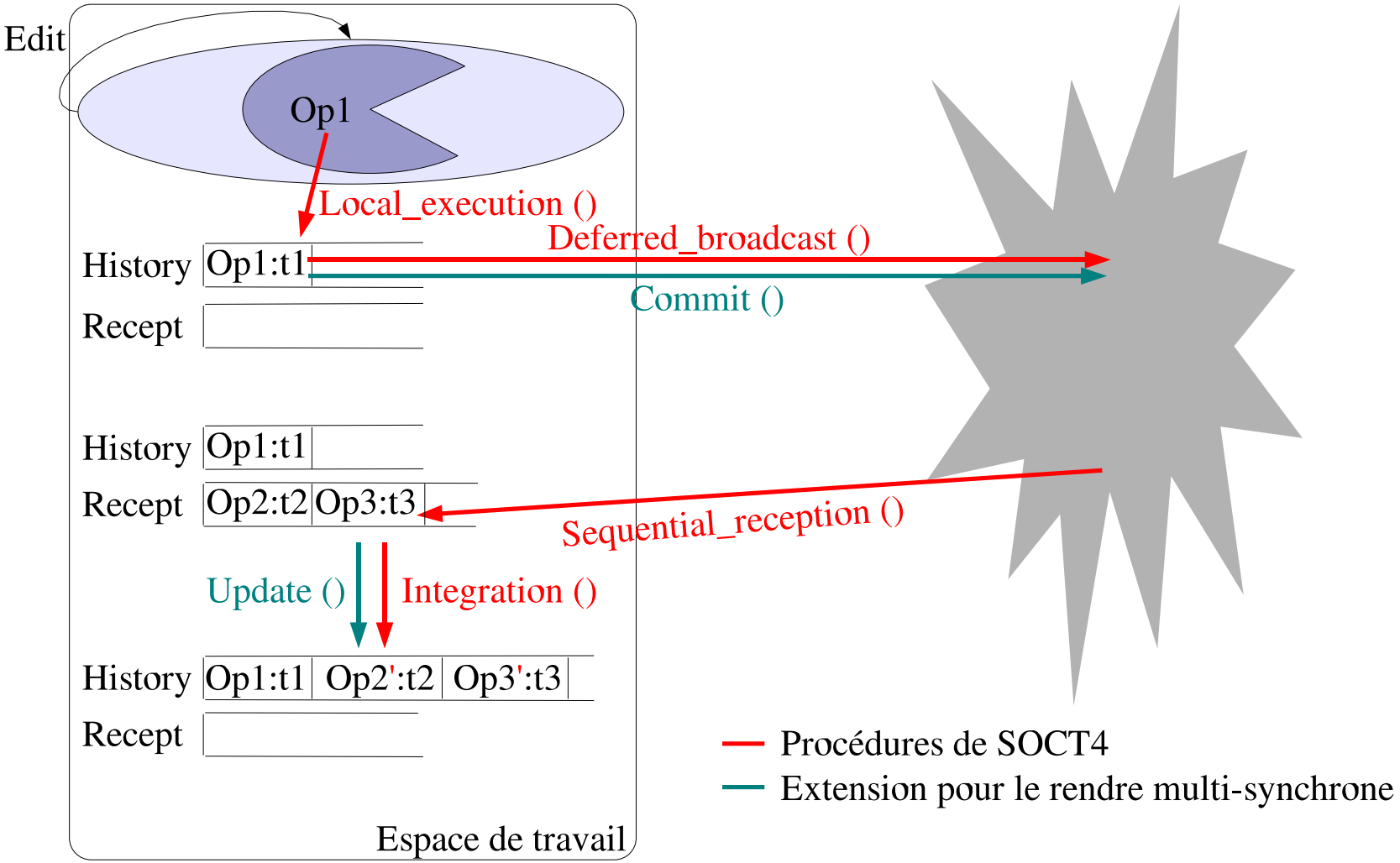
```
ranspose_forward (insert(p1, c1) , insert(p2, c2) ) {
/ Param1 : opération locale , Param 2: opération distante

ase p1 ? p2 of
1 < p2 : return insert(p2 +1, c2);
 p1 > p2 : return insert(p2, c2);
1 = p2 : if  c1 = c2 then return id
   else  // Cas d'un conflit
       // Choix selon valeur du caractère
   if  code (c2) > code (c1)
hen return insert(p2, c2)
  else return insert(p2+1, c2);
ndif;
  endif;
ndcase
```

3.4 Extensions de SOCT4

- SOCT4 suppose une connexion permanente entre les sites
  - Une opération est diffusée immédiatement aux sites distants
- Travail collaboratif multi-synchrone [Bouazza and Molli, 2000]
  - Travail local avec possibilité de mesurer la divergence
  - Synchronisation périodique possible pour faciliter la convergence
  - Modes de travail couplé et découplé, avec basculement d'un mode à l'autre
    - En mode couplé (phases synchrones)
      - ★ utilisation de SOCT4
    - En mode découplé (phases asynchrones)
      - ★ Placement des opérations générées localement dans une file, sans estampille
      - ★ Mise en attente des opérations distantes reçues pour traitement différé
  - L'utilisateur/application décide quand
    - envoyer les opérations locales
    - intégrer les opérations distantes

Extensions de SOCT4 pour le multisynchrone



Extensions de SOCT4

- SOCT4 avec un site mobile [Vidot, 2002]
  - Hypothèse d'un site mandataire fixe
    - Séquenceur centralisé
    - Relais du site mobile pour diffuser les opérations locales
  - Non implémenté

4 Contribution : SOCT4 mobile

Chateigner et al., 2004]

- Pas de site fixe
  - Communication directe entre les sites joignables (Utilisation de Javagroups)
  - Séquenceur réparti
- Déconnexion volontaire
  - Diffusion préalable des opérations locales déjà estampillées
- Pendant la déconnexion
  - Travail isolé sur les composants déconnectés et journalisation des opérations locales
- A la reconnexion
  - Récupération des opérations des autres sites et intégration dans l'histoire
  - Diffusion des opérations effectuées localement après estampillage
- En cours d'implémentation

4.1 Procédure de reconnexion

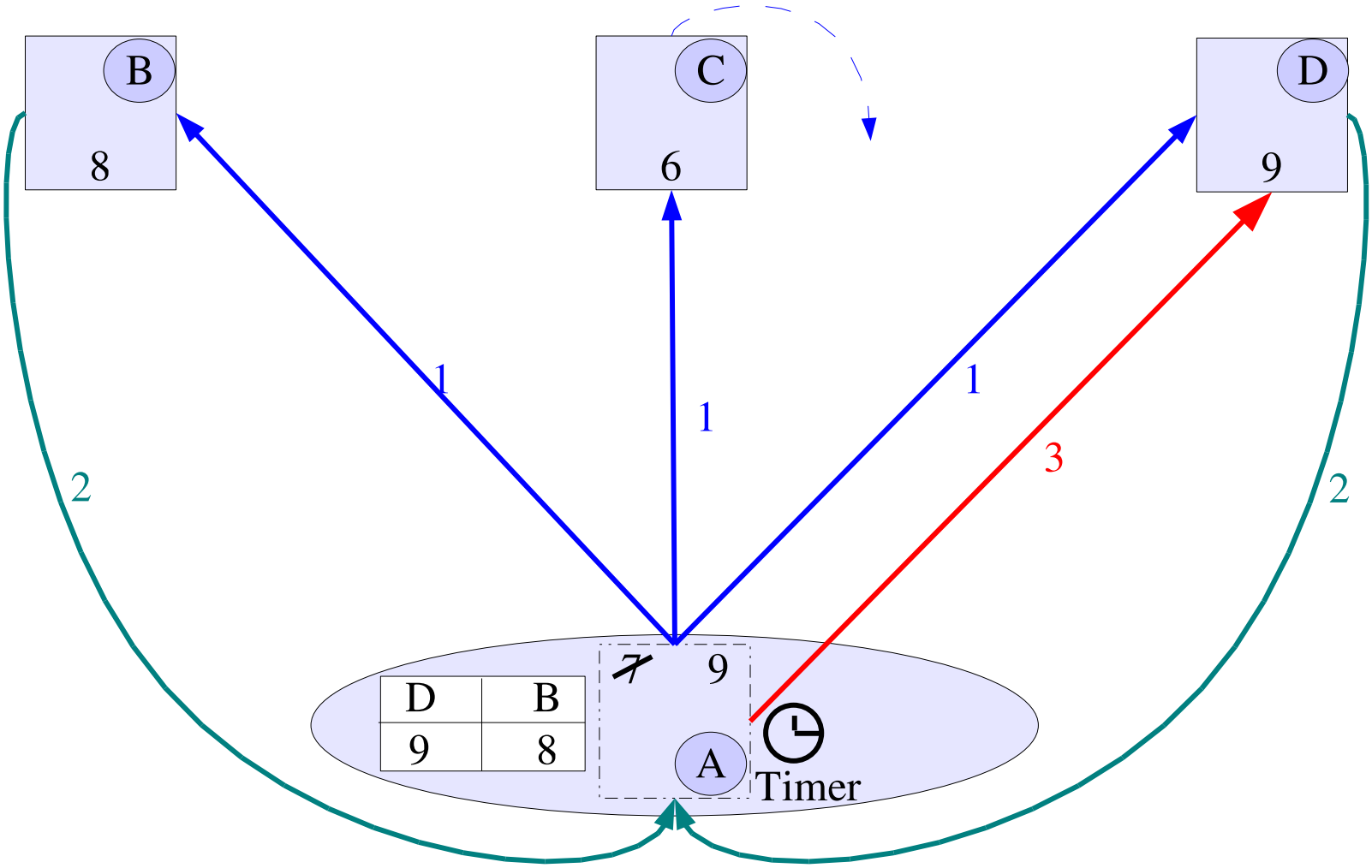
Travail collaboratif multi-synchrone

```
Proc coupler ()
    update();
    commit();
    coupled = true;
endproc
```

Travail collaboratif adapté à la mobilité

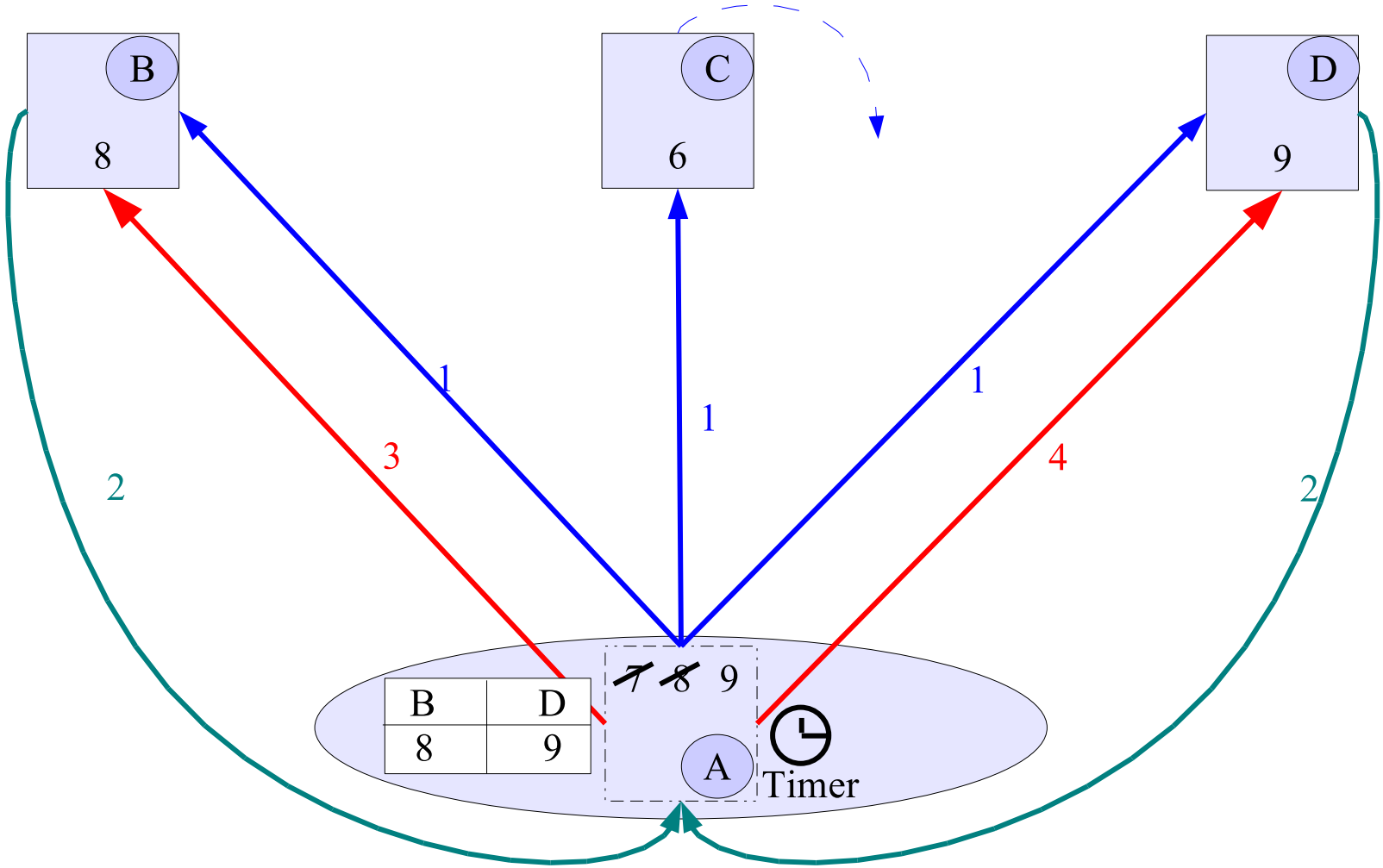
```
Proc startReconcil ()
    coupled = true;
    getTimestamps ();
    propagate(msg);
    broadcast (Ld);
endproc
```

4.2 Procédure de récupération



1 : RequestHistory (Expeditor, History)
2 : SendHistory (Expeditor, History)
3 : RequestDiff (Expeditor, History)

4.3 Procédure de récupération



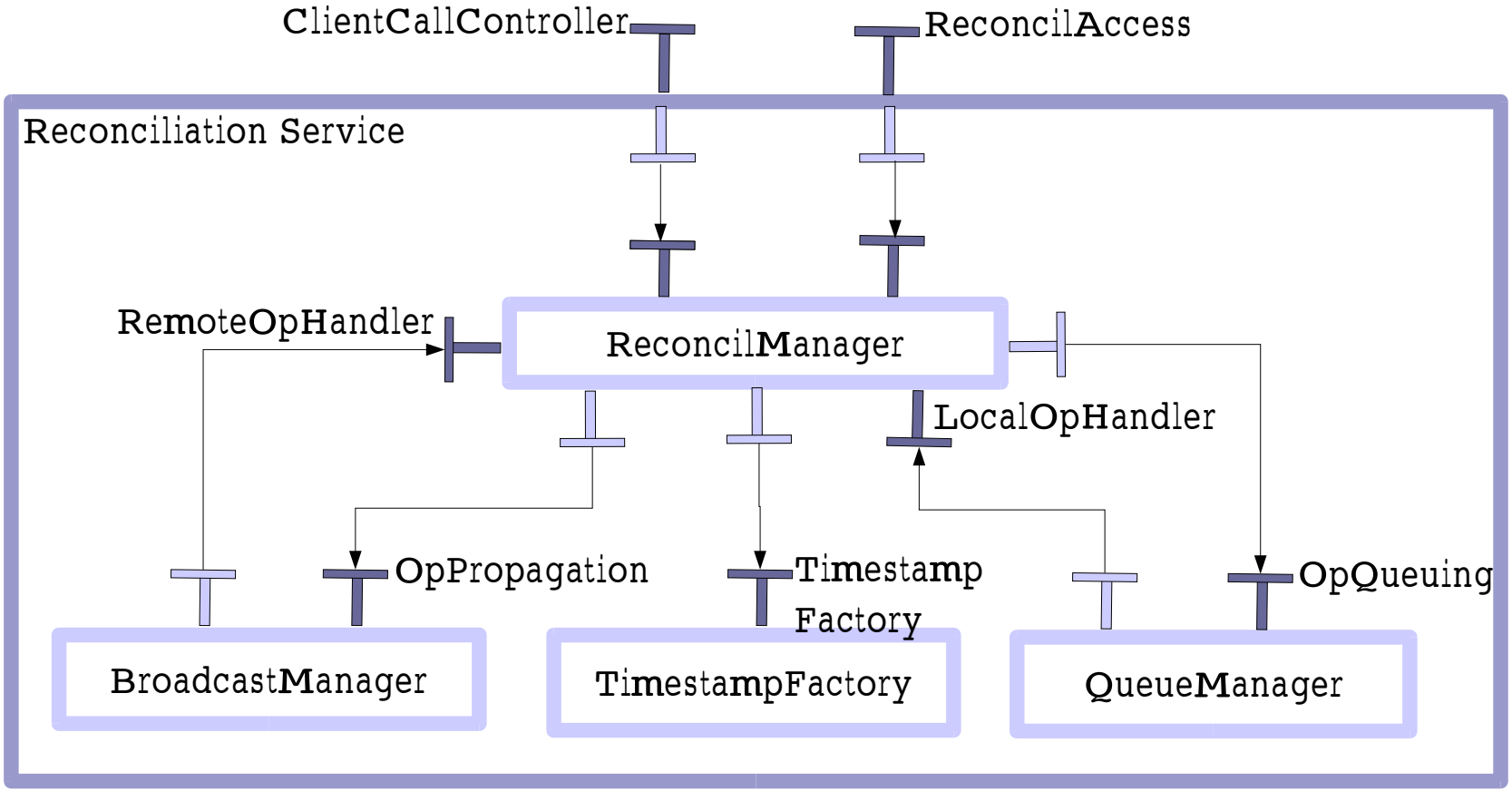
1 : RequestHistory (Expeditor, History)
2 : SendHistory (Expeditor, History)
3 : RequestDiff (Expeditor, History)

4.4 Procédure de récupération

- Un site A, se reconnecte et souhaite récupérer les messages non reçus pendant son absence.
- 1. A diffuse à son groupe la dernière opération intégrée Ld,
- 2. Chaque membre M du groupe connecté reçoit la requête :
  - si Ldlocal = Ld alors M1 ne fait rien
  - si LdLocal > Ld alors M1 renvoie LdLocal
  - si LdLocal < Ld alors lancer procédure de récupération
- 3. A stocke les réponses dans une table.
- 4. A, en fonction de la table, élit les sites primordiaux pour obtenir les opérations manquantes.

5 Architecture du Service de Réconciliation

- Basé sur le modèle de composants CORBA
- Géré comme un service CORBA extra-fonctionnel
- Intégré dans OpenCCM via un conteneur extensible [Vadet, M. et al, 2002]
- Projet ITEA Osmose - Mobility Framework
- Modèle Fractal :



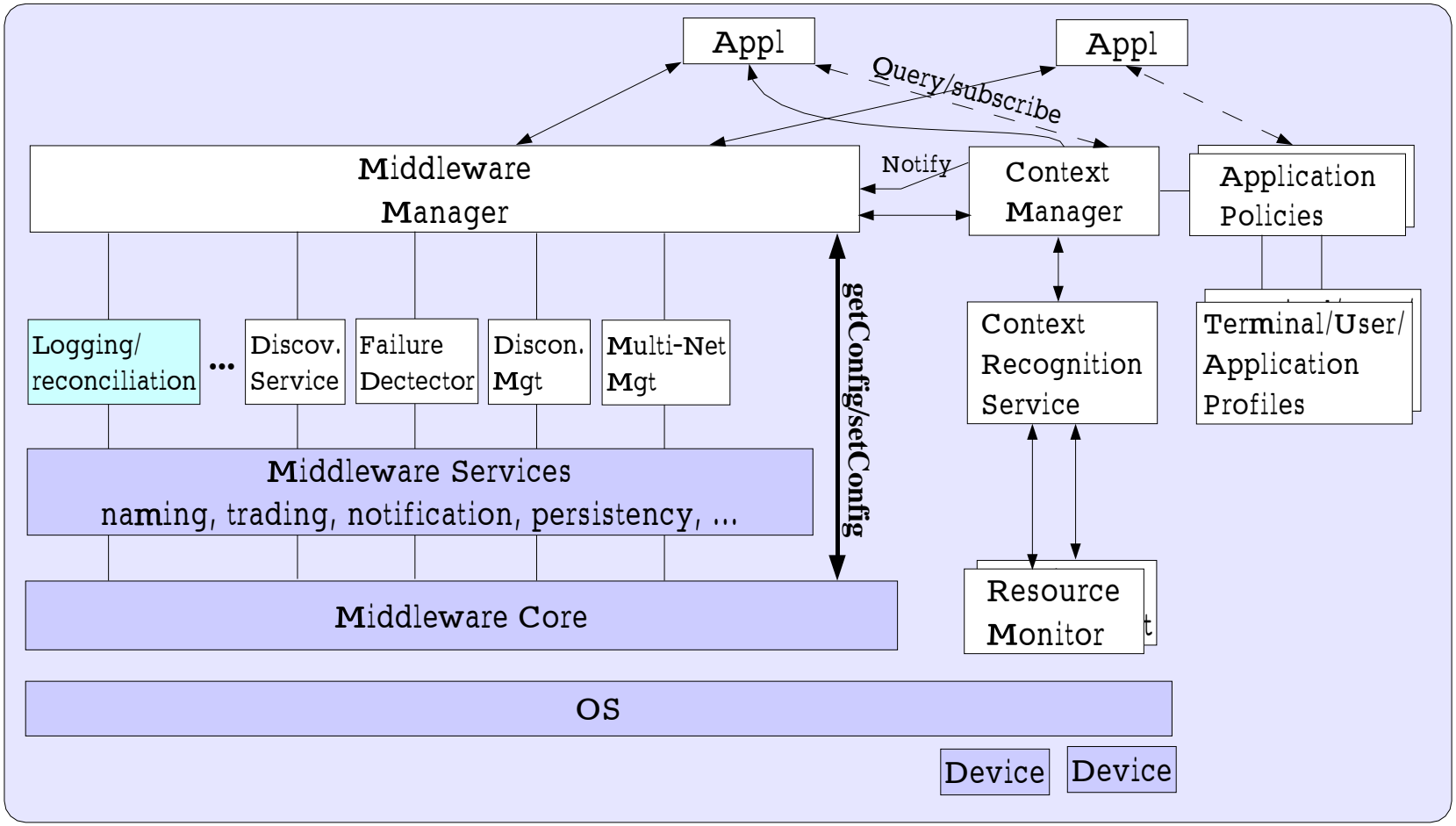
5.1 Description du Service de Réconciliation

- Composant ReconcilManager
  - coordonne tous les sous-composants
  - implémente l'algorithme de réconciliation SOCT4 [Vidot, 2002]
- Composant BroadcastManager
  - gère les communications entre les sites
  - repose sur JavaGroups [JG,]
- Composant TimestampFactory
  - Fournit des tickets selon un ordre séquentiel
  - basé sur un séquenceur réparti tolérant aux fautes [Baldoni et al., 2002]
- Composant QueueManager
  - Gère des files de messages pour les opérations locales et distantes [Chateigner et al., 2003, Chateigner et al., 2004]

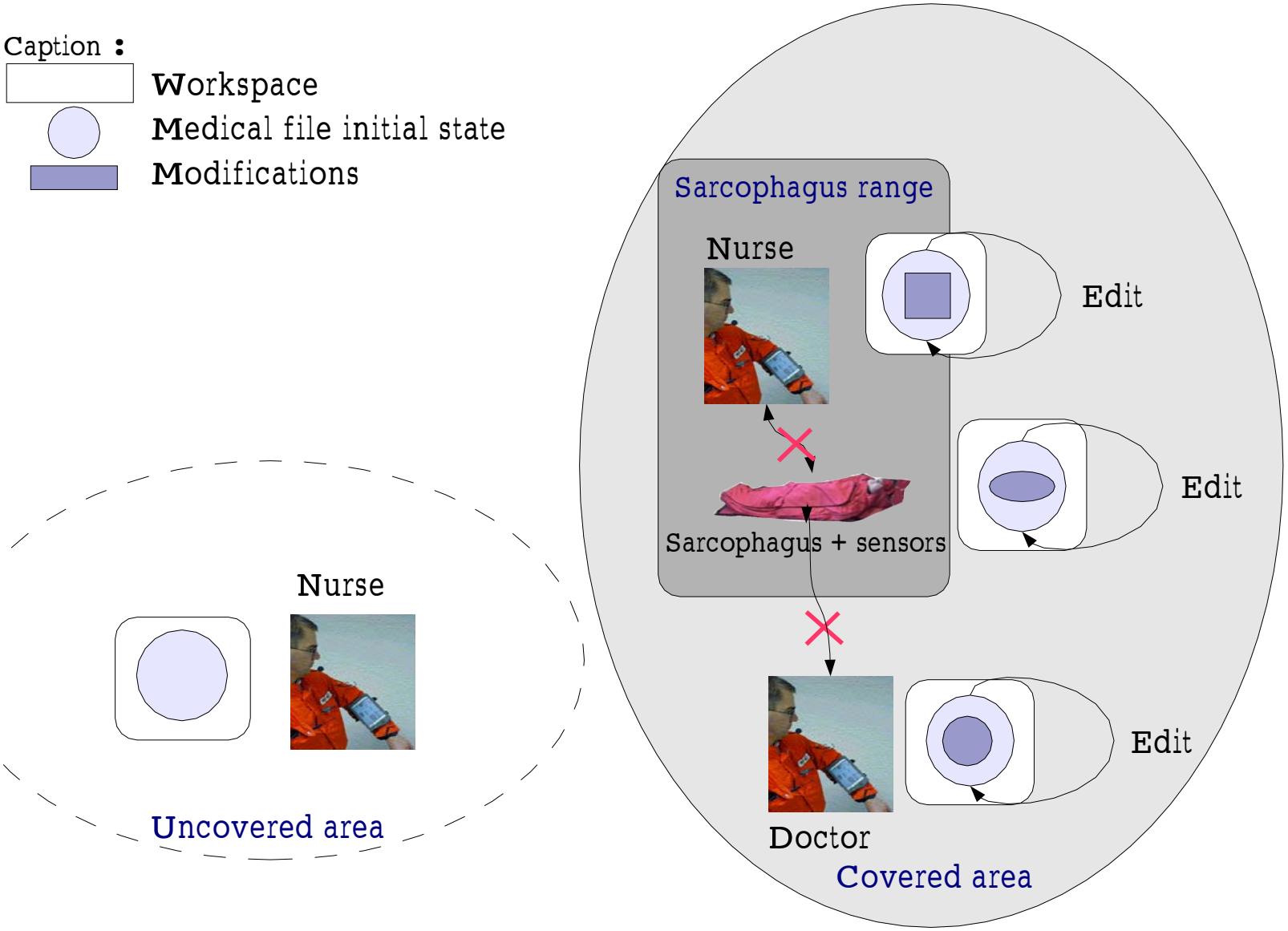
6 Démonstrateur

- Projet AMPROS (Adaptative Middleware Platform for PROactive and reconfigurable Systems) - Programme franco-finlandais RNTL ProACT
- Partenaires : **HUT** (Helsinky University of Technology), **INT** (Institut National des Télécommunications), et **Thalès Communications**
- Objectif : Fournir un ensemble de services intergiciels basés sur des composants CORBA pour
  - la reconfiguration dynamique
  - le déploiement sensible au contexte
  - la gestion des déconnexions en environnement mobile

6.1 Architecture globale de la plate-forme Ampros



6.2 Gestion de catastrophe



Optimistic replication => Divergence => Reconciliation

6.3 Scenario de suivi médical des victimes avec déconnexion

- Hypothèse : réplication des fiches médicales des victimes
  - où : au niveau des équipements des victimes et des PDA des sauveteurs
  - Pourquoi : performances et continuïté de service
- Problème : divergence sur les fiches médicales
  - en raison des communications
    - Mobilité des sauveteurs
    - Portée limitée des communications sans fil
  - en raison des écritures multiples
    - Mise à jour de la fiche médicale directement à partir des capteurs équipant la victime
    - Intervention des sauveteurs
    - Intervention de plusieurs sauveteurs au cours du temps
  - Besoin : Réconciliation des fiches médicales

6.4 Déroulement

- Création d'une fiche médicale au niveau de l'équipement de la victime
- Attribution d'un numéro unique (différé si connectivité insuffisante)
- Cas d'une déconnexion du sauveteur après téléchargement de la fiche
  - Phase 0 - Initialisation du terminal
    - Selon le profil du secouriste, déploiement et démarrage des services appropriés
    - Démande éventuelle de services supplémentaires, si le gestionnaire de cache l'autorise
    - Synchronisation régulière des composants déconnectés présents dans le cache avec les composants distants
  - Phase 1 - Travail local suite à déconnexion
    - Si déconnexion volontaire, mise à jour du cache
    - Si déconnexion subite, cache en l'état
    - Opérations exécutées localement puis placées sans estampille dans une file d'opérations

- Phase 2 - Reconnexion
  - Récupération des opérations distantes dans la file de réception puis intégration selon l'ordre global continu
  - Estampillage des opérations locales en attente puis diffusion
- Phase 3 - Collaboration (avec connectivité satisfaisante)
  - Une opération locale est immédiatement exécutée puis placée en attente sans estampille
  - Demande d'une estampille au séquenceur
  - Réception puis intégration des opérations distantes au fil de l'eau
  - Diffusion de l'opération locale après intégration des opérations concurrentes qui la précèdent dans l'ordre global continu

7 Conclusion et perspectives

- Contributions à SOCT4 :
  - Gestion de la mobilité (procédure de récupération)
  - Utilisation d'un séquenceur réparti tolérant aux fautes
- Service extra-fonctionnel d'un middleware à base de composants
  - Repose sur le modèle de conteneur ouvert d'OpenCCM
- Perspectives :
  - Gestion du mode partiellement connecté
  - Aide à la génération des transformées
  - Limitation de la divergence - Lien avec solutions utilisées dans le domaine des jeux
  - Mise en oeuvre et validation

References

[G,] JGroups Home page. `http ://www.jgroups.org/javagroupsnew/docs/index.html`.

[Baldoni et al., 2002] Baldoni, R., Marchetti, C., and Tucci Pergiovanni, S. (2002). A Fault-Tolerant Sequencer for Timed Asynchronous Systems. In *In Euro-Par 2002, LNCS 2400*, Berlin Heidelberg, Germany.

[Bouazza and Molli, 2000] Bouazza, A. and Molli, P. (Philadelphia, Pennsylvania, USA, Décembre 2000). Unifying coupled and uncoupled collaborative work in virtual teams. *In ACM CSCW workshop on collaborative editing systems*.

[Chateigner et al., 2003] Chateigner, L., Chabridon, S., and Bernard, G. (Octobre 2003). Intergiciel pour l'informatique nomade : réplication optimiste et réconciliation. In *In Manifestation des jeunes chercheurs STIC, MAJECSTIC*, Marseille, France.

[Chateigner et al., 2004] Chateigner, L., Chabridon, S., Sabri, N., and Bernard, G. (2004). Service de réconciliation pour la synchronisation de copies. In *Actes de la 1ère*, Nice, France.

[Kermarrec et al., 2001] Kermarrec, A., Rowstron, A., Shapiro, M., and Druschel, P. (26-29 Août 2001). The icecube approach to the reconciliation of divergent replicas. In *Proceedings of the 20th ACM Symposium on Principles of Distributed Computing (PODC 2001)*, Newport, Rhode Island (USA).

[Mattern, 1989] Mattern, F. (1989). Virtual Time and Global States of Distributed Systems. In *Proceedings of the International Workshop on Parallel and Distributed Algorithms*, Amsterdam, NH.

[Pucheral, P. et al., 2004] Pucheral, P. et al. (2004). Mobile Databases : a Selection of Open Issues and Research Directions. *ACM SIGMOD Record*.

[Saito, Y. and Shapiro, M.,] Saito, Y. and Shapiro, M. Optimistic Replication, MSR-TR-2003-60, 2003. Technical report.

[Terry et al., 1995] Terry, D. B., Theimer, M. M., Petersen, K., and Demers, A. J. (1995). Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System. In *Proc. 15th*, pages 172–183.

[Vadet, M. et al, 2002] Vadet, M. et al (September, 2002). IST COACH (Component Based Open Source Architecture for Distributed Telecom Applications) PROJECT. `http ://www.ist-coach.org`.

[Vidot, 2002] Vidot, N. (Septembre 2002). Convergence des Copies dans les Environnements Collaboratifs Répartis. Thèse de doctorat, Université Montpellier-2, Montpellier, France.