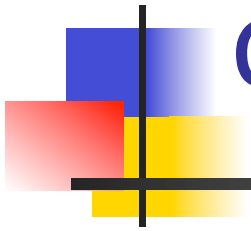
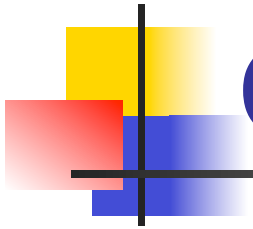


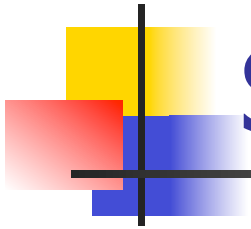
Synchronisation de données divergentes

Pascal Molli ,Maître de Conférence
Université Henri Poincaré, Nancy 1
Projet Inria ECOO



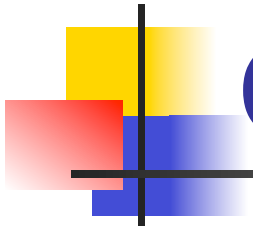
Contexte

- Travail mobile
  - Avant la déconnection -> réplication
  - A la reconnection -> synchronisation
- Travail de groupe
  - Avant l'insularisation -> réplication
  - Avant la publication -> réconciliation



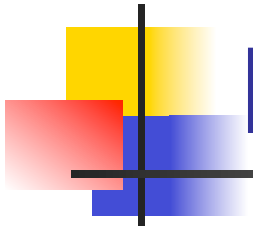
Synchroniser ??

- Copies identiques après synchro…
- Mais quel état de convergence ?
  - Selon quel critère critère de correction (~sérialisabilité)
- Quelle granularité de réconciliation ?
- Pour quels types d'objets ?
  - Entier, texte, XML, tables… même synchroniseur ?
- Qui résout les conflits?
  - Système ? Utilisateur ? administrateur ?
- Quand résoudre les conflits ? Pendant la synchro ou après ??



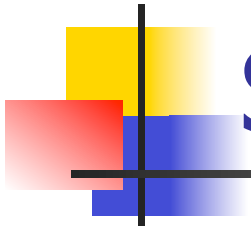
Objectif

- Créer un synchroniseur
  - Sûr
    - critère de correction
  - Générique
    - Même algo qqsoit le type des données partagées
  - À fine granularité
- Créer des réseaux de synchronisation
  - Modélisation de procédés « dataflow »



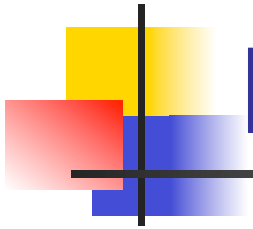
État de l'art

- Synchroniseurs de fichiers
  - Unisson, IntelliSync, Microsoft Synchronizer
- Synchroniseurs de données
  - ActiveSync, Hotsync, I-Sync
- Système distribués
  - CODA, Bayou, Ficus, IceCube
- Basé de données répliquées
  - Réplication asymétrique



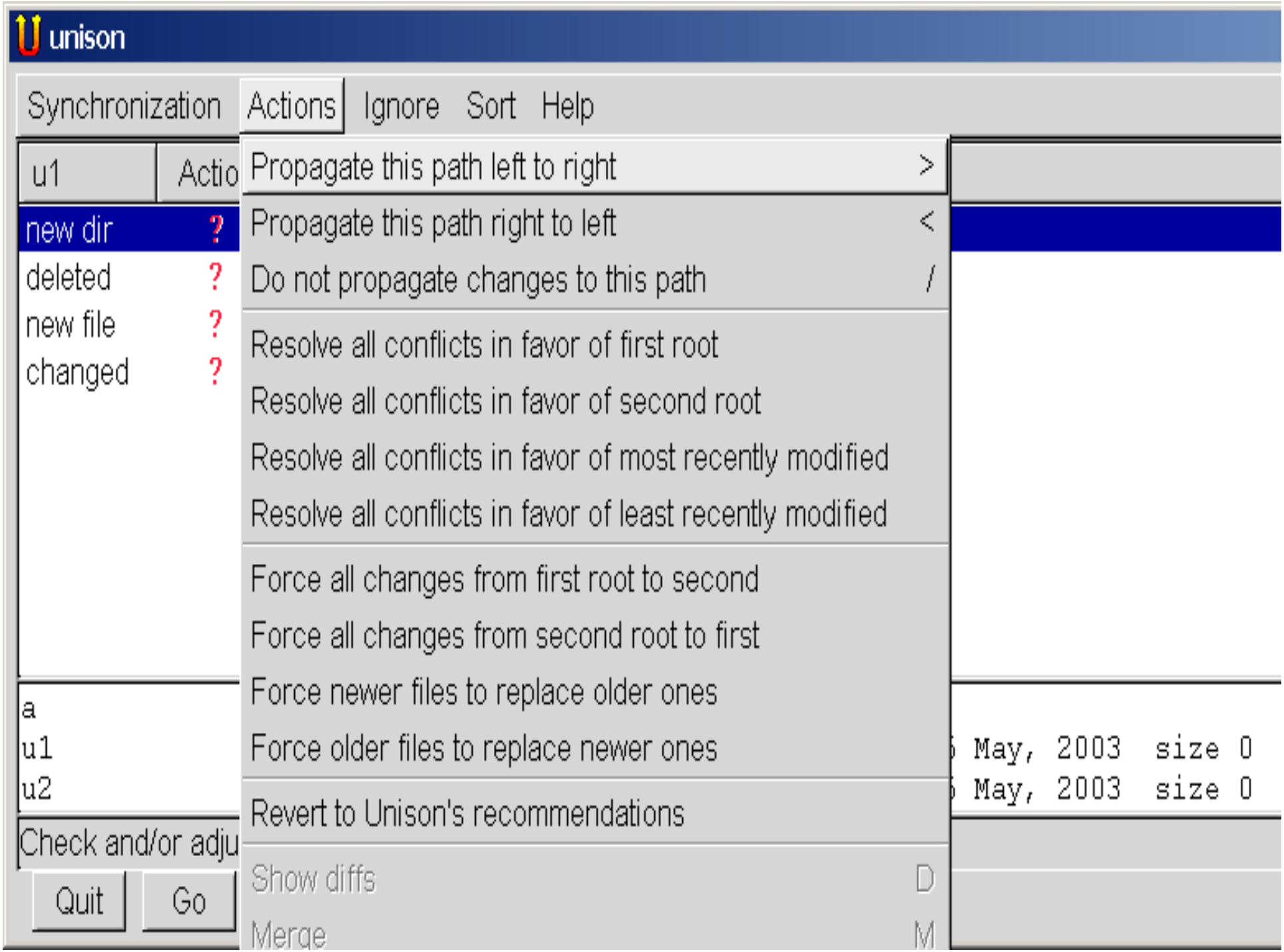
Synchroniseur de fichiers

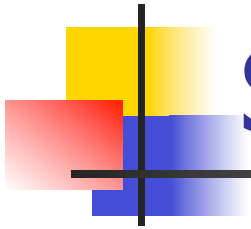
- Unisson, rsync, Intellisync, Windows Synchronizer
- Propager les mises-à-jour non-conflictuelles
- Déléguer aux utilisateurs la résolution des conflits



Exemples de conflits

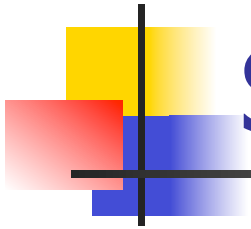
- User1
  - MkFile (/a)
- User2
  - Mkdir (/a)
  - Synchronize(/)
- User1
  - Edit(/a/b)
  - Synchronize(/)
- User2
  - Rm(/a)


unison
Synchronization
Actions
Ignore
Sort
Help
u1
Actio
new dir ?
deleted ?
new file ?
changed ?
Propagate this path left to right >
Propagate this path right to left <
Do not propagate changes to this path /
Resolve all conflicts in favor of first root
Resolve all conflicts in favor of second root
Resolve all conflicts in favor of most recently modified
Resolve all conflicts in favor of least recently modified
Force all changes from first root to second
Force all changes from second root to first
Force newer files to replace older ones
Force older files to replace newer ones
Revert to Unison's recommendations
Show diffs D
Merge M
a
u1
u2
May, 2003 size 0
May, 2003 size 0
Check and/or adju
Quit
Go



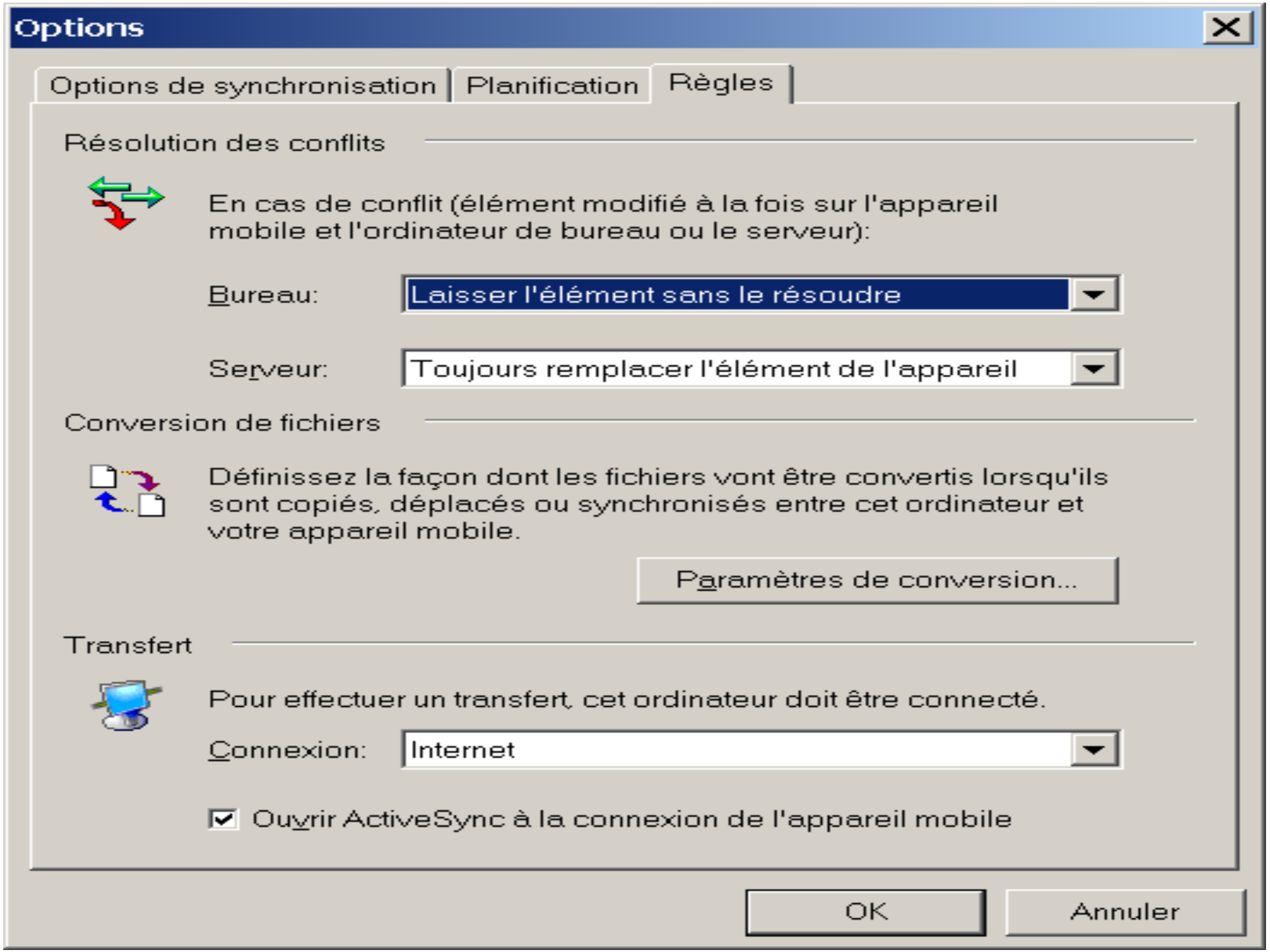
Synchroniseurs de fichiers

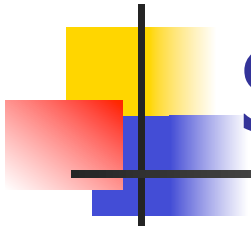
- PB: Granularité de conflits
  - Ajouter un paragraphe / Corriger les fôtes d'ortografe…
- Convergence grossière vers un état sans conflit



Synchroniseurs de données

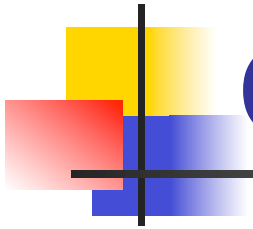
- ActiveSync (microsoft), HotSync (Palm), I-Sync (apple)…
- ~synchroniseurs de fichiers
- + connaissance de certains types de fichiers…


Options
Options de synchronisation
Planification
Règles
Résolution des conflits
En cas de conflit (élément modifié à la fois sur l'appareil mobile et l'ordinateur de bureau ou le serveur):
Bureau:
Laisser l'élément sans le résoudre
Serveur:
Toujours remplacer l'élément de l'appareil
Conversion de fichiers
Définissez la façon dont les fichiers vont être convertis lorsqu'ils sont copiés, déplacés ou synchronisés entre cet ordinateur et votre appareil mobile.
Paramètres de conversion...
Transfert
Pour effectuer un transfert, cet ordinateur doit être connecté.
Connexion:
Internet
Ouvrir ActiveSync à la connexion de l'appareil mobile
OK
Annuler



Synchroniseurs de données

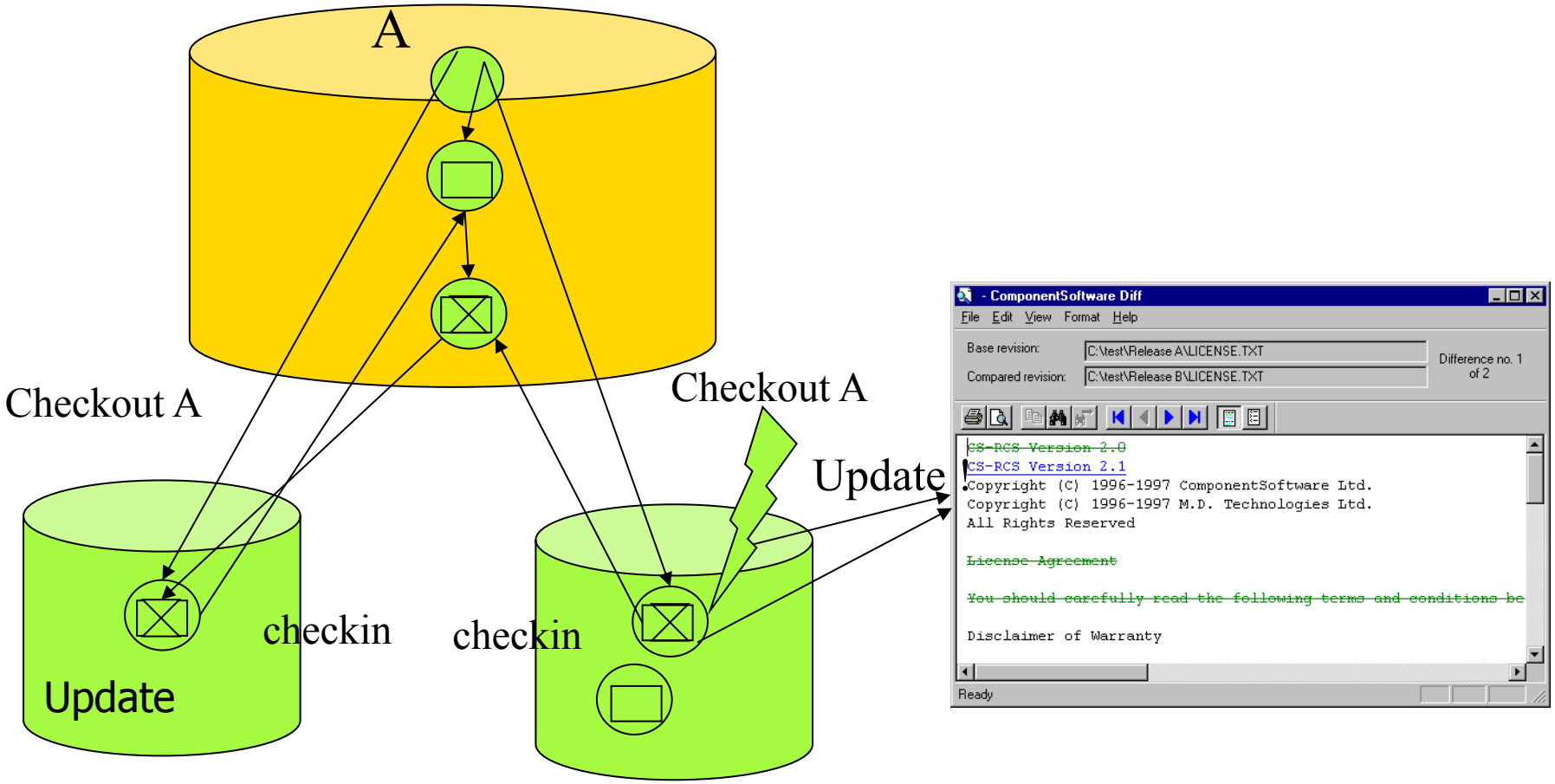
- PB: généricité.
- Certains conflits sont résolus selon le type de fichier…
- Convergence vers un état pouvant contenir des conflits

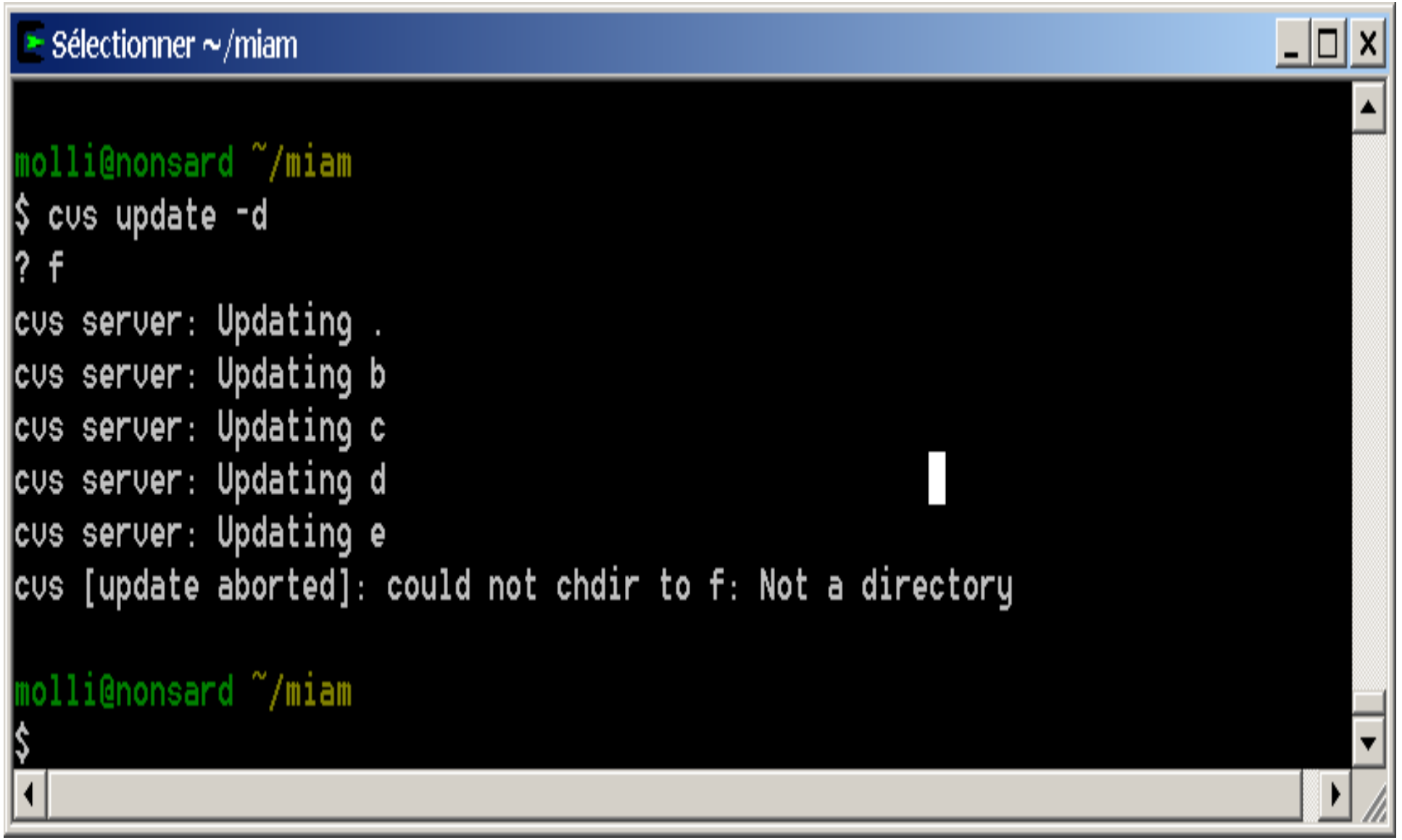


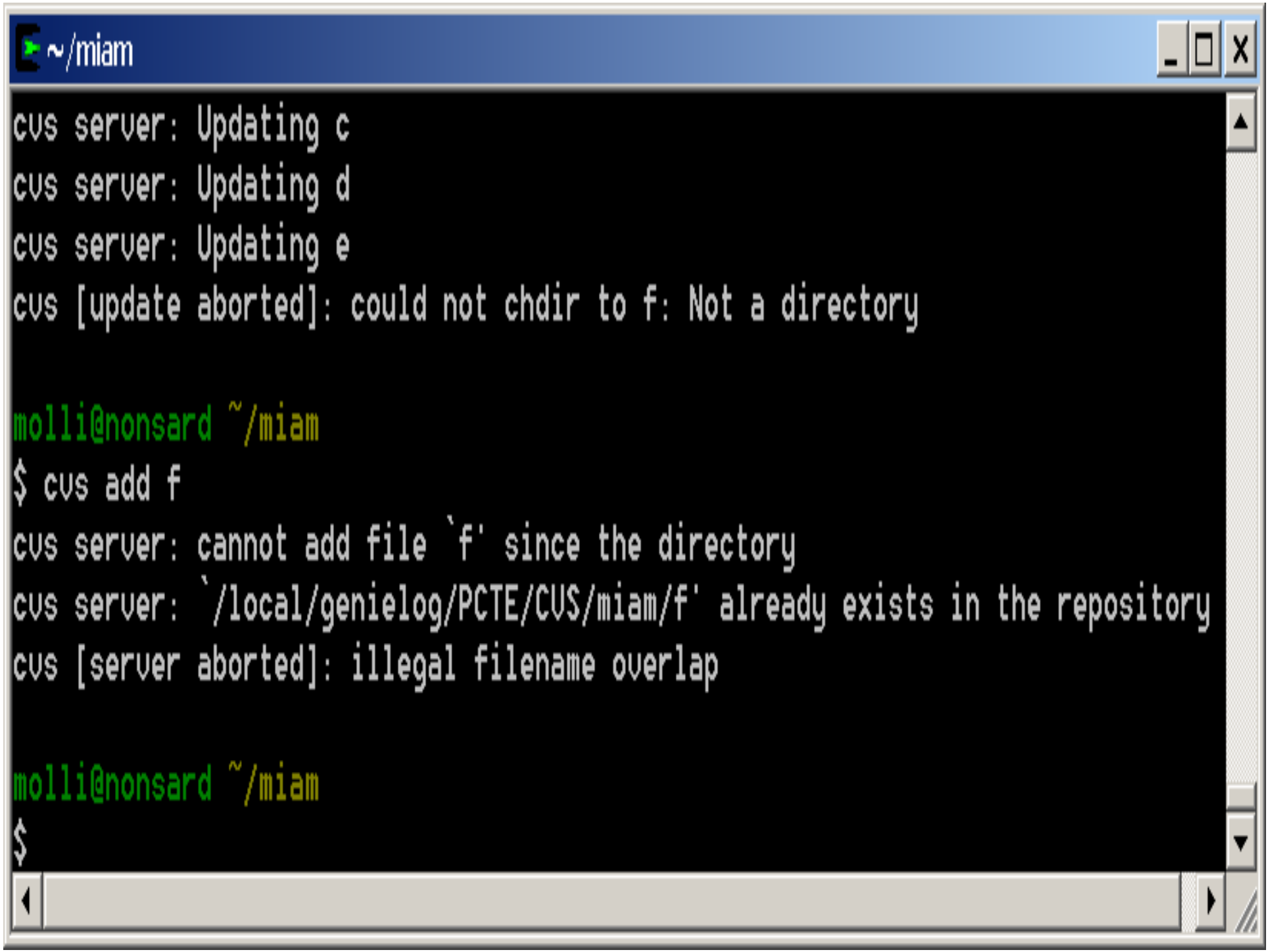
CM Tools

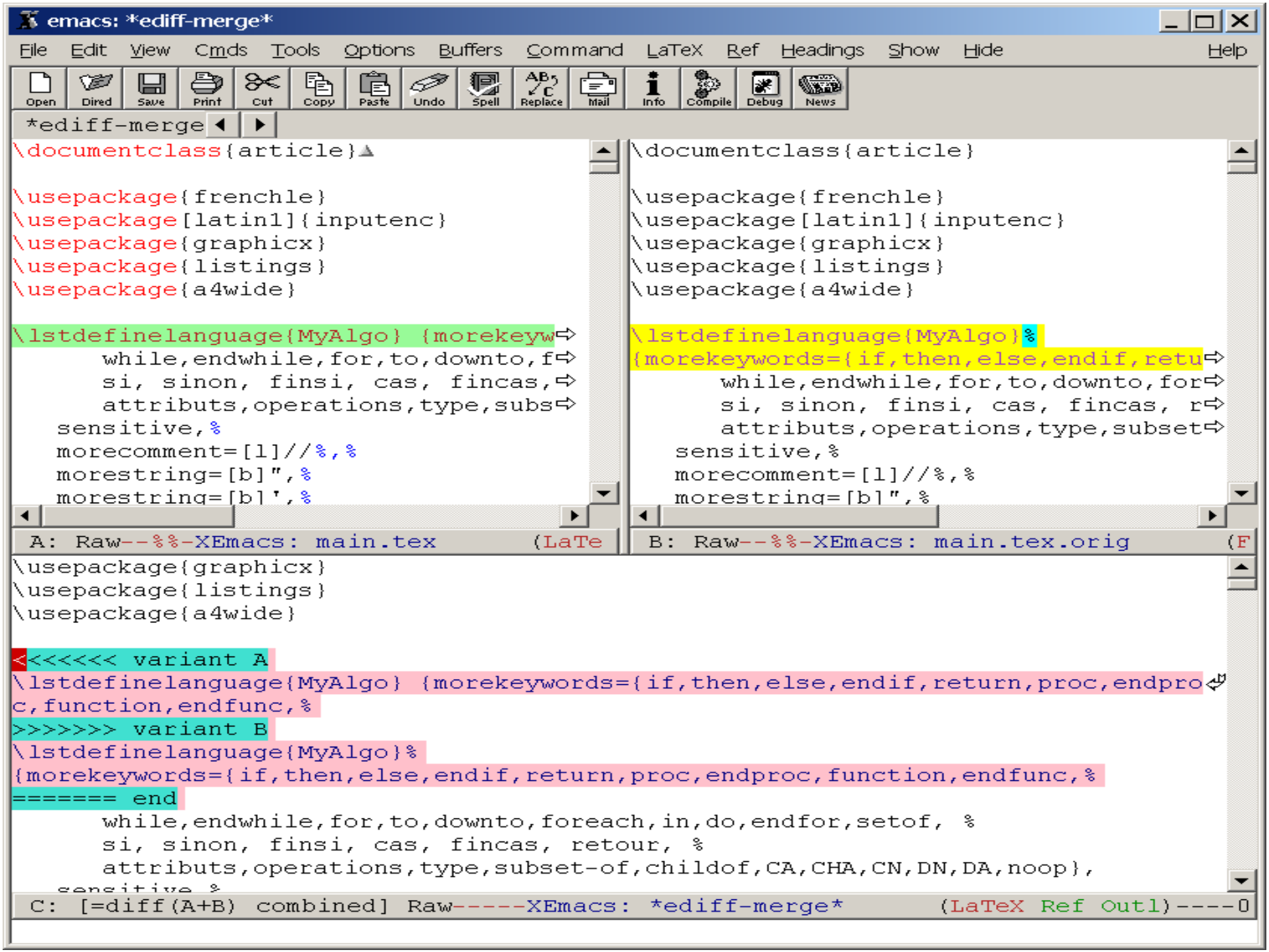
- CVS, ClearClase, SourceSafe…
- Assure la réconciliation de certains types de conflits:
  - Update-Update sur fichiers textes ou XML
    - (algo diff3, XML Merge)
  - Mais pas tous…
    - Rmdir/update file
    - (mkdir/mkfile…)
  - Convergence non garantie dans tous les cas…

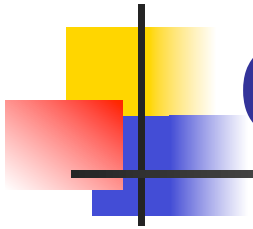
Copy-Modify-Merge




Sélectionner ~/miam
molli@nonsard ~/miam
$ cvs update -d
? f
cvs server: Updating .
cvs server: Updating b
cvs server: Updating c
cvs server: Updating d
cvs server: Updating e
cvs [update aborted]: could not chdir to f: Not a directory
molli@nonsard ~/miam
$

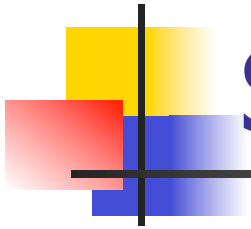

~/miam
cvs server: Updating c
cvs server: Updating d
cvs server: Updating e
cvs [update aborted]: could not chdir to f: Not a directory
molli@nonsard ~/miam
$ cvs add f
cvs server: cannot add file `f' since the directory
cvs server: `/local/genielog/PCTE/CVS/miam/f' already exists in the repository
cvs [server aborted]: illegal filename overlap
molli@nonsard ~/miam
$


emacs: *ediff-merge*
File Edit View Cmds Tools Options Buffers Command LaTeX Ref Headings Show Hide Help
Open Dired Save Print Cut Copy Paste Undo Spell Replace Mail Info Compile Debug News
*ediff-merge
\documentclass{article}
\usepackage{frenchle}
\usepackage[latin1]{inputenc}
\usepackage{graphicx}
\usepackage{listings}
\usepackage{a4wide}
\lstdefinelanguage{MyAlgo} {morekeyw
while,endwhile,for,to,downto,f
si, sinon, finsi, cas, fincas,
attributs,operations,type,subs
sensitive,%
morecomment=[l]//%,%
morestring=[b]",%
morestring=[b]',%
A: Raw--%%-XEmacs: main.tex (LaTe
\documentclass{article}
\usepackage{frenchle}
\usepackage[latin1]{inputenc}
\usepackage{graphicx}
\usepackage{listings}
\usepackage{a4wide}
\lstdefinelanguage{MyAlgo}%
{morekeywords={if,then,else,endif,retu
while,endwhile,for,to,downto,for
si, sinon, finsi, cas, fincas, r
attributs,operations,type,subset
sensitive,%
morecomment=[l]//%,%
morestring=[b]",%
B: Raw--%%-XEmacs: main.tex.orig (F
\usepackage{graphicx}
\usepackage{listings}
\usepackage{a4wide}
<<<<<< variant A
\lstdefinelanguage{MyAlgo} {morekeywords={if,then,else,endif,return,proc,endpro
c,function,endfunc,%
>>>>>>> variant B
\lstdefinelanguage{MyAlgo}%
{morekeywords={if,then,else,endif,return,proc,endproc,function,endfunc,%
======= end
while,endwhile,for,to,downto,foreach,in,do,endfor,setof, %
si, sinon, finsi, cas, fincas, retour, %
attributs,operations,type,subset-of,childof,CA,CHA,CN,DN,DA,noop},
C: [=diff(A+B) combined] Raw-----XEmacs: *ediff-merge* (LaTeX Ref Outl)----0



QQ idées tout de même

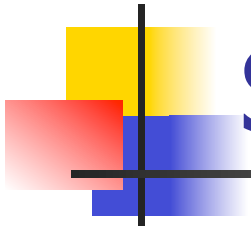
- PB:
  - Certain conflits résolus selon le principe de compensation (diff3)
  - Convergence vers un état avec conflits
  - D'autres sont délégués à l'utilisateur…
- Pas de convergence dans tous les cas…



Systèmes distribués

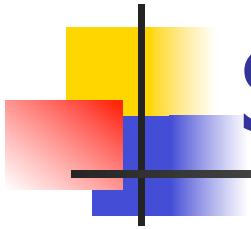
- Coda,Bayou,Ficus, Inter-mezzo
- Réplication et cohérence faible
  - Algorithmes épidémiques
- Propager les mises-à-jour non conflictuelles
- Exécuter des procédures de fusions sur les MAJ Conflictuelles
- Si elles échouent -> résolution de conflits déléguées aux utilisateurs…

```
Bayou_Write(
 update = {insert, Meetings, 12/18/95, 1:30pm, 60min, ``Budget Meeting"},
 dependency_check = {
    query = ``SELECT key FROM Meetings WHERE day = 12/18/95
            AND start < 2:30pm AND end > 1:30pm",
    expected_result = EMPTY},
 mergeproc = {
   alternates = {{12/18/95, 3:00pm}, {12/19/95, 9:30am}};
   newupdate = {};
   FOREACH a IN alternates {
   # check if there would be a conflict
   IF (NOT EMPTY (
   SELECT key FROM Meetings WHERE day = a.date
   AND start < a.time + 60min AND end > a.time))
   CONTINUE;
   # no conflict, can schedule meeting at that time
   newupdate = {insert, Meetings, a.date, a.time, 60min, ``Budget Meeting"};
   BREAK;
   }
   IF (newupdate = {}) # no alternate is acceptable
   newupdate = {insert, ErrorLog, 12/18/95, 1:30pm, 60min, ``Budget Meeting"};
RETURN newupdate;}
)
```



Systèmes Distribués

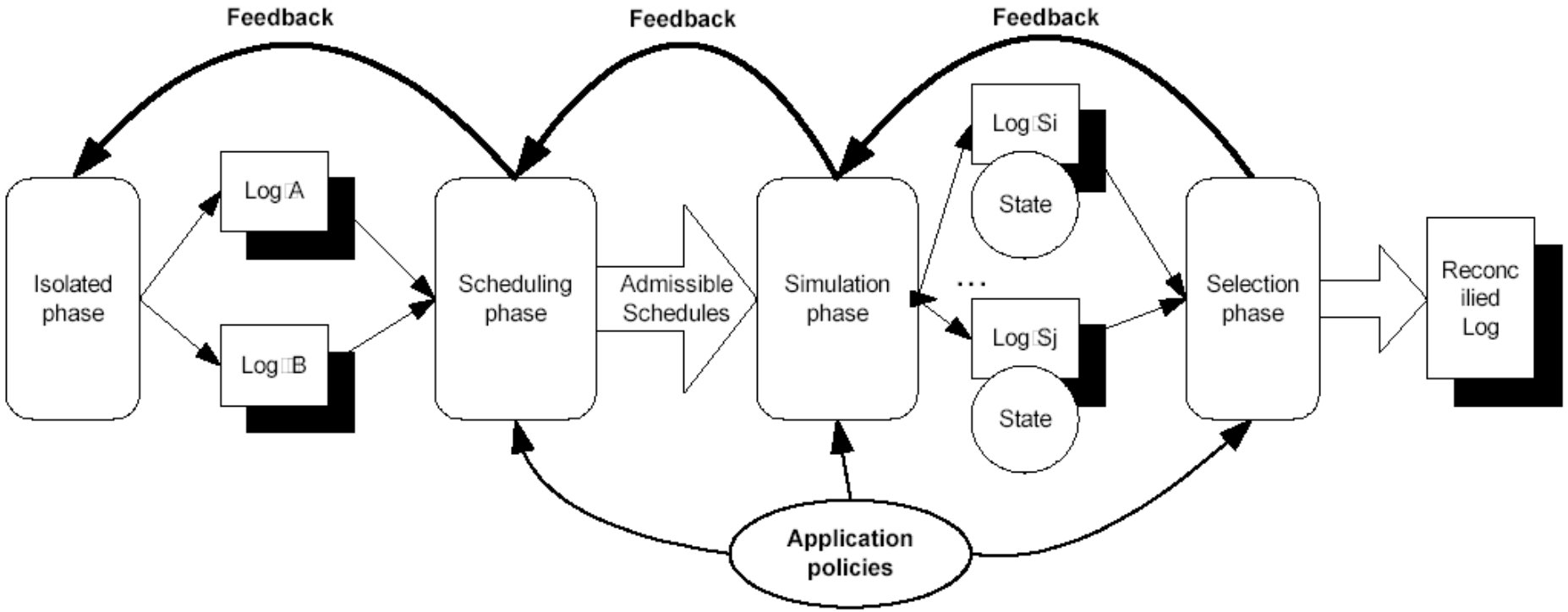
- PB: les conflits non résolus sont délégués à un administrateur
- Les données en attendant sont
  - soit gelées
  - soit elles n’ont pas convergées.

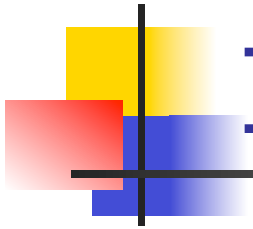


Système Distribué: IceCube

- Basé opération
  - Input: 2 journaux d'opérations concurrentes
  - Tests de toutes les histoires possibles
  - Élimination des histoires ne respectant pas certaines contraintes
  - Choix parmi les histoires restantes…
- Minimise les conflits

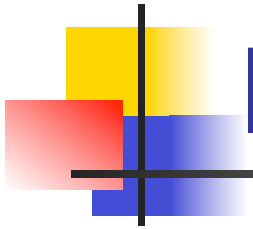
Approche IceCube





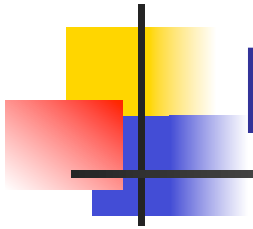
IceCube

- PB:
  - Explosion combinatoire
  - Opérations non modifiées ->
    - Conflits non résolus délégués aux utilisateurs…



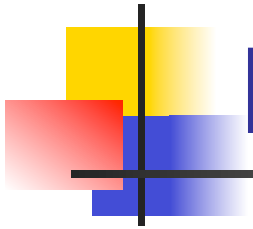
BD Répliquées

- Oracle, Sybase (Advanced replication)
- Plusieurs master-sites
  - Mêmes tuples
    - Modification en parallèle
      - Un conflit !



BD Répliquées

- Dans Oracle:
  - Détection automatique
    - Conflit de mise à jour,
    - Conflit d'unicité
    - Conflit de destruction
  - Résolution basée sur des procédures de fusions



BD répliquées

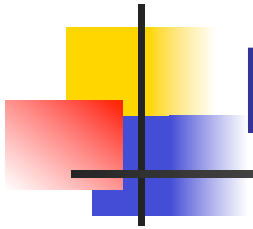
- Procédures de fusion pour conflit de mise à jour
  - Basé sur « latest timestamp »
    - Convergence garanties
    - (comme dans Lotus)
  - Ou sur « overwrite »
    - Convergence non garantie

*: values always increase

**: ordered update value

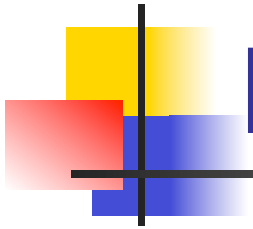
Bd répliquées (fusion update)

Procédure de fusion	Convergence
Additive	Yes
Average	NO
Discard	NO
Earliest TimeStamp	NO
Max	YES*
Min	YES*
Priority Group	YES**
Site Priority	NO



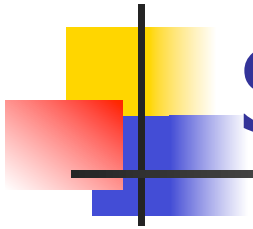
BD répliquées

- Fusion pour les conflits d'unicité
  - Append the global site name of the originating site to the column value from the originating site.
  - Append a generated sequence number to the column value from the originating site.
  - Discard the row value from the originating site.



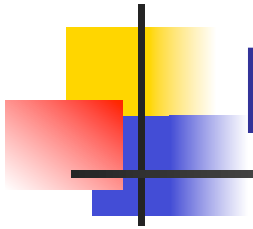
BD répliquées

- Fusion pour les conflits de destruction
  - Pas de procédure !
- Finalement:
  - Pas de convergence garantie dans le cas général !!



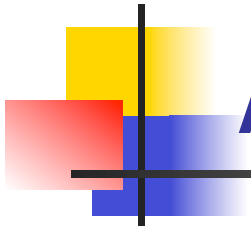
Synthèse

- Propagation des mises à jour non conflictuelles
- Tentative de résolution automatique pour certains conflits
- Si échec -> Résolution
  - par les utilisateurs (CM)
  - par un administrateur (le reste)
  - Soit pendant la synchro (interactif),
  - Soit après (compensation)



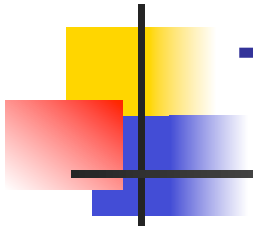
Problèmes…

- Pas de critère de correction !
  - ~sérialisabilité
- La correction d'une synchronisation n'est tout simplement pas définie !!
- Pas de cadre formel !



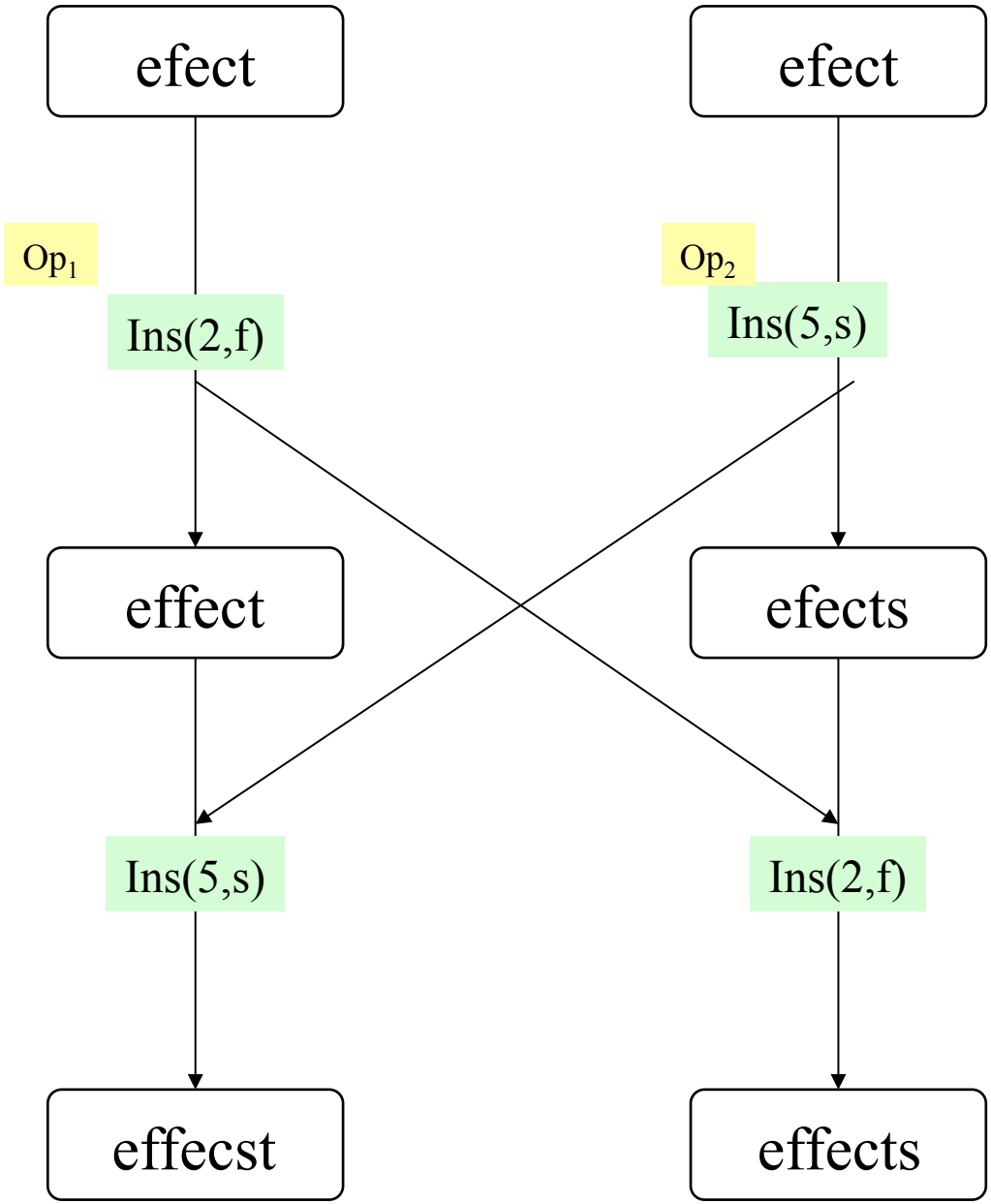
Approche

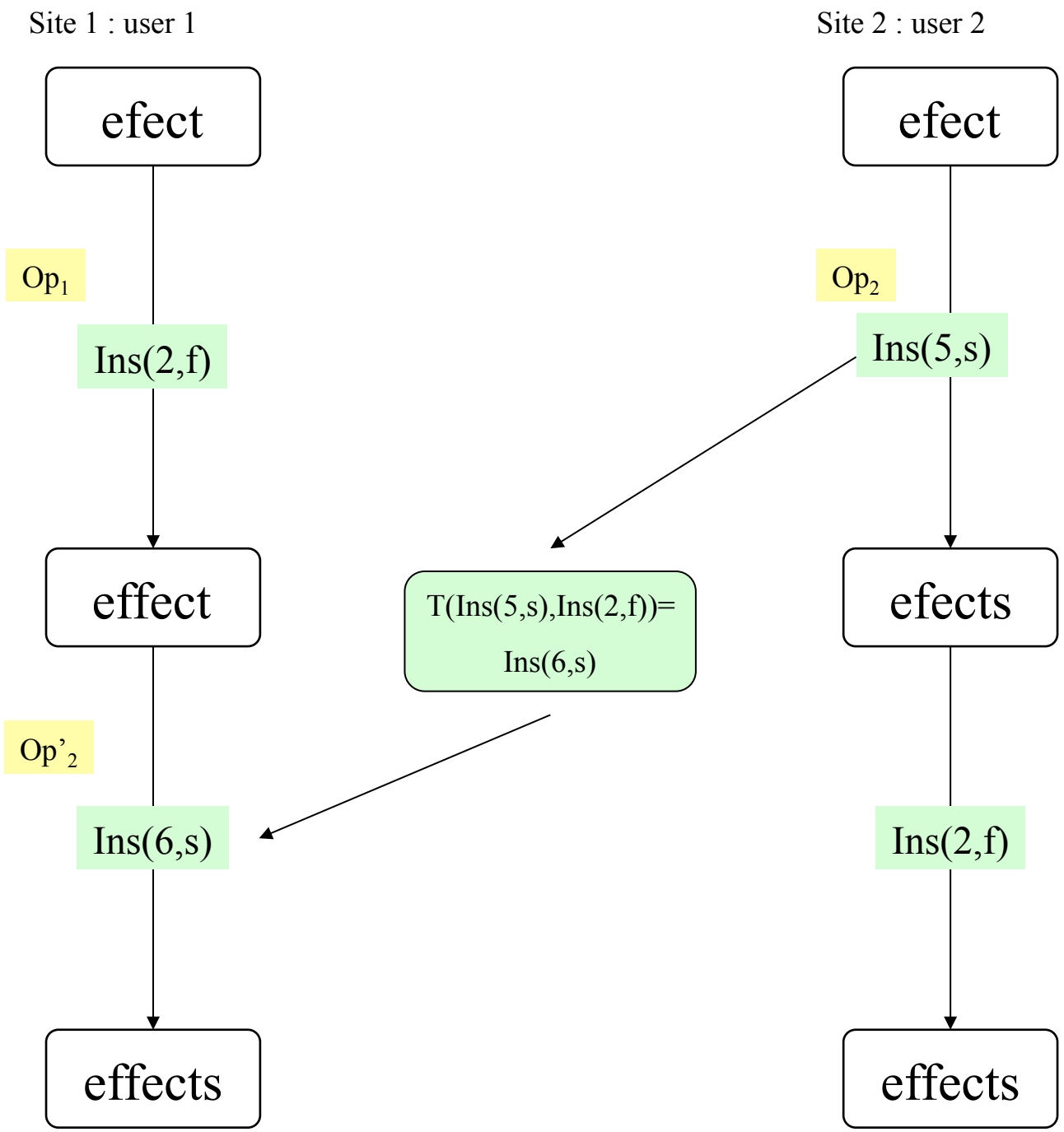
- Transformées opérationnelles
- Un cadre théorique pour raisonner…
- Développé à l'origine pour le real-time groupware (CSCW)
  - Éditeurs collaboratifs synchrone (texte, vectoriel)
  - Premier papier en 89 (Ellis&Gibbs)

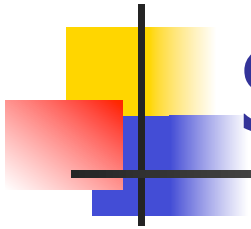


Transformées opérationnelles

- N sites
- Chaque site a 1 copie des objets partagés
- Une opération :
  1) Générée et exécutée sur 1 site
  2) Envoyée aux autres sites
  3) Reçues par les autres sites
  4) Ré-exécutée par les autres sites.


efect
efect
Op1
Op2
Ins(2,f)
Ins(5,s)
effect
efects
Ins(5,s)
Ins(2,f)
effecst
effects

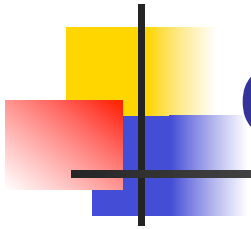

Site 1 : user 1
Site 2 : user 2
efect
efect
Op1
Op2
Ins(2,f)
Ins(5,s)
effect
T(Ins(5,s),Ins(2,f))=
Ins(6,s)
efects
Op'2
Ins(6,s)
Ins(2,f)
effects
effects



Structure de modèle

- Algorithmes (dOpt, GOTO, aDopted, SOCT1,2,3,4,5)
  - Responsable de la propagation des opérations
  - Appelle les fonctions de transformation quand nécessaire
  - Indépendant des types objets partagés
- Fonctions de transformations:
  - Spécifique au type des objets partagés
  - 2 opération concurrentes op1,op2 définies sur un même état '*s*'
  - Retourne op'1 equivalent à op1 sur 's o op2'

Critère de cohérence des copies



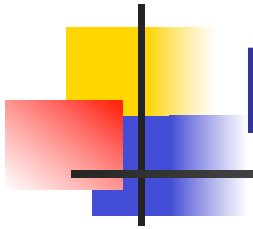
- Causalité :
  - Si op2 est exécutée après op1 sur 1 site, alors op2 est exécutée après op1 sur tous les sites.
- Convergence
  - Au repos, toutes les copies sont identiques
- Intention
  - Quand une opération est transformée, la transformée à un effet équivalent à l'original

Correction des fonctions de transformations

- Les algorithmes garantissent Causalité, Intention, Convergence ssi…
- Les fonctions de transformation garantissent:

$$C1 : op_1.op_2^{op_1} \equiv op_2.op_1^{op_2}$$

$$C2 : op_3^{op_1:op_2} = op_3^{op_2:op_1}$$



Problème

- Preuve des conditions C1 et C2 sur les fonctions de transformation
- 123 cas à vérifier pour ins(p,c), del(p)
- Et à re-vérifier à chaque changement !
- Utilisation d'un prouveur de thèorème spécifique (SPIKE + VOTE), Coopération projet CASSIS

ELLIS (89)

$T_{ii}(\underline{\text{Ins}}(p_1,c_1,pr_1), \underline{\text{Ins}}(p_2,c_2,pr_2)) :-$
if $p_1 < p_2$ **return** $\underline{\text{Ins}}(p_1,c_1)$
else if $p_1 > p_2$ **return** $\underline{\text{Ins}}(p_1 + 1, c_1,pr_1)$
else if $c_1 == c_2$ **return** $\underline{\text{Id}}()$
else if $pr_1 > pr_2$ **return** $\underline{\text{Ins}}(p_1 + 1,c_1,pr_1)$
else return $\underline{\text{Ins}}(p_1,c_1,pr_1)$

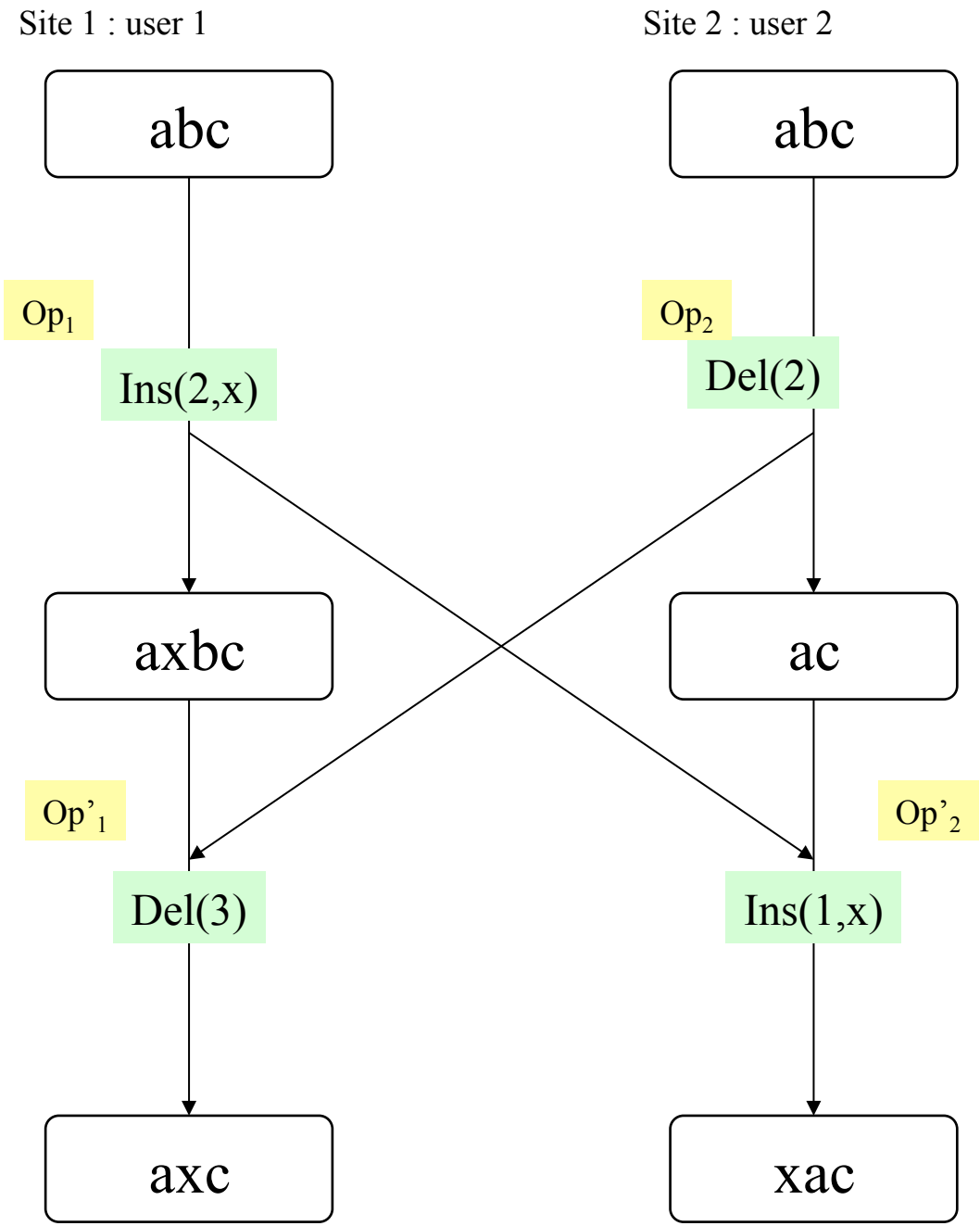
$T_{id}(\underline{\text{Ins}}(p_1,c_1,pr_1), \underline{\text{Del}}(p_2,pr_2)) :-$
if $p_1 < p_2$ **return** $\underline{\text{Ins}}(p_1,c_1,pr_1)$
else return $\underline{\text{Ins}}(p_1 - 1,c_1,pr_1)$

$T_{di}(\underline{\text{Del}}(p_1,pr_1),\underline{\text{Ins}}(p_2,c_2,pr_2)) :-$
if $p_1 < p_2$ **return** $\underline{\text{Del}}(p_1,pr_1)$
else return $\underline{\text{Del}}(p_1 + 1,pr_1)$

$T_{dd}(\underline{\text{Del}}(p_1,pr_1),\underline{\text{Del}}(p_2,pr_2)) :-$
if $p_1 < p_2$ **return** $\underline{\text{Del}}(p_1,pr_1)$
else if $p_1 > p_2$ **return** $\underline{\text{Del}}(p_1 - 1,pr_1)$
else return $\underline{\text{Id}}()$

Pr= priorité basé

sur l'identifieur du site émetteur


Site 1 : user 1
Site 2 : user 2
abc
abc
Op1
Op2
Ins(2,x)
Del(2)
axbc
ac
Op'1
Op'2
Del(3)
Ins(1,x)
axc
xac

Ellis !!

Ressel 96

$T_{ii}(\underline{\text{Ins}}(p_1,c_1,u_1)),(\underline{\text{Ins}}(p_2,c_2,u_2)) :-$
if $p_1<p_2$ or ($p_1=p_2$ and $u_1<u_2$) **return** $\underline{\text{Ins}}(p_1,c_1,u_1)$
else return $\underline{\text{Ins}}(p_1+1,c_1,u_1)$

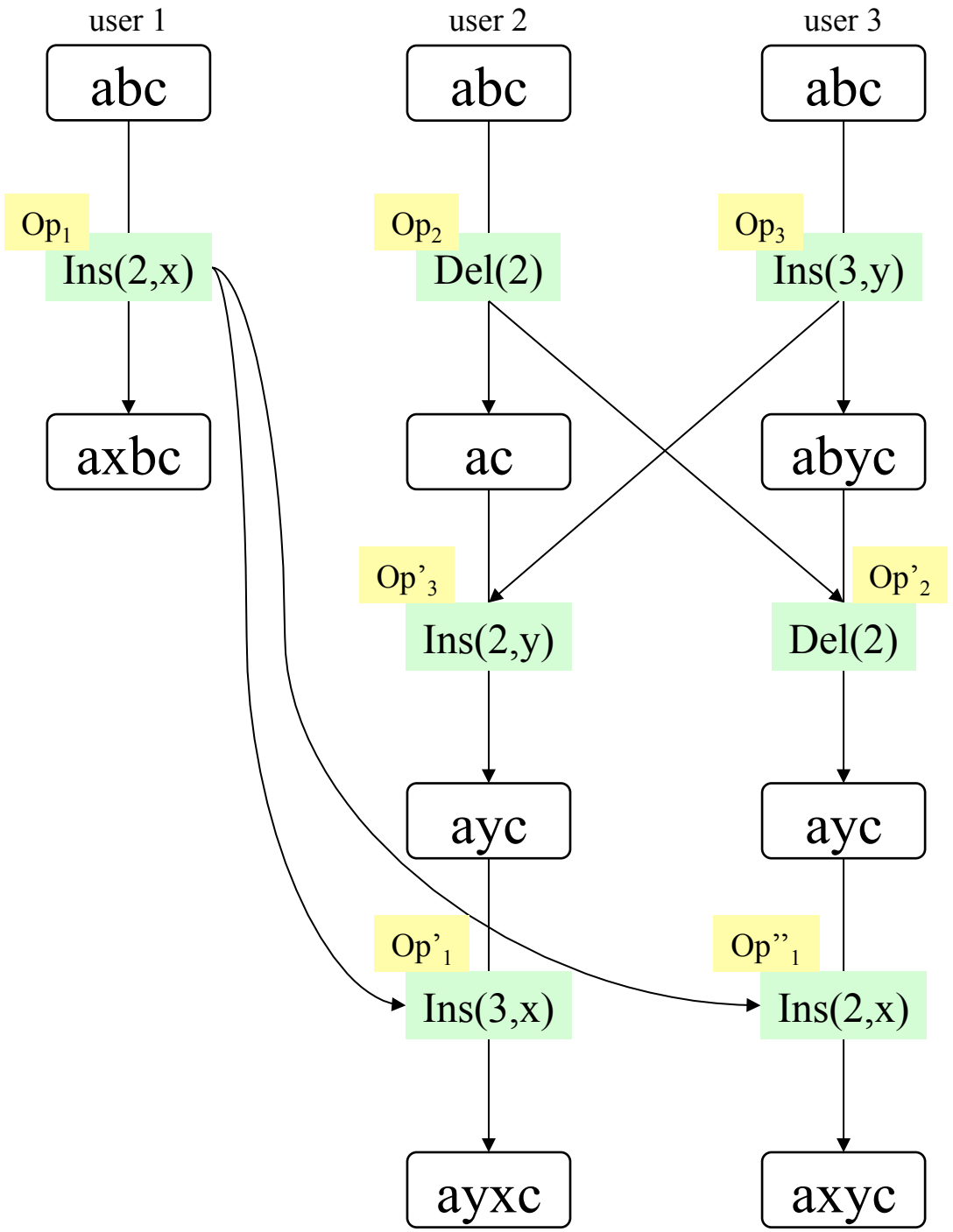
$T_{dd}(\underline{\text{Del}}(p_1,u_1)),\underline{\text{Del}}(p_2,u_2) :-$
if $p_1<p_2$ **return** $\underline{\text{Del}}(p_1,u_1)$
else if $p_1>p_2$ **return** $\underline{\text{Del}}(p_1-1,u_1)$
else return $\underline{\text{Id}}()$

$T_{id}(\underline{\text{Ins}}(p_1,c_1,u_1), \underline{\text{Del}}(p_2,u_2)) :-$
if $p_1 \leq p_2$ **return** $\underline{\text{Ins}}(p_1,c_1,u_1)$
else return $\underline{\text{Ins}}(p_1-1,c_1,u_1)$

$T_{di}(\underline{\text{Del}}(p_1,u_1),\underline{\text{Ins}}(p_2,c_2,u_2)) :-$
if $p_1 < p_2$ **return** $\underline{\text{Del}}(p_1,u_1)$
else return $\underline{\text{Del}}(p_1+1,u_1)$

U1: priorité du site du user 1

Les priorités sont totalement ordonnées


user 1
user 2
user 3
abc
abc
abc
Op1
Ins(2,x)
Op2
Del(2)
Op3
Ins(3,y)
axbc
ac
abyc
Op'3
Ins(2,y)
Op'2
Del(2)
ayc
ayc
Op'1
Ins(3,x)
Op''1
Ins(2,x)
ayxc
axyc

RESSEL !!

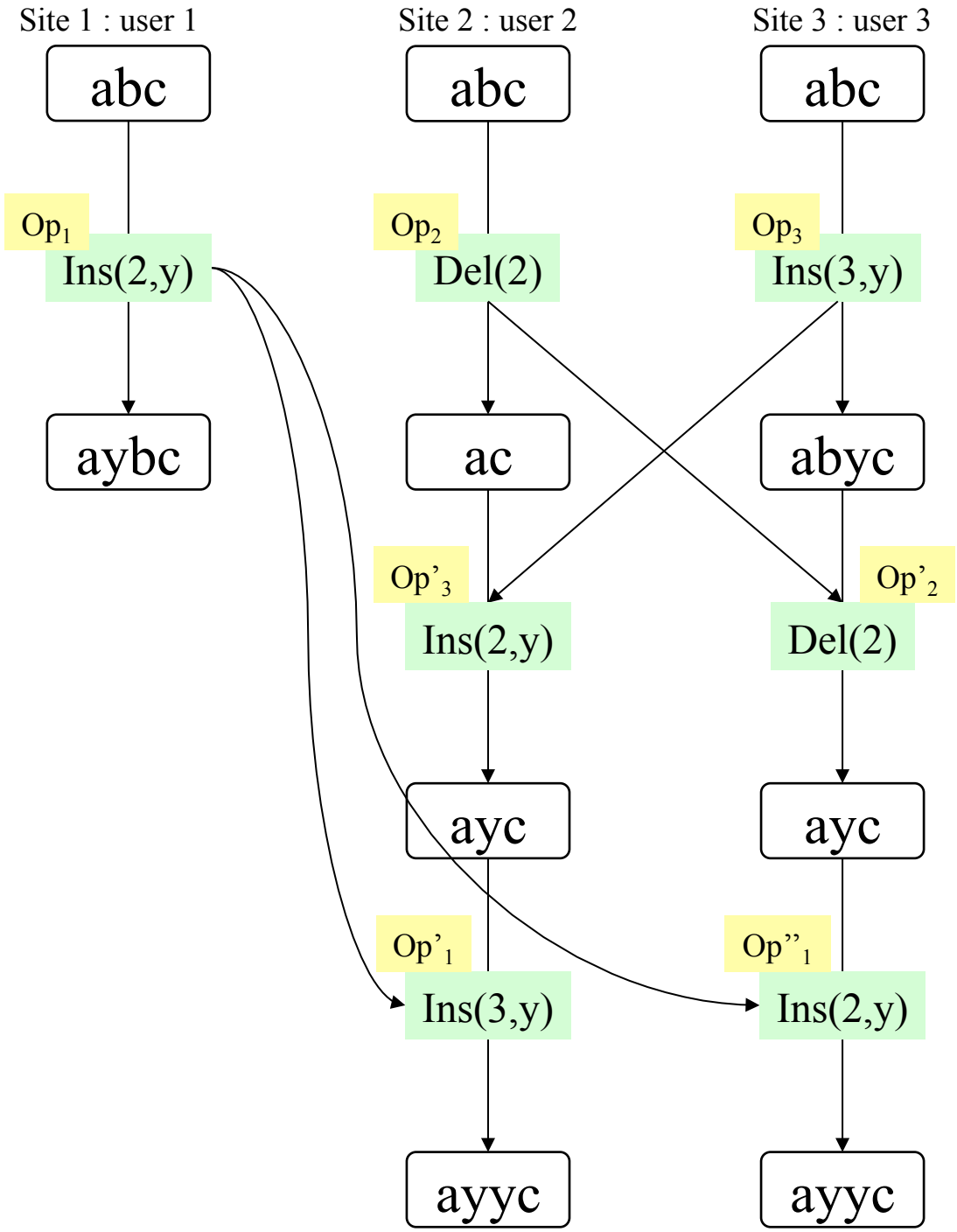
SUN (98)

T(Ins(p_1,c_1),Ins(p_2,c_2)) :−
  if $p_1 < p_2$ **return** Ins(p_1, c_1)
  else return Ins($p_1 + 1$, c_1)

T(Ins(p_1,c_1),Del(p_2)) :−
  if $p_1 \leq p_2$ **return** Ins(p_1,c_1)
  else return Ins($p_1 - 1$,c_1)

T(Del(p_1),Ins(p_2,c_2)) :−
  if $p_1 < p_2$ **return** Del(p_1)
  else return Del($p_1 + 1$)

T(Del(p_1),Del(p_2)) :−
  if $p_1 < p_2$ **return** Del(p_1)
  else if $p_1 > p_2$ **return** Del($p_1 - 1$)
  else return Id()


Site 1 : user 1
Site 2 : user 2
Site 3 : user 3
abc
abc
abc
Op1
Ins(2,y)
Op2
Del(2)
Op3
Ins(3,y)
aybc
ac
abyc
Op'3
Ins(2,y)
Op'2
Del(2)
ayc
ayc
Op'1
Ins(3,y)
Op''1
Ins(2,y)
ayyc
ayyc

SUN !!

Suleiman97 (Ok mais surpécifié)

T(Ins(p_1,c_1,b_1,a_1),Ins(p_2,c_2,b_2,a_2)) :–
  if $p_1 < p_2$ **return** Ins(p_1,c_1,b_1,a_1)
  else if $p_1 > p_2$ **return** Ins($p_1 + 1$,c_1,b_1,a_1)
  else // $p_1 == p_2$
    if $(b_1 \cap a_2) \neq \emptyset$ **return** Ins($p_1 + 1$,c_1,b_1,a_1)
    else if $(a_1 \cap b_2) \neq \emptyset$ **return** Ins(p_1,c_1,b_1,a_1)
    else if code(c_1)>code(c_2) **return** Ins(p_1,c_1,b_1,a_1)
    else if code(c_1)<code(c_2) **return** Ins($p_1 + 1$,c_1,b_1,a_1)
    else return Id()

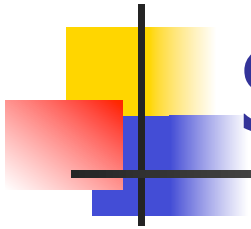
T(Ins(p_1,c_1,b_1,a_1),Del(p_2)) :–
  if $p_1 > p_2$ **return** Ins($p_1 - 1$,c_1,b_1+Del(p_2),a_1)
  else return Ins(p_1,c_1,b_1,a_1+Del(p_2))

Nous (Ecsw03)

$T(\underline{Ins}(p_1,ip_1,c_1),\underline{Ins}(p_2,ip_2,c_2)) :-$
  if $p_1<p_2$ **return** $\underline{Ins}(p_1,ip_1,c_1)$
  else if $p_1>p_2$ $\underline{Ins}(p_1+1,ip_1,c_1)$
  else // $p_1==p_2$
   if $ip_1<ip_2$ **return** $\underline{Ins}(p_1,ip_1,c_1)$
   else if $ip_1>ip_2$ **return** $\underline{Ins}(p_1+1,ip_1,c_1)$
   else // $ip_1==ip_2$
    if $\text{code}(c_1) < \text{code}(c_2)$ **return** $\underline{Ins}(p_1,ip_1,c_1)$
    else if $\text{code}(c_1) > \text{code}(c_2)$ **return** $\underline{Ins}(p_1+1,ip_1,c_1)$
    else // $c_1==c_2$
      return $\underline{Id}()$

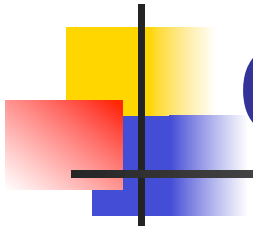
$T(\underline{Ins}(p_1,ip_1,c_1),\underline{Del}(p_2)) :-$
  if $p_1 > p_2$ **return** $\underline{Ins}(p_1 - 1,ip_1,c_1)$
  else **return** $\underline{Ins}(p_1,ip_1,c_1)$

Ip= point d'insertion



Synchronisation…

- Appliquer le modèle des transformées au pb de synchronisation.
- Gérer **tous** les conflits avec le principe de compensation (~à la diff3).
- Exemple:
  - Synchroniser un système de fichiers
  - + le contenu des fichiers (pour les types gérés).



Objectif du synchroniseur

- Réconcilier en respectant
  - Causalité, convergence, intention
  - Compensation
- Granularité : opération
- Pour les types d'objets définis

Synchro algorithme (basé SOCT4)

```
Sync(log, N_s) :-
  while ((op_r = getOp(N_s+1))!=∅)
    for (i=0; i<log.size(); i++)
      op_l=log[i];
      log[i]=T(op_l, op_i)
      op'_i=T(op_i, op_l);
    endfor
    execute(op'i)
    N_s=N_s+1
  endwhile

  for (i=0; i<log.size(); i++)
    op'_l=log[i];
    if send(op'_l, N_s+1) then
      N_s=N_s+1
    else
      error 'need to synchronize'
    endif
  endfor
```

Synchro exemple

$site_1$	$site_2$
op_1	op_3
op_2	op_4
$s_1 = synchronize$	
	$s_2 = synchronize$
$s_3 = synchronize$	

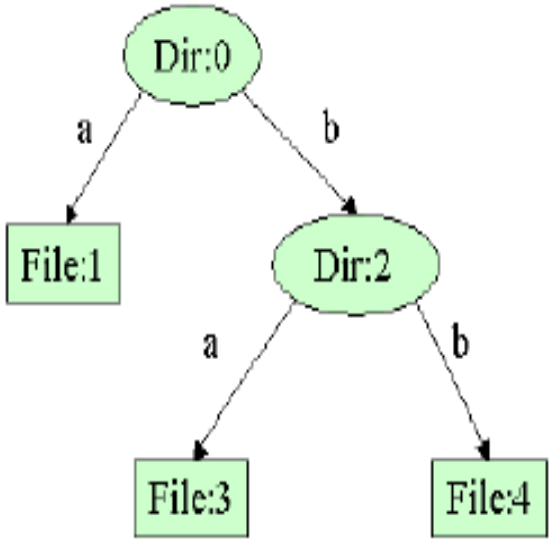
Après S3, sur site1

op_1
op_2
$op_3'' = T(T(op_3, op_1), op_2)$
$op_4'' = T(T(op_4, op_1'), op_2')$

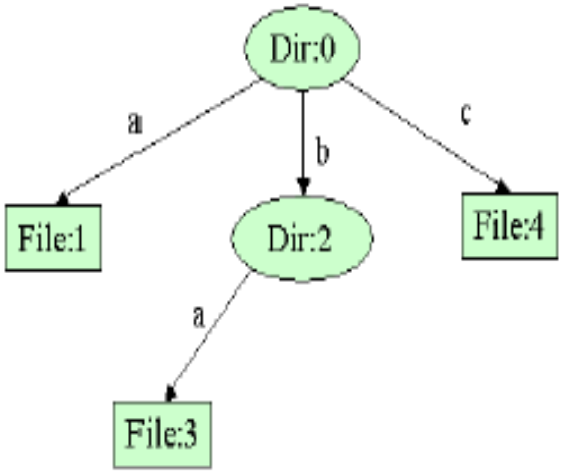
Après S3, sur site2

op_3
op_4
$op_1'' = T(T(op_1, op_3), op_4)$
$op_2'' = T(T(op_2, op_3'), op_4')$

Système de fichier



(a) Initial tree



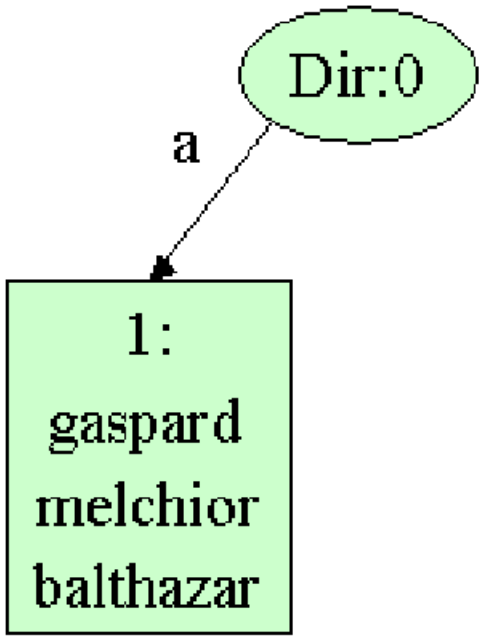
(b) tree after $mv(2, 4, b, 0, c)$

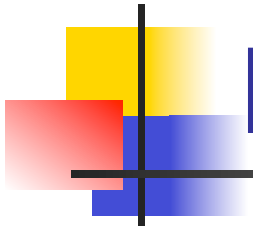
T(mkfile/mkfile)

$$
\begin{array}{l}
T(\underline{\mathrm{mf}}(id_1, idp_1, n_1, s_1), \underline{\mathrm{mf}}(id_2, idp_2, n_2, s_2)) = \\
\textbf{if } (idp_1 == idp_2) \textbf{ then} \\
\quad \textbf{if } (n_1 == n_2) \textbf{ then} \\
\quad\quad \textbf{if } (id_1 < id_2) \textbf{ then return} \\
\quad\quad \underline{\mathrm{mf}}(id_1, idp_1, max(s_1) \odot id_1, \\
\quad\quad\quad\quad s_1 \cup \{max(s_1) \odot id_1\}) \\
\quad\quad \textbf{else return} \\
\quad\quad\quad \underline{\mathrm{mv}}(idp_2, id_2, n_2, idp_2, max(s_2) \odot id_2, \\
\quad\quad\quad\quad s_2 \setminus \{n_2\} \cup \{max(s_2) \odot id_2\}) \\
\quad\quad\quad \boxplus\ \underline{\mathrm{mf}}(id_1, idp_1, n_1, s_2 \cup \{max(s_2) \odot id_2\}) \\
\quad\quad \textbf{endif} \\
\quad \textbf{else} \\
\quad \textbf{return } \underline{\mathrm{mf}}(id_1, idp_1, n_1, s_2 \cup \{n_1\}) \\
\quad \textbf{endif} \\
\textbf{else} \\
\quad \textbf{return } \underline{\mathrm{mf}}(id_1, idp_1, n_1, s_1) \\
\textbf{endif};
\end{array}
$$

s=ensemble des noms des nœuds frères + soi-même

Exemple : état initial





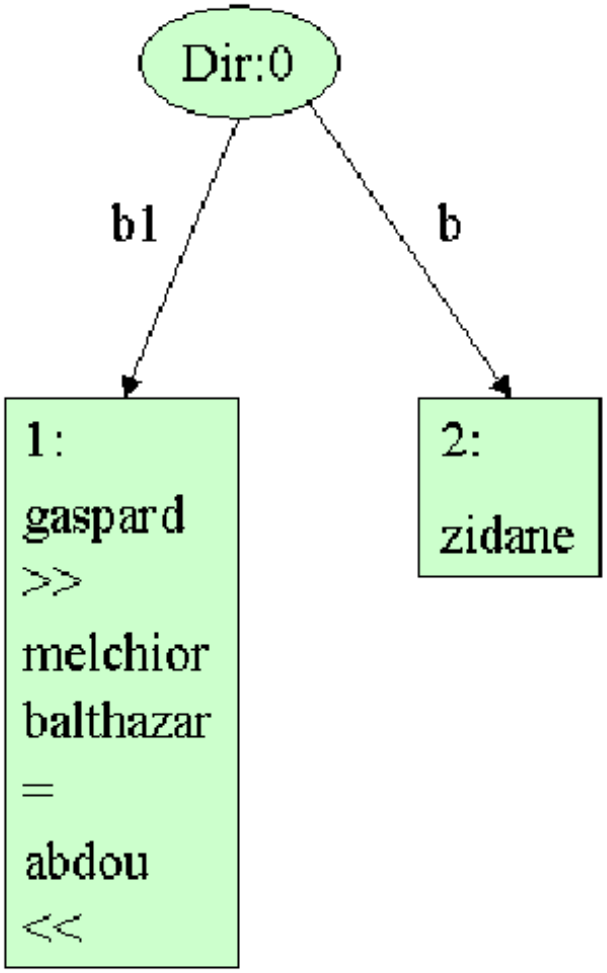
Exemple : scénario

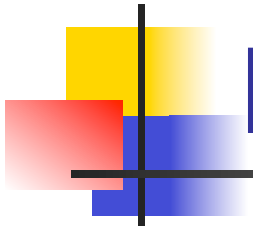
u_1	u_2	u_3
$op_1 = mv(0, 1, a, 0, b, \{b\})$	$op_3 = mf(2, 0, b, \{a, b\})$	$op_5 = ab(1, 3, 3, \{"abdou"\})$
$op_2 = db(1, 2, \{"melchior", "balthazar"\})$	$op_4 = ab(2, 0, 0, \{"zidane"\})$	
$s_1 = synchronize$		
	$s_2 = synchronize$	
		$s_3 = synchronize$
	$s_4 = synchronize$	
$s_5 = synchronize$		

Exemple : opérations exécutées

mf(1,0,a,{a}),
ab(1,0,0,{"gaspard","melchior","balthazar"}),
op_1=mv(0,1,a,0,b,{b}),
op_2=db(1,2,{"melchior","balthazar"}),
op_3^2=mv(0,1,b,0,b1,{b1}) ⊞ mf(2,0,b,{b,b1}),
op_4^2=ab(2,0,0,{"zidane"}),
op_5^4=ab(1,2,2,{">>","melchior",
"balthazar","=","abdou","<<"});

Exemple: état final



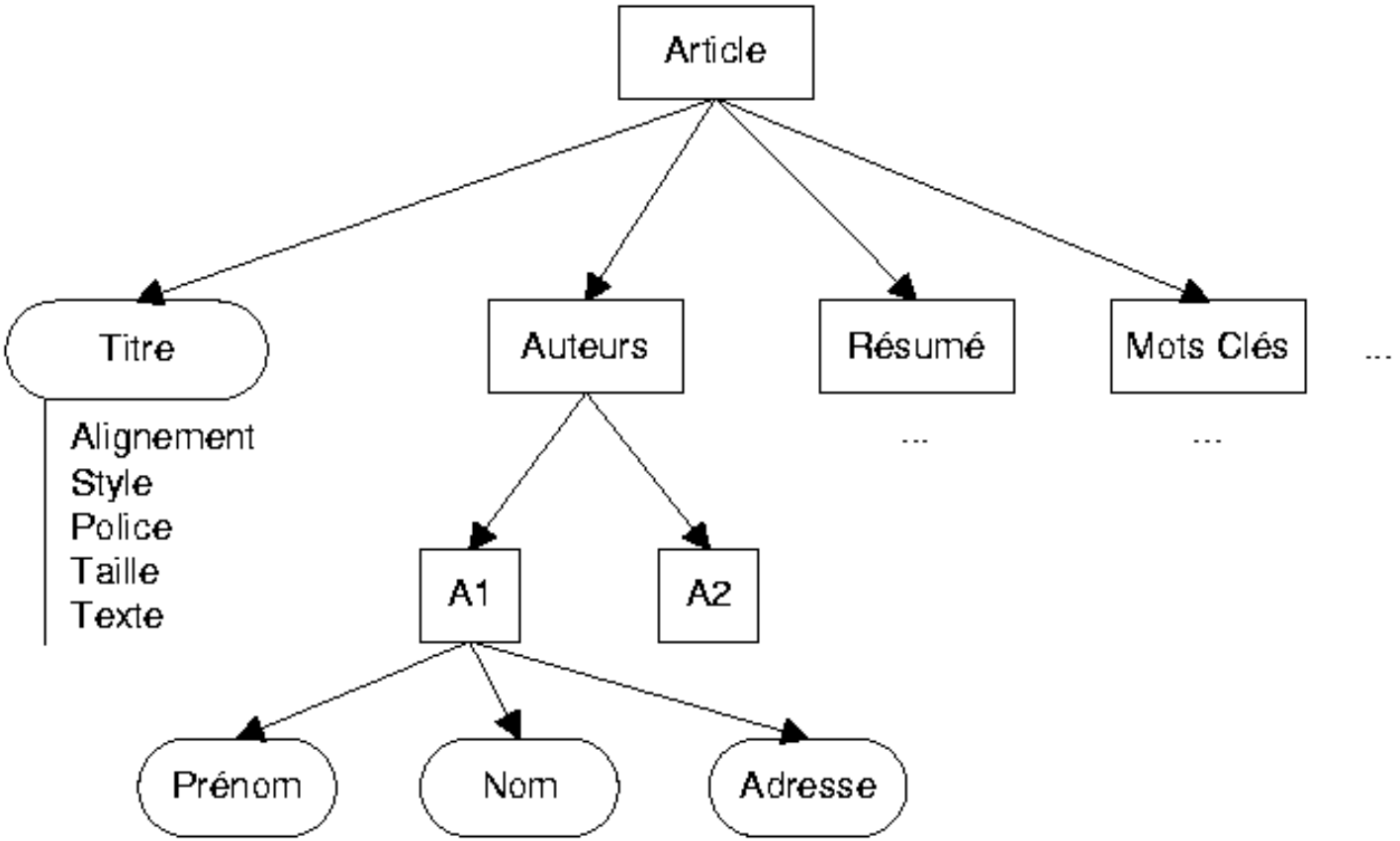


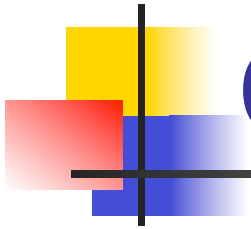
Editeurs SAMS

- Synchrone
- Asynchrone
- Multi-synchrone (synchroniseur)

- Synchrone =synchronisation temps réél

Un exemple en XML





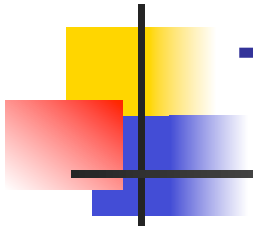
Opérations XML

Les opérations de manipulation de la structure

```
createObject(obj OID, parent OID, type ObjectClass, p POSITION)
deleteObject(obj OID)
undeleteObject(obj OID)
moveObject(obj OID, toParent OID, p POSITION)
copyObject(destObj OID, srcObj OID, toParent OID, p POSITION)
```

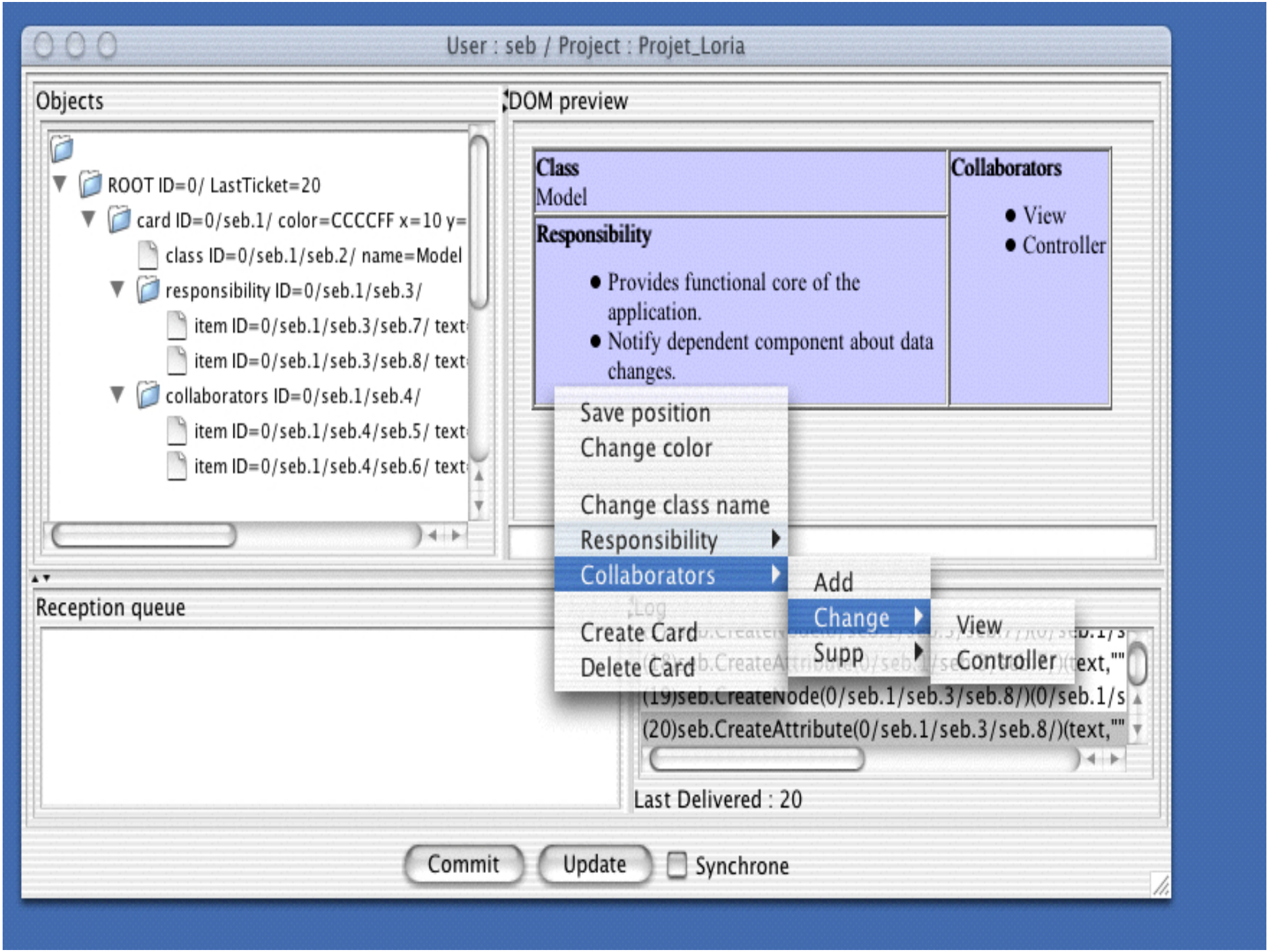
Les opérations de manipulation du contenu

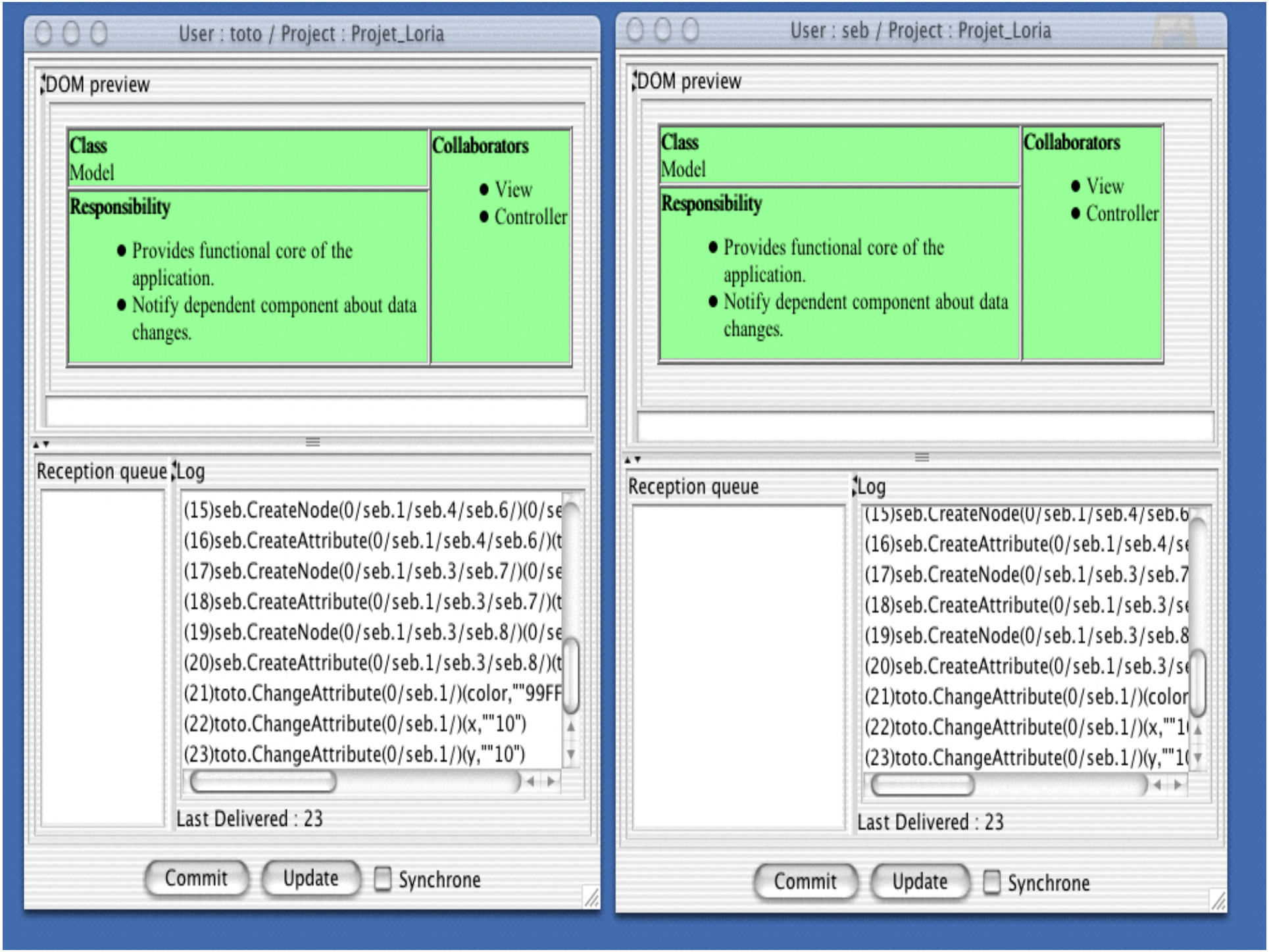
```
changeAttribute(obj OID, name AttribName, val AttribValue)
```

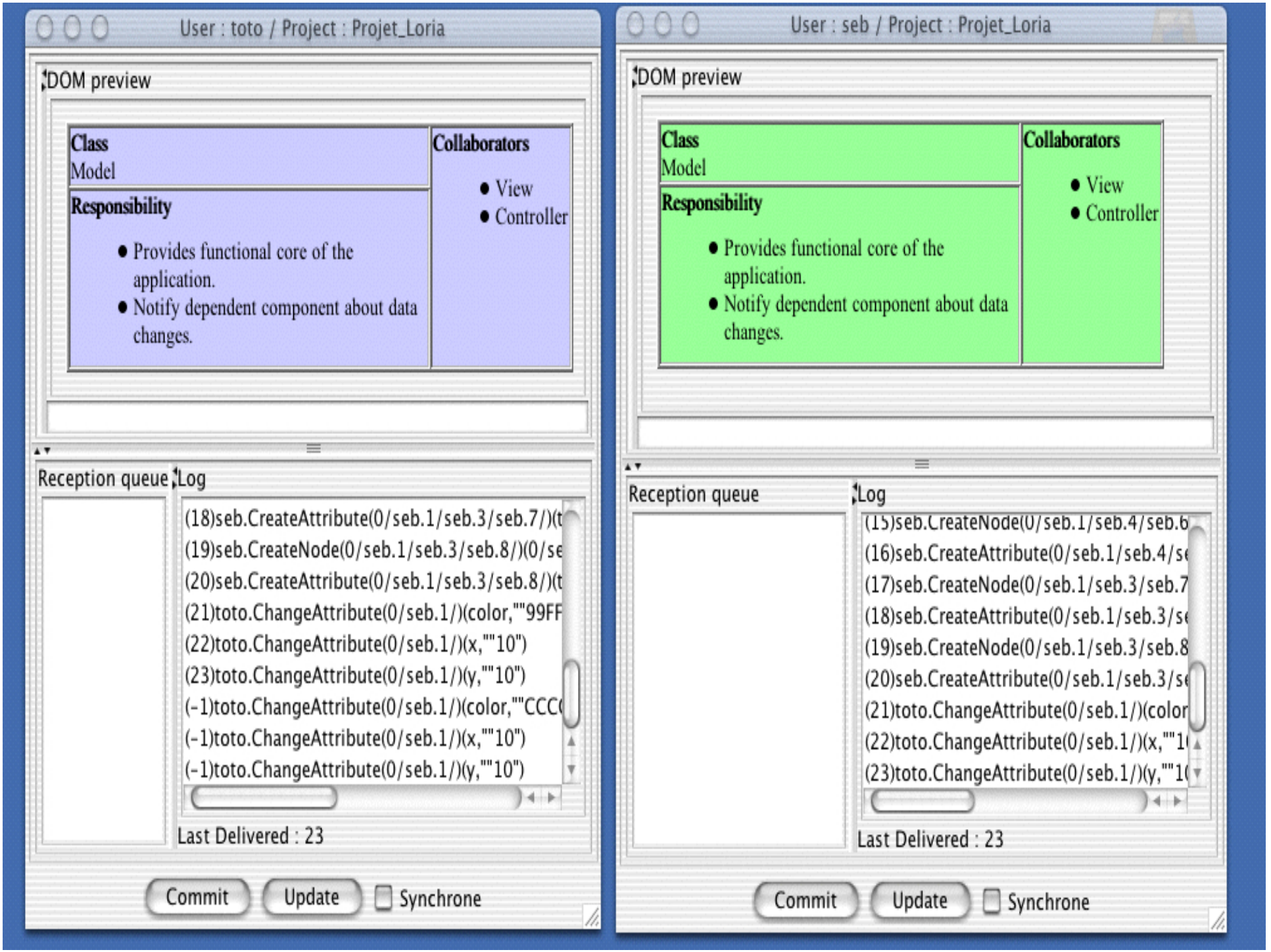


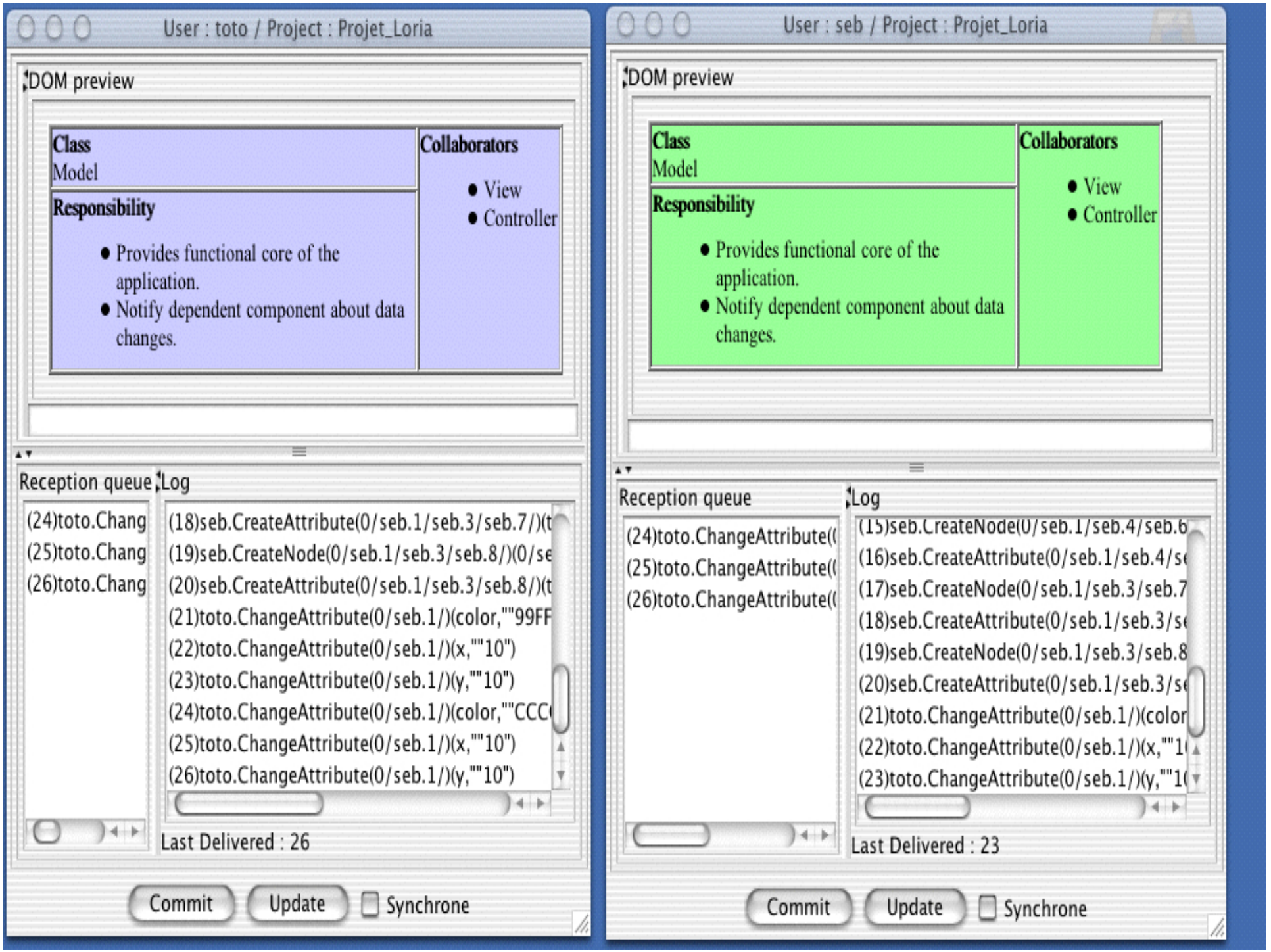
Transformées XML

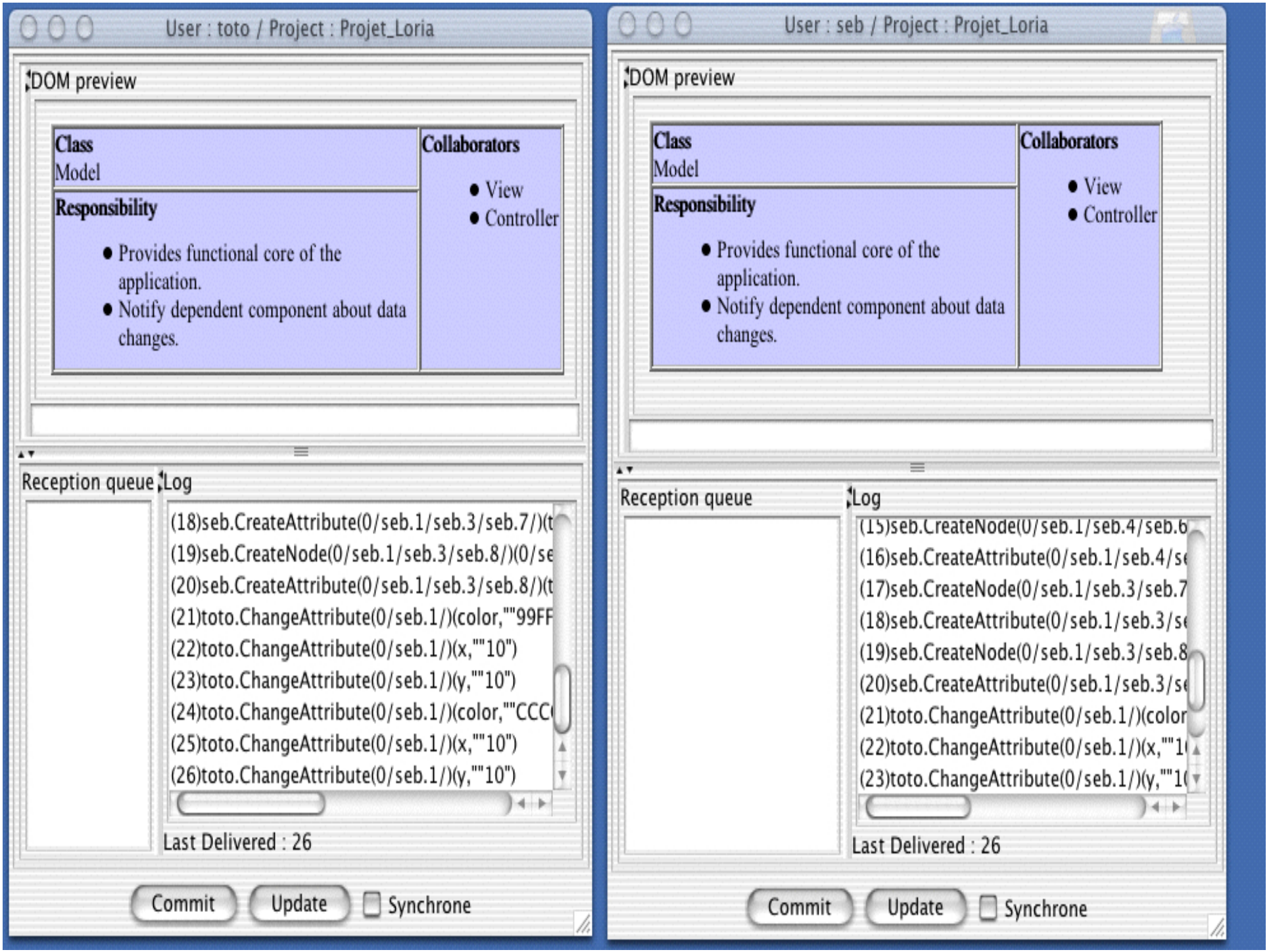
```
T(ChangeAttribute(n1,a1,v1),ChangeAttribute(n2,a2,v2)):-
    if n1=n2 and a1=a2 and  v1=v2
        return noop /* nothing to do */
    if n1=n2 and  a1=a2 and v1<>v2
        return ChangeAttribute(n1,a1,max(v1,v2))
    else
        return ChangeAttribute(n1,a1,v1)

T(CreateNode(n1,t1),DeleteNode(n2)):-
  if( n1 ChildOf n2)
      return noop       /* nothing to do */
  else
      return CreateNode(n1,t1)
```

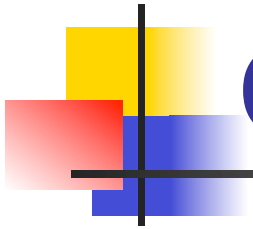

User : seb / Project : Projet_Loria
Objects
DOM preview
ROOT ID=0/ LastTicket=20
card ID=0/seb.1/ color=CCCCFF x=10 y=
class ID=0/seb.1/seb.2/ name=Model
responsibility ID=0/seb.1/seb.3/
item ID=0/seb.1/seb.3/seb.7/ text
item ID=0/seb.1/seb.3/seb.8/ text
collaborators ID=0/seb.1/seb.4/
item ID=0/seb.1/seb.4/seb.5/ text
item ID=0/seb.1/seb.4/seb.6/ text
Class
Model
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Collaborators
View
Controller
Save position
Change color
Change class name
Responsibility
Collaborators
Create Card
Delete Card
Add
Change
Supp
View
Controller
Reception queue
(19)seb.CreateNode(0/seb.1/seb.3/seb.8/)(0/seb.1/s
(20)seb.CreateAttribute(0/seb.1/seb.3/seb.8/)(text,""
Last Delivered : 20
Commit
Update
Synchrone


User : toto / Project : Projet_Loria
DOM preview
Class
Model
Collaborators
View
Controller
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Reception queue
Log
(15)seb.CreateNode(0/seb.1/seb.4/seb.6/)(0/se
(16)seb.CreateAttribute(0/seb.1/seb.4/seb.6/)(t
(17)seb.CreateNode(0/seb.1/seb.3/seb.7/)(0/se
(18)seb.CreateAttribute(0/seb.1/seb.3/seb.7/)(t
(19)seb.CreateNode(0/seb.1/seb.3/seb.8/)(0/se
(20)seb.CreateAttribute(0/seb.1/seb.3/seb.8/)(t
(21)toto.ChangeAttribute(0/seb.1/)(color,""99FF
(22)toto.ChangeAttribute(0/seb.1/)(x,""10")
(23)toto.ChangeAttribute(0/seb.1/)(y,""10")
Last Delivered : 23
Commit
Update
Synchrone
User : seb / Project : Projet_Loria
DOM preview
Class
Model
Collaborators
View
Controller
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Reception queue
Log
Last Delivered : 23
Commit
Update
Synchrone


User : toto / Project : Projet_Loria
DOM preview
Class
Model
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Collaborators
View
Controller
Reception queue
Log
(18)seb.CreateAttribute(0/seb.1/seb.3/seb.7/)(t
(19)seb.CreateNode(0/seb.1/seb.3/seb.8/)(0/se
(20)seb.CreateAttribute(0/seb.1/seb.3/seb.8/)(t
(21)toto.ChangeAttribute(0/seb.1/)(color,""99FF
(22)toto.ChangeAttribute(0/seb.1/)(x,""10")
(23)toto.ChangeAttribute(0/seb.1/)(y,""10")
(-1)toto.ChangeAttribute(0/seb.1/)(color,""CCC
(-1)toto.ChangeAttribute(0/seb.1/)(x,""10")
(-1)toto.ChangeAttribute(0/seb.1/)(y,""10")
Last Delivered : 23
Commit
Update
Synchrone
User : seb / Project : Projet_Loria
DOM preview
Class
Model
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Collaborators
View
Controller
Reception queue
Log
(15)seb.CreateNode(0/seb.1/seb.4/seb.6
(16)seb.CreateAttribute(0/seb.1/seb.4/se
(17)seb.CreateNode(0/seb.1/seb.3/seb.7
(18)seb.CreateAttribute(0/seb.1/seb.3/se
(19)seb.CreateNode(0/seb.1/seb.3/seb.8
(20)seb.CreateAttribute(0/seb.1/seb.3/se
(21)toto.ChangeAttribute(0/seb.1/)(color
(22)toto.ChangeAttribute(0/seb.1/)(x,""1
(23)toto.ChangeAttribute(0/seb.1/)(y,""1
Last Delivered : 23
Commit
Update
Synchrone


User : toto / Project : Projet_Loria
DOM preview
Class
Model
Collaborators
View
Controller
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Reception queue
Log
(24)toto.Chang
(25)toto.Chang
(26)toto.Chang
(18)seb.CreateAttribute(0/seb.1/seb.3/seb.7/)(t
(19)seb.CreateNode(0/seb.1/seb.3/seb.8/)(0/se
(20)seb.CreateAttribute(0/seb.1/seb.3/seb.8/)(t
(21)toto.ChangeAttribute(0/seb.1/)(color,""99FF
(22)toto.ChangeAttribute(0/seb.1/)(x,""10")
(23)toto.ChangeAttribute(0/seb.1/)(y,""10")
(24)toto.ChangeAttribute(0/seb.1/)(color,""CCC
(25)toto.ChangeAttribute(0/seb.1/)(x,""10")
(26)toto.ChangeAttribute(0/seb.1/)(y,""10")
Last Delivered : 26
Commit
Update
Synchrone
User : seb / Project : Projet_Loria
DOM preview
Class
Model
Collaborators
View
Controller
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Reception queue
Log
(24)toto.ChangeAttribute(
(25)toto.ChangeAttribute(
(26)toto.ChangeAttribute(
(15)seb.CreateNode(0/seb.1/seb.4/seb.6
(16)seb.CreateAttribute(0/seb.1/seb.4/se
(17)seb.CreateNode(0/seb.1/seb.3/seb.7
(18)seb.CreateAttribute(0/seb.1/seb.3/se
(19)seb.CreateNode(0/seb.1/seb.3/seb.8
(20)seb.CreateAttribute(0/seb.1/seb.3/se
(21)toto.ChangeAttribute(0/seb.1/)(color
(22)toto.ChangeAttribute(0/seb.1/)(x,""1
(23)toto.ChangeAttribute(0/seb.1/)(y,""1
Last Delivered : 23
Commit
Update
Synchrone

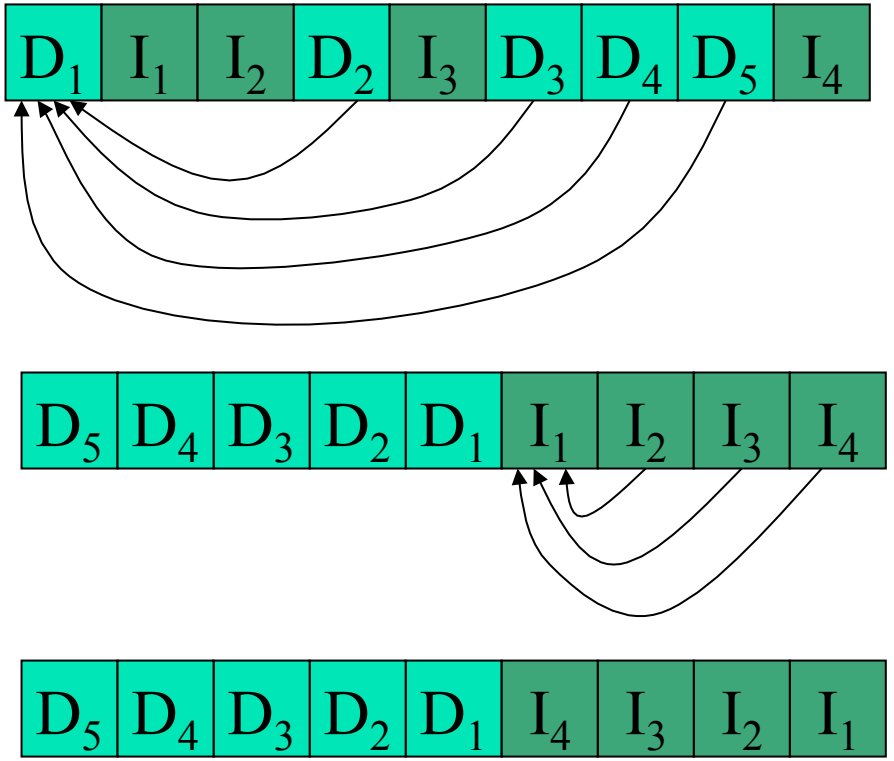

User : toto / Project : Projet_Loria
DOM preview
Class
Model
Collaborators
View
Controller
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Reception queue
Log
(18)seb.CreateAttribute(0/seb.1/seb.3/seb.7/)(t
(19)seb.CreateNode(0/seb.1/seb.3/seb.8/)(0/se
(20)seb.CreateAttribute(0/seb.1/seb.3/seb.8/)(t
(21)toto.ChangeAttribute(0/seb.1/)(color,""99FF
(22)toto.ChangeAttribute(0/seb.1/)(x,""10")
(23)toto.ChangeAttribute(0/seb.1/)(y,""10")
(24)toto.ChangeAttribute(0/seb.1/)(color,""CCC
(25)toto.ChangeAttribute(0/seb.1/)(x,""10")
(26)toto.ChangeAttribute(0/seb.1/)(y,""10")
Last Delivered : 26
Commit
Update
Synchrone
User : seb / Project : Projet_Loria
DOM preview
Class
Model
Collaborators
View
Controller
Responsibility
Provides functional core of the application.
Notify dependent component about data changes.
Reception queue
Log
(15)seb.CreateNode(0/seb.1/seb.4/seb.6
(16)seb.CreateAttribute(0/seb.1/seb.4/se
(17)seb.CreateNode(0/seb.1/seb.3/seb.7
(18)seb.CreateAttribute(0/seb.1/seb.3/se
(19)seb.CreateNode(0/seb.1/seb.3/seb.8
(20)seb.CreateAttribute(0/seb.1/seb.3/se
(21)toto.ChangeAttribute(0/seb.1/)(color
(22)toto.ChangeAttribute(0/seb.1/)(x,""1
(23)toto.ChangeAttribute(0/seb.1/)(y,""1
Last Delivered : 26
Commit
Update
Synchrone



Compression d'histoire

- PB: OT -> explosion des logs
- Utilisation de transformées en arrière pour compresser les logs…

Algorithme


D1 I1 I2 D2 I3 D3 D4 D5 I4
D5 D4 D3 D2 D1 I1 I2 I3 I4
D5 D4 D3 D2 D1 I4 I3 I2 I1

Transposée de `insert` – `insert`...

- Algorithme :

si $p_2 < p_1$ **alors**
    insert(p_2, c_2) • *insert*($p_1 + c_2.taille$, c_1)
sinon si $p_2 > p_1 + c_1.taille$ **alors**
    insert($p_2 - c_1.taille$, c_2) • *insert*(p_1, c_1)
sinon // $p_1 \leq p_2 \leq p_1 + c_1.taille$
    insert(p_1, (c_1.*insert*($p_2 - p_1$, c_2)) // Factorisation
finsi

Transposée de `delete` – `delete`...

- Algorithme :

si p2 + lg2 < p1 **alors**
    delete(p2, lg2) • *delete*(p1 – lg2, lg1)
sinon si p2 > p1 **alors**
    delete(p2 + lg1, lg2) • *delete* (p1, lg1)
sinon // p2 ≤ p1 ≤ p2 + lg2
    delete(p2, lg1 + lg2) // Factorisation
finsi

Transposée de `delete` – `insert`...

- Algorithme :

si p2 < p1 **alors**

insert(p_2, c_2) • *delete*(p_1 + c_2.*taille*, lg_1)

sinon

insert(p_2 + lg_1, c_2) • *delete*(p_1, lg_1)

finsi

Transposée de `insert` – `delete`...

Algorithme :

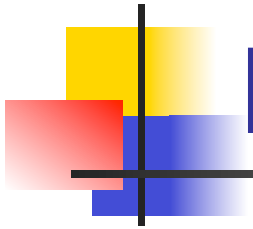
si $p_2 + lg_2 < p_1$ **alors**
    delete(p_2, lg_2) • *insert*($p_1 - lg_2$, c_1)
sinon si $p_2 > p_1 + c_1.taille$ **alors**
    delete($p_2 - c_1.taille$, lg_2) • *insert*(p_1, c_1)
sinon si $p_2 < p_1$ **alors**
    si $p_2 + lg_2 < p_1 + c_1.taille$ **alors**
        delete(p_2, $p_1 - p_2$) • *insert*(p_2, $c_1.droite(c_1.taille - p_2 - lg_2 + p_1)$)
    sinon
        delete(p_2, $lg_2 - c_1.taille$) // Factorisation
    finsi
sinon
    si $p_2 + lg_2 < p_1 + c_1.taille$ **alors** // Factorisation
        insert(p_1, $c_1.gauche(p_2 - p_1) + c_1.droite(c_1.taille - p_2 - lg_2 + p_1)$)
    sinon
        delete(p_1, $p_2 + lg_2 - p_1 - c_1.taille$)
        • *insert*(p_1, $c_1.gauche(p_2 - p_1)$)
    finsi
finsi

log

(16)seb.CreateAttribute(0/seb.5/)(color,""FFFFFF")
(17)seb.CreateNode(0/seb.9/)(0/seb.9/,"""")
(18)seb.CreateNode(0/seb.9/seb.10/)(0/seb.9/seb.10/,"""")
(19)seb.CreateAttribute(0/seb.9/seb.10/)(name,"""")
(20)seb.CreateNode(0/seb.9/seb.11/)(0/seb.9/seb.11/,"""")
(21)seb.CreateNode(0/seb.9/seb.12/)(0/seb.9/seb.12/,"""")
(22)seb.CreateAttribute(0/seb.9/)(x,""10")
(23)seb.CreateAttribute(0/seb.9/)(y,""10")
(24)seb.CreateAttribute(0/seb.9/)(color,""FFFFFF")
(25)seb.CreateNode(0/seb.13/)(0/seb.13/,"""")
(26)seb.CreateNode(0/seb.13/seb.14/)(0/seb.13/seb.14/,"""")
(27)seb.CreateAttribute(0/seb.13/seb.14/)(name,"""")
(28)seb.CreateNode(0/seb.13/seb.15/)(0/seb.13/seb.15/,"""")
(29)seb.CreateNode(0/seb.13/seb.16/)(0/seb.13/seb.16/,"""")
(30)seb.CreateAttribute(0/seb.13/)(x,""10")
(31)seb.CreateAttribute(0/seb.13/)(y,""10")
(32)seb.CreateAttribute(0/seb.13/)(color,""FFFFFF")
(33)seb.ChangeAttribute(0/seb.9/)(x,""438")
(34)seb.ChangeAttribute(0/seb.9/)(y,""132")
(35)seb.ChangeAttribute(0/seb.1/)(x,""13")
(36)seb.ChangeAttribute(0/seb.1/)(y,""131")
(37)seb.ChangeAttribute(0/seb.5/)(x,""430")
(38)seb.ChangeAttribute(0/seb.5/)(y,""7")
(39)seb.ChangeAttribute(0/seb.13/)(x,""10")
(40)seb.ChangeAttribute(0/seb.13/)(y,""10")
(41)seb.ChangeAttribute(0/seb.13/)(color,""CCFF33")
(42)seb.ChangeAttribute(0/seb.5/)(color,""66FF66")
(43)seb.ChangeAttribute(0/seb.9/)(color,""0033CC")
(44)seb.ChangeAttribute(0/seb.1/)(color,""FF0000")
(45)seb.ChangeAttribute(0/seb.9/)(x,""2")
(46)seb.ChangeAttribute(0/seb.9/)(y,""139")
(47)seb.ChangeAttribute(0/seb.5/)(x,""13")
(48)seb.ChangeAttribute(0/seb.5/)(y,""11")
(49)seb.ChangeAttribute(0/seb.13/)(x,""392")
(50)seb.ChangeAttribute(0/seb.13/)(y,""7")
(51)seb.ChangeAttribute(0/seb.1/)(x,""38")
(52)seb.ChangeAttribute(0/seb.1/)(y,""18")
(53)seb.DeleteNode(0/seb.1/)
(54)seb.ChangeAttribute(0/seb.5/)(x,""13")
(55)seb.ChangeAttribute(0/seb.5/)(y,""11")
(56)seb.DeleteNode(0/seb.5/)
(57)seb.ChangeAttribute(0/seb.9/)(x,""49")
(58)seb.ChangeAttribute(0/seb.9/)(y,""11")
(59)seb.DeleteNode(0/seb.9/)
(60)seb.ChangeAttribute(0/seb.13/)(x,""283")
(61)seb.ChangeAttribute(0/seb.13/)(y,""69")
(62)seb.ChangeAttribute(0/seb.13/)(color,""3333FF")
(63)seb.ChangeAttribute(0/seb.13/)(x,""10")
(64)seb.ChangeAttribute(0/seb.13/)(y,""16")

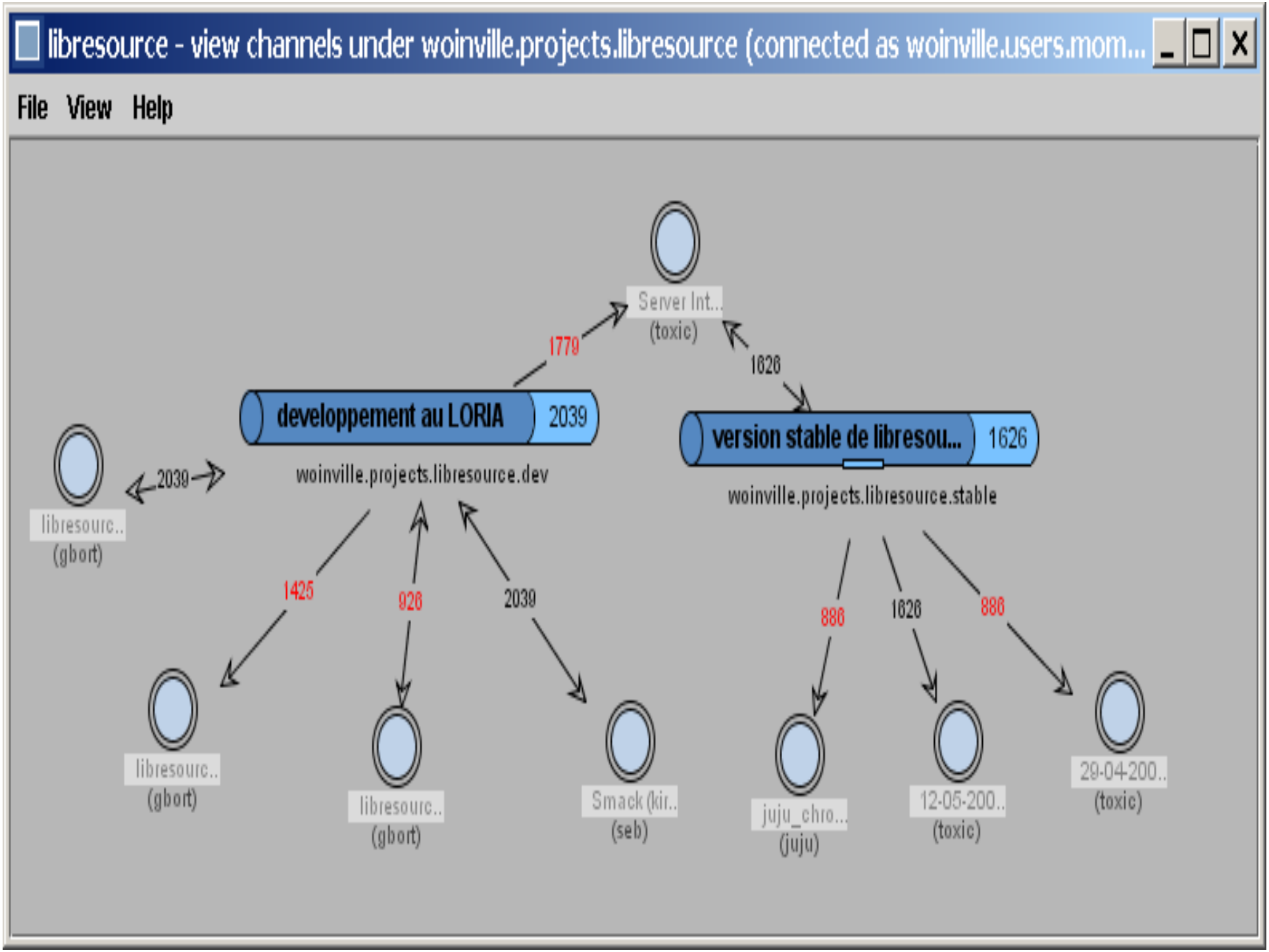
(25)seb.CreateNode(0/seb.13/)(0/seb.13/,"""")
(26)seb.CreateNode(0/seb.13/seb.14/)(0/seb.13/seb.14/,"""")
(27)seb.CreateAttribute(0/seb.13/seb.14/)(name,"""")
(28)seb.CreateNode(0/seb.13/seb.15/)(0/seb.13/seb.15/,"""")
(29)seb.CreateNode(0/seb.13/seb.16/)(0/seb.13/seb.16/,"""")
(30)seb.CreateAttribute(0/seb.13/)(x,""10")
(31)seb.CreateAttribute(0/seb.13/)(y,""10")
(32)seb.CreateAttribute(0/seb.13/)(color,""FFFFFF")
(53)seb.DeleteNode(0/seb.1/)
(56)seb.DeleteNode(0/seb.5/)
(59)seb.DeleteNode(0/seb.9/)
(62)seb.ChangeAttribute(0/seb.13/)(color,""3333FF")
(63)seb.ChangeAttribute(0/seb.13/)(x,""10")
(64)seb.ChangeAttribute(0/seb.13/)(y,""16")

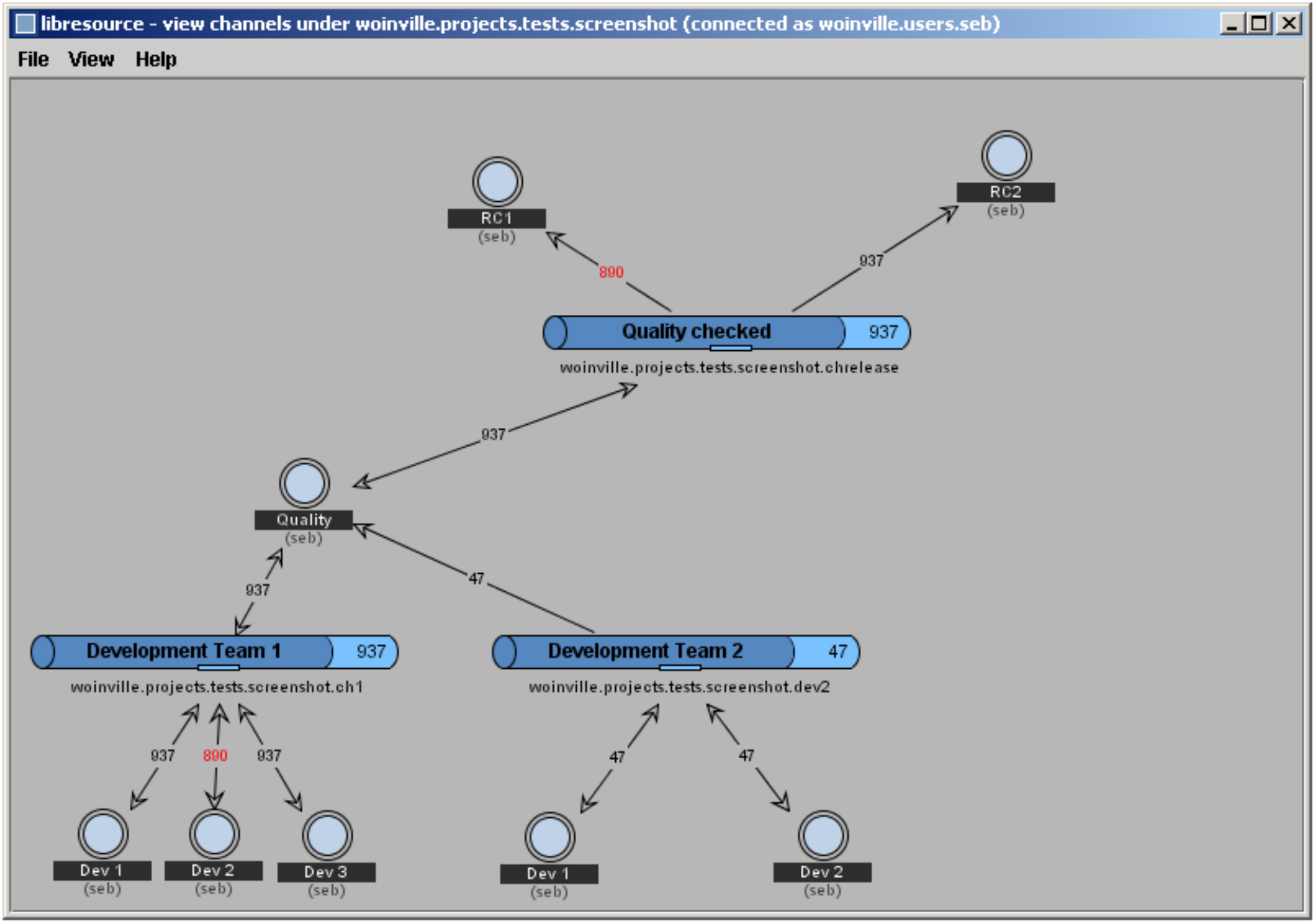
Compress | save | update

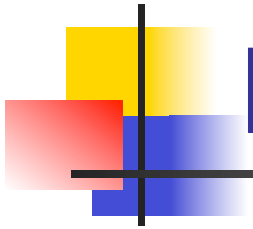


Réseaux de synchronisation

- N channels (log based data sharing)
- 1 Channel:
  - N Replicates (on server or on local computers)
- 1 Replicate **synchronize** with **1-n** channels

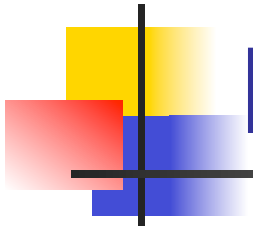

libresource - view channels under woinville.projects.libresource (connected as woinville.users.mom...
File View Help
Server Int..
(toxic)
1779
1626
developpement au LORIA
2039
version stable de libresou...
1626
woinville.projects.libresource.dev
woinville.projects.libresource.stable
2039
libresourc..
(gbort)
1425
926
2039
886
1626
886
libresourc..
(gbort)
libresourc..
(gbort)
Smack (kir..
(seb)
juju_chro...
(juju)
12-05-200..
(toxic)
29-04-200..
(toxic)


libresource - view channels under woinville.projects.tests.screenshot (connected as woinville.users.seb)
File
View
Help
RC1
(seb)
RC2
(seb)
890
937
Quality checked
937
woinville.projects.tests.screenshot.chrelease
937
Quality
(seb)
937
47
Development Team 1
937
woinville.projects.tests.screenshot.ch1
Development Team 2
47
woinville.projects.tests.screenshot.dev2
937
890
937
47
47
Dev 1
(seb)
Dev 2
(seb)
Dev 3
(seb)
Dev 1
(seb)
Dev 2
(seb)



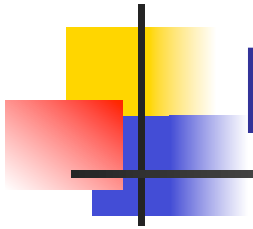
Réseau de synchro

- PB: Propriétés garanties:
  - Propriétés locales -> propriétés globales ??
- Nouvelles propriétés sur le réseau lui-même ?
  - Monotonie ? non-déterminisme ? Continuité ?



Réseaux de synchronisation

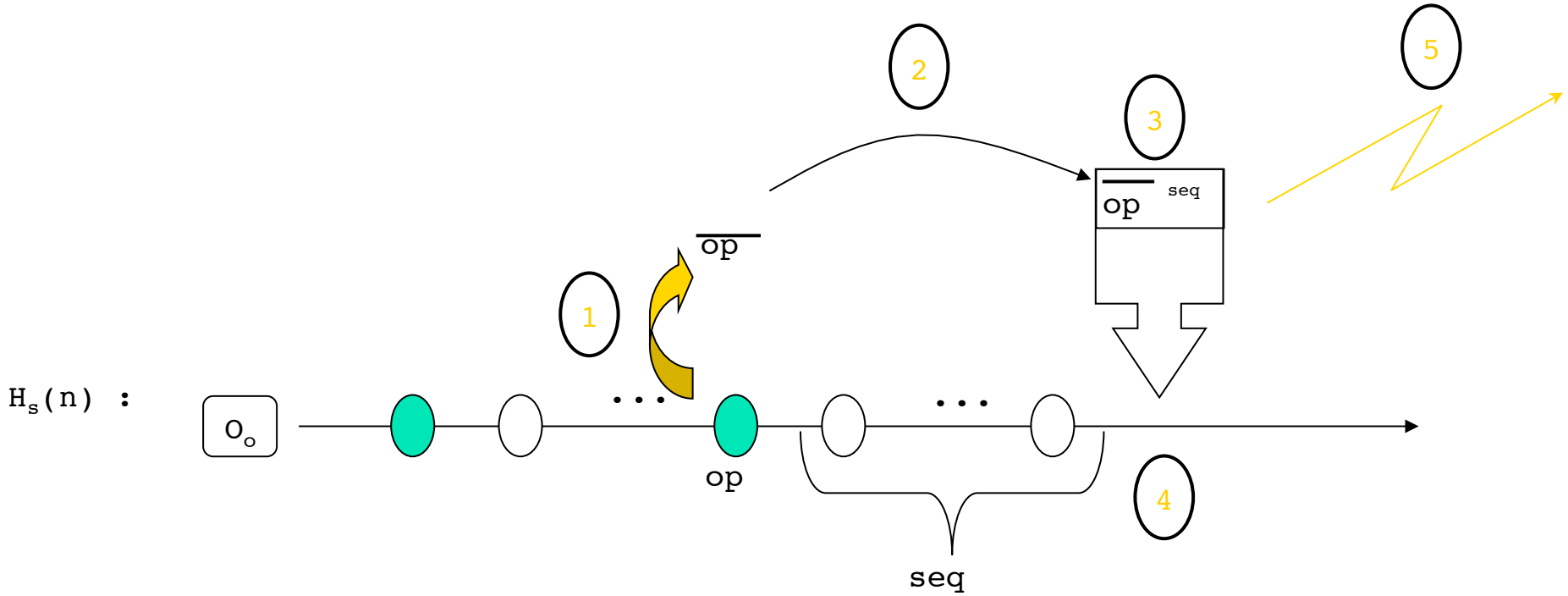
- Application : modélisation et exécution de procédés
  - support pour les itérations
  - Support pour le parallélisme
  - Concurrent engineering…



Pb de l'annulation

- Annuler
  - Pas seulement la dernière opération
  - Une alternative à la compensation…
  - Revenir sur les choix arbitraires des fonctions de transformtion

Algorithme naïf (basé sur le principe 1) (vidot03)



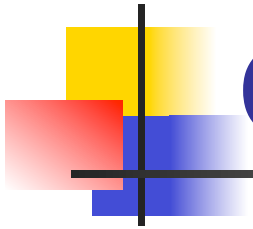
Problèmes Annulation…

- 2 nouvelles propriétés

$$C_3^{'} : op_k^{op_j . \overline{op_j}} = op_k$$

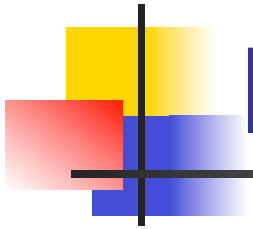
$$C_4^{'} : \overline{\overline{op_j}^{op_i}}^{op_j} = \overline{op_j^{op_i}}$$

- Pb preuves…



Conclusions

- Synchroniser :
  - Un pb ouvert
- Transformations opérationnelles:
  - Un modèle pour raisonner sur cohérence des copies



Perspectives

- Formalisation stricte du modèle
- Transformées opérationnelles vérifiant C1, C2, C'3,C'4 ?
- Transformées opérationnelles et cohérence sémantique
- Métriques de divergence
- Propriétés des réseaux de synchro