

# Sommaire

1. Généralités sur le travail en groupe et sur les systèmes coopératifs
2. La communication médiatisée
3. Les interfaces multi-utilisateurs
4. Les workflow
5. Le Wfmc
6. Les liens entre le traitement des données et le processus de travail
7. La conscience de groupe
8. Les produits groupware
9. Un exemple de groupware : BSCW
10. Deux applications particulières de travail en groupe

# Définitions

## **Travail coopératif**

Il est constitué de processus de travail liés par la nature de leur contenu, c'est-à-dire qui appartiennent à la production d'un produit particulier.

## **Groupware ou Computer Supported Cooperative Work (CSCW)**

Ensemble d'aides informatiques spécialisées conçues pour être utilisées par des groupes de travail coopératif formés autour d'un projet.

## **Coopération étroite**

- le groupe est bien établi
- le rôle de chacun et les règles sont bien définis
- chaque nouvelle conversation est liée aux précédentes
- chaque relation entre deux personnes est significative pour les autres

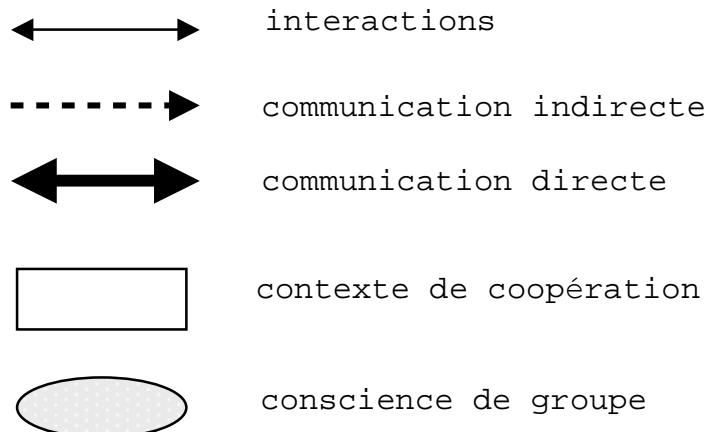
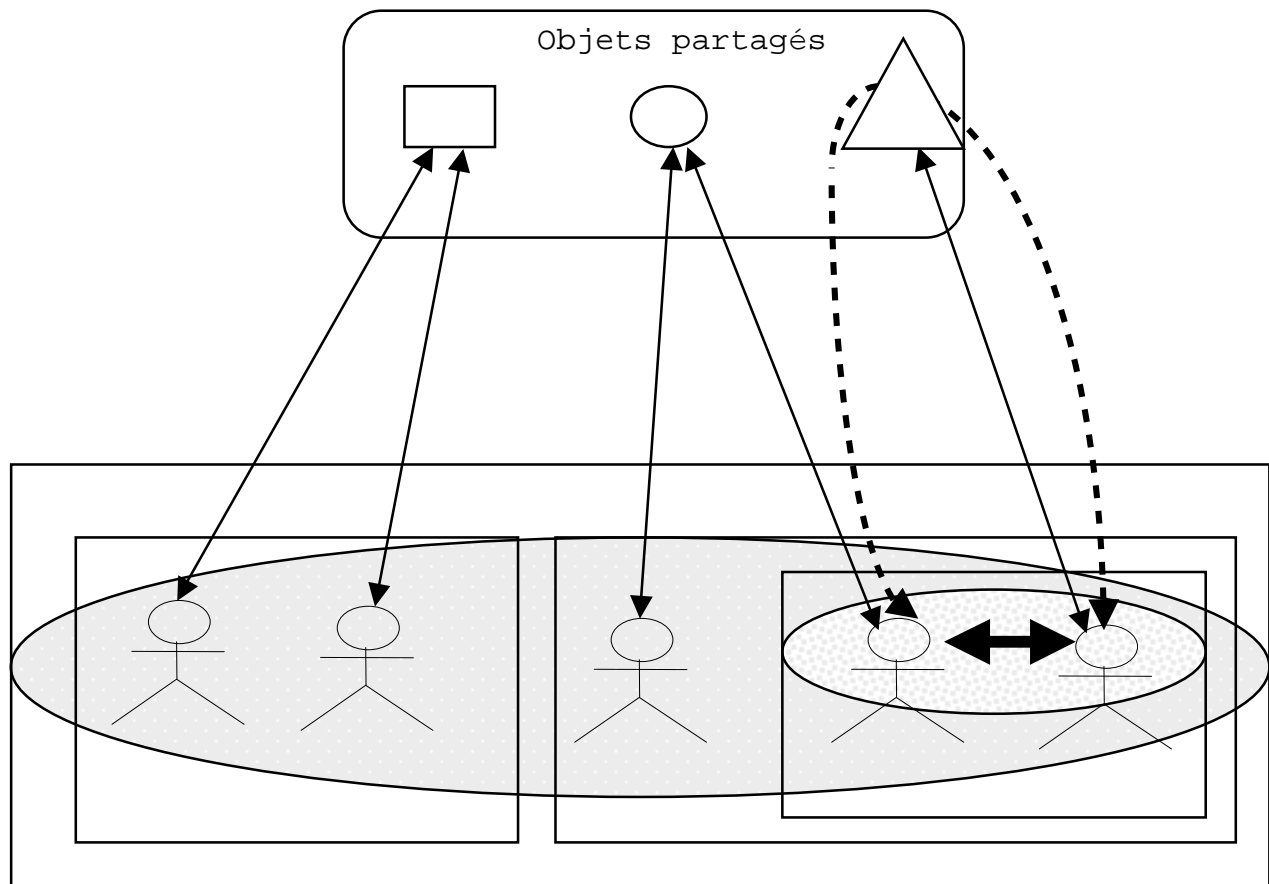
## **Coopération large**

- le groupe est large, mal délimité et changeant dans le temps
- les rôles sont mal ou pas définis
- les interactions entre les individus sont liées par un sujet d'intérêt mais sont occasionnelles
- les conversations ne sont pas nécessairement liées

# Modèle élémentaire de travail coopératif

Le travail coopératif et ses technologies, Jaques Lonchamp, Lavoisier

*Communication, production/partage, coordination/collaboration/co décision*



# Différents aspects du travail de groupe

- **coordination**

synchronisation des personnes et des actions

cohérence des actions individuelles/l'ensemble du processus

- **collaboration**

contribution des individus pour produire un savoir partagé

compréhension commune de l'objectif et du processus

impossibilité d'isoler la contribution de chacun à l'obtention du résultat

- **codécision**

voisin de la collaboration

# Besoins de support

## **Pour la communication**

utilisateur-ordinateur et  
utilisateur(s) - utilisateur(s)

## **pour la coordination**

- distribuer et sélectionner facilement les messages
- lier des messages au sein d'une conversation
- enregistrer et classer des messages-conversations avec leur statut en cours
- modéliser le coordination
- classer et sélectionner les processus récurrents (actes de paroles, requêtes, approbations, ...)

## **pour la collaboration**

- structurer l'information pour qu'elle reflète la manière dont elle a été créée
- accéder aux informations selon le rôle
- assister les flux de questions-réponses sur la tâche en cours
- partager un espace commun avec la vision pour chacun de ce que font les autres

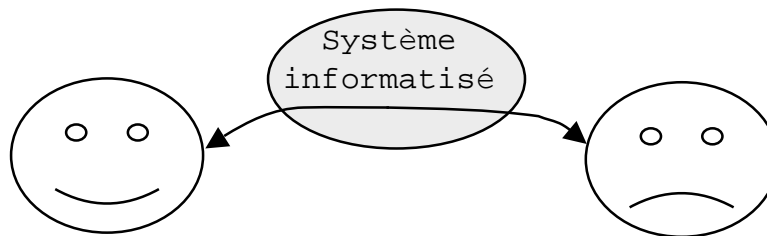
## **pour la codécision**

- partager toute l'information utile pour prendre la décision
- partager les critères de décision
- partager les décisions déjà prises au cours du processus
- gérer les conversations pour possibilités

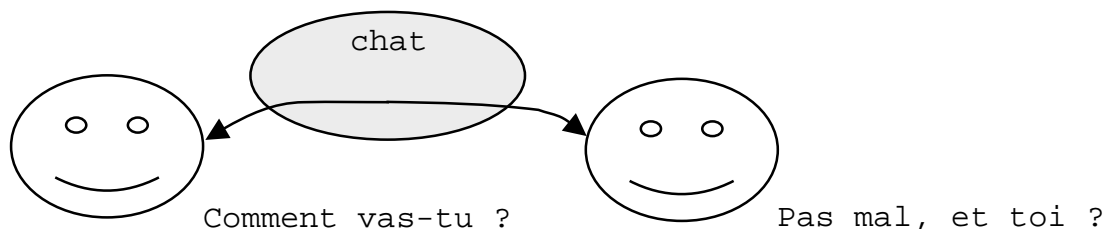
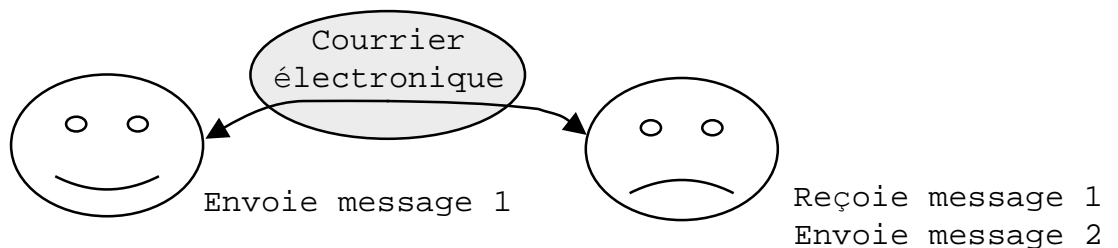
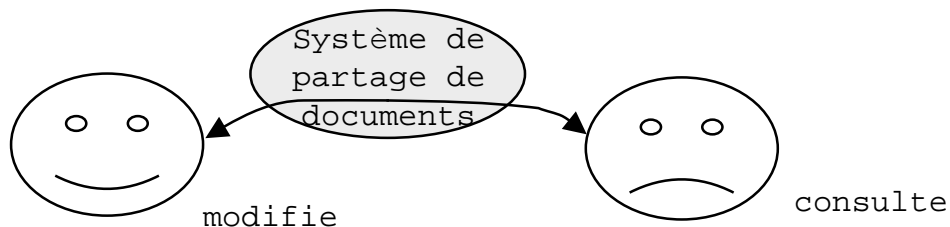
# Systeme coopératif

**Tout système informatisé visant à assister un groupe d'utilisateurs qui travaillent ensemble et interagissent dans le but de réaliser une tâche commune**

Les utilisateurs interagissent via le système informatisé et ils ont une *conscience mutuelle* de leurs activités



Quelques exemples



## **Le challenge**

Passer de l'ordinateur outil de production individuelle  
à  
l'ordinateur outil de production collective

## **Les raisons**

- Le passage d'une économie d'industries manufacturières dépendantes de la circulation des biens vers l'industrie des services dépendante de la circulation d'informations
- La décentralisation et l'extension géographique des opérations
- Développement des réseaux et d'Internet facilitant le passage d'un réseau utilisé comme média passif de publication d'informations à une plateforme de développement d'applications interactives puis à un espace de travail pour des équipes distribuées (voir le site SourceForge de développement de logiciels libres Open source)
- Expansion des connections permanentes facilitant le passage de formes asynchrones de coopération à des formes synchrones

# **Les différents types de systèmes coopératifs**

## **Deux classifications :**

- du point de vue de l'espace et du temps
- du point de vue des fonctionnalités



## Classification du point de vue de l'espace et du temps

	<i>Même moment</i>	<i>Moments différents</i>
<i>Même lieu</i>	Interactions face à face	Interactions asynchrones
<i>Lieux différents</i>	Interactions distribuées synchrones	Interactions distribuées asynchrones

	<i>Même moment</i>	<i>Moments différents</i>
<i>Même lieu</i>	Aide aux réunions	Base de documents
<i>Lieux différents</i>	Tableau blanc partagé	Co-conception de document
	Télé conférence	Messagerie

## **Classification du point de vue des fonctionnalités**

- **routage**

- messagerie électronique
  - workflow administratif

- **modélisation de processus, systèmes de coordination**

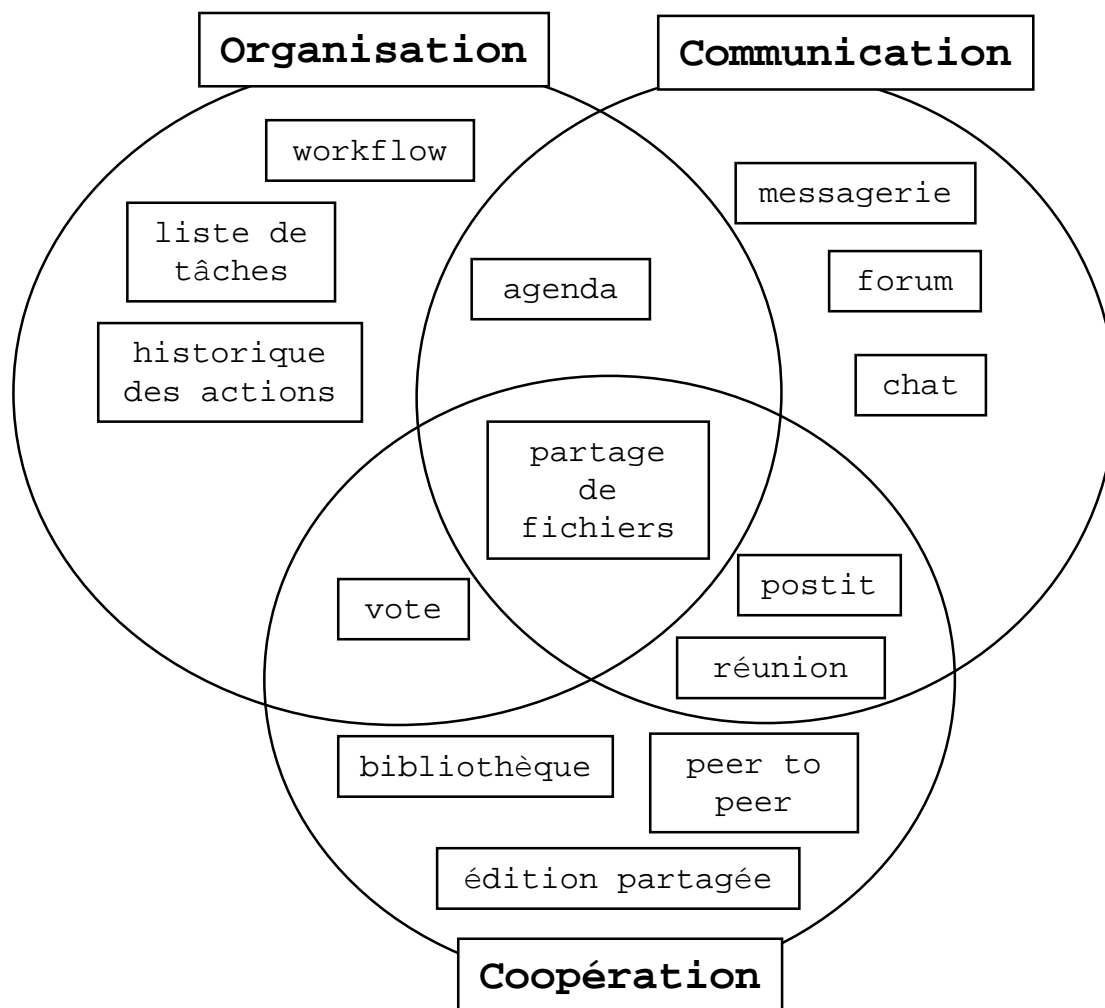
- workflow coopératif

- **décision de groupe**

- suivi : agenda de groupe, suivi de projet
  - conférence : forum, rédaction coopérative, réunions thématiques (avec classification des propos, organisation de votes, ...), vidéo-conférence

- **gestion électronique de documents**

- mémoire : documentation qualité ISO9000, bibliothèque de projet
  - kiosque : lettre d'information, revue de presse



# **Des outils à la conjonction de cinq disciplines**

- **Les systèmes distribués**

- décentralisation du contrôle et des données
  - maintien de la cohérence globale
  - récupération sur panne

- **les communications**

- utilisation de protocoles d'échange

- **les interactions homme-machine**

- développement d'interfaces homme-machine de groupe

- **l'intelligence artificielle**

- adaptation de l'utilisation de l'outil à différents profils d'utilisateurs

- **la sociologie**

- étude de protocoles d'interactions et de travail

# **Ce que doit prendre en compte un système coopératif**

## **Les aspects sociologiques :**

- La disparité entre le travail supplémentaire demandé et les bénéfices obtenus
- Le pourcentage d'utilisateurs du système
- Les mises en cause de règles sociales qui régissent le fonctionnement de l'organisation
- La prise en compte des exceptions et improvisations qui caractérisent la plupart des tâches collectives
- La cohabitation et la transition entre travail individuel et travail collectif
- L'efficacité à la fois des tâches individuelles et des tâches collectives
- La difficulté d'adoption des systèmes coopératifs
- Le dilemme entre la protection de la vie privée et le besoin de rendre visible l'activité de chacun pour favoriser la coopération

## **Les aspects techniques**

- Système multi-utilisateurs qui, ne doivent pas donner l'illusion à chaque utilisateur d'être seul, mais au contraire assister et encourager la propagation des activités entre utilisateurs
- Assurer la synchronisation entre les contributions des différents utilisateurs
- Gérer les processus
- Gérer la concurrence et le contrôle des accès
- Mettre en place la persistance des informations avec des fonctionnalités appropriées par exemple, comparaison et fusion de versions
- Prendre en charge les aspects distribués : communications à travers les réseaux, ordonnancement des messages échangés, réplication et synchronisation des objets
- Gérer le couplage de vues et la conscience de groupe
- Présenter des propriétés ergonomiques

# La communication médiatisée

Faciliter la communication homme-homme et fournir au groupe la possibilité de parvenir à une forme d'intelligence collective

## Les caractéristiques :

- La visibilité : les interlocuteurs se voient mutuellement
- L'audibilité : les interlocuteurs s'entendent mutuellement
- La co-temporalité : l'émission et la réception d'un message sont presque simultanées
- La simultanéité : les interlocuteurs peuvent émettre et recevoir simultanément
- La séquence : la communication se fait par une succession de tours de parole
- La mémorisation : chaque interlocuteur peut consulter les précédents messages reçus
- La révision : chaque interlocuteur peut réviser un message avant de l'envoyer

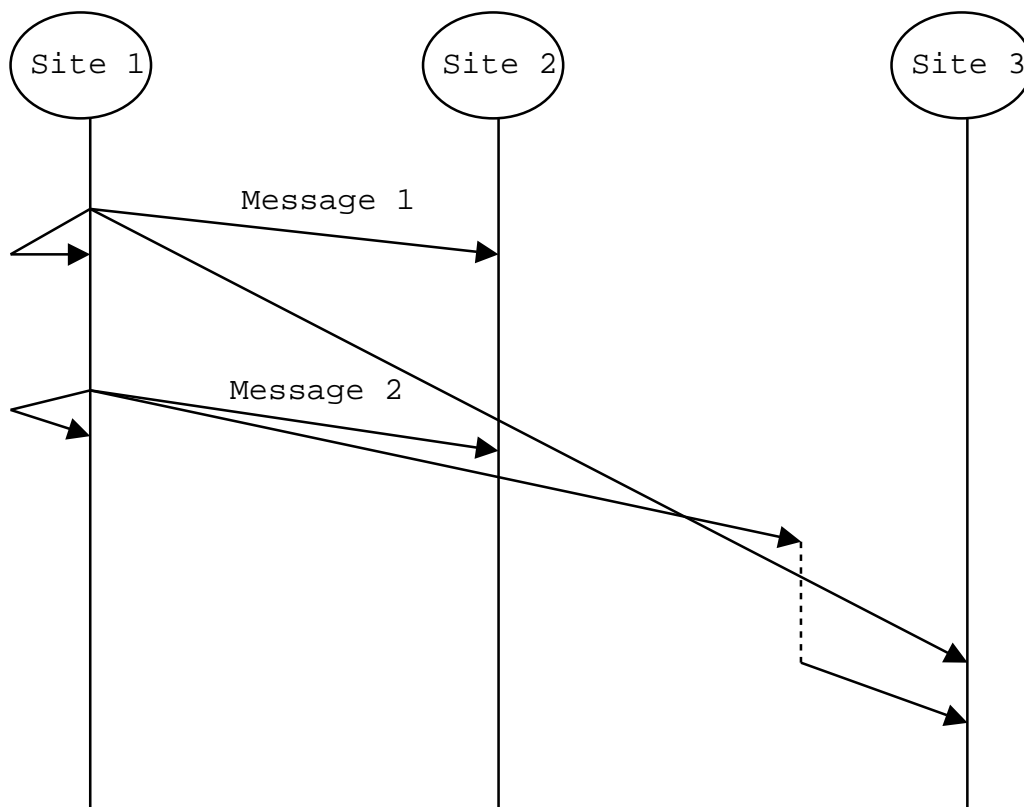
**Outils type :**

- les conférences asynchrones textuelles ou forums électroniques
- Les messageries instantanées et les outils de discussion en ligne (chats)
- Les conférences synchrones audio/vidéo
- L'assistance de groupe, à la prise de décision
- Le partage d'applications
- Les coordinateurs de conversation
- La circulation de documents
- Les environnements de collaboration virtuelle
- Le CoWeb



## L'ordre FIFO :

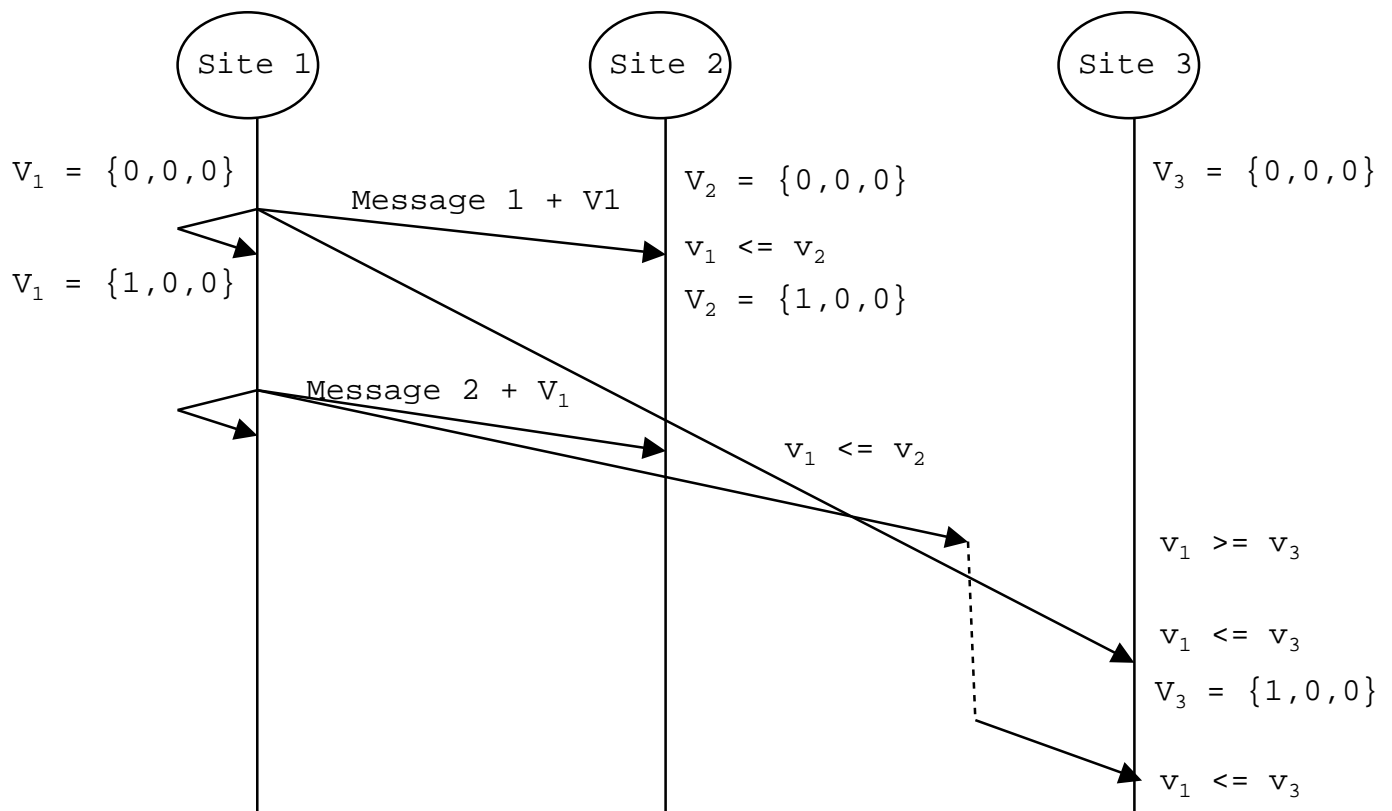
Les messages mal ordonnancés sont retenus jusqu'à réception des messages qui le précèdent



## L'ordre causal

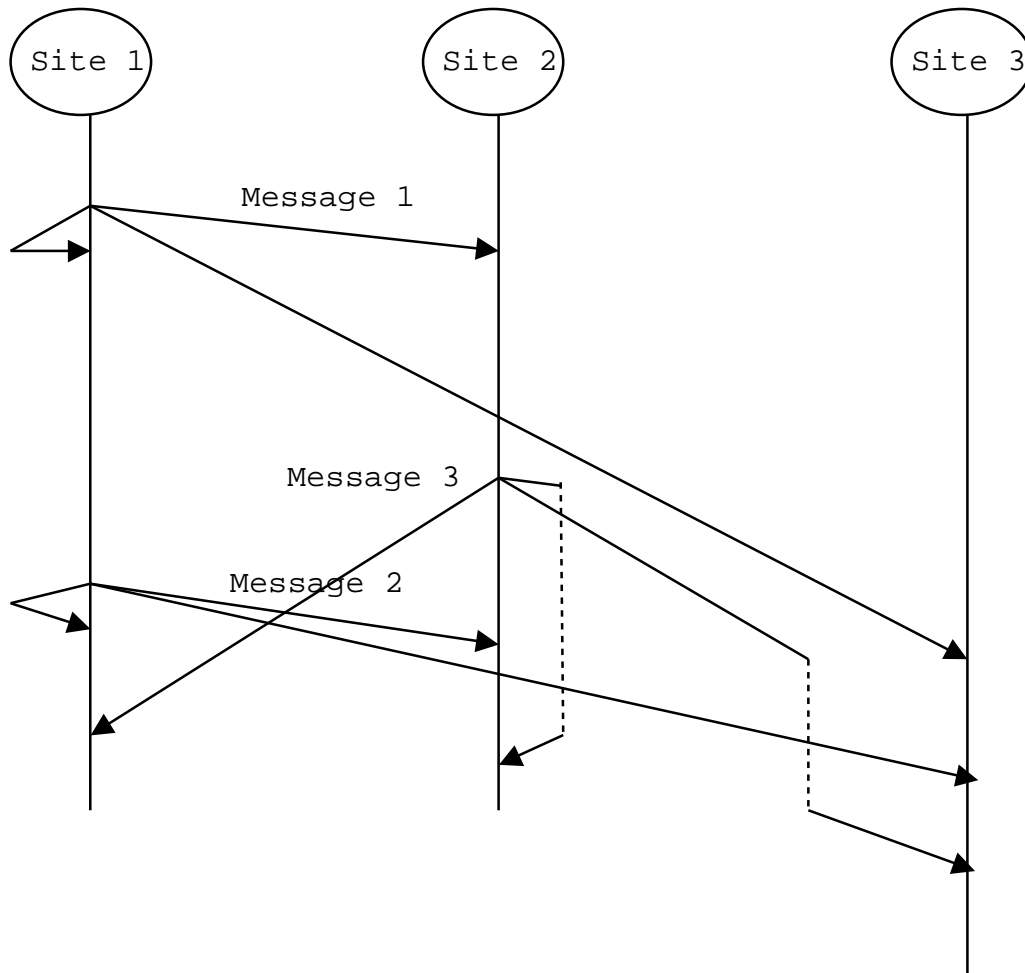
La livraison des messages est faite selon l'ordre FIFO

Si un site  $S_i$  envoie un message  $m2$  après avoir reçu un message  $m1$  alors tous les sites qui reçoivent  $m1$  et  $m2$  les reçoivent dans cet ordre



## L'ordre total

Tous les sites reçoivent les messages dans le même ordre déterminé par un arbitre



# **Les interfaces multi-utilisateurs**

## **Représentation de l'espace**

- Les métaphores
- Le couplage de vues

## **Représentation du temps**

Les sessions coopératives explicites ou implicites

## **Les métaphores :**

- La procédure : chaque utilisateur peut exécuter un ensemble fini d'actions dont le résultat et l'impact sur les vues des autres utilisateurs sont déterminés à l'avance
- La réunion : la coopération entre les utilisateurs nécessite qu'ils soient présents au même instant dans l'espace défini par l'activité auquel ils contribuent. L'activité se termine avec le départ du dernier utilisateur. Les objets produits ne persistent pas sauf action spécifique de sauvegarde.
- La pièce : L'espace partagé est composé d'un ensemble de pièces. Une pièce correspond à une activité. Elle contient les documents et les outils nécessaires pour mener l'activité. Lorsque deux utilisateurs se trouvent dans la même pièce, ils peuvent travailler ensemble sur les documents contenus dans la pièce. Pour définir une activité, on crée une pièce et on y met des documents. Un mécanisme de déplacement entre les pièces est fourni par le système.
- Le terrain : L'espace partagé est représenté par une structure abstraite contenant des données et dotée d'un système d'aide à la navigation entre les données. Les activités ne sont pas planifiées à l'avance. C'est l'accès simultané à un même document qui engendre une activité.

## **Le couplage de vues :**

- Mode WYSIWIS (What you see is What I see) : toutes les vues des différents participants à une même activité sont similaires (y compris la position du curseur et de la barre de défilement)
- Mode WYSIAWIS (What you see is almost what I see) : les répercussions sur les autres vues des modifications sont différées
- Mode WYSIWIMS (What you see is what I may see) : les modifications des données ne sont pas répercutées systématiquement sur les autres vues mais les utilisateurs peuvent les demander quand ils veulent.

## **La représentation des données**

Même si différents participants travaillent sur les mêmes données, chacun peut choisir son mode de représentation indépendamment du choix des autres utilisateurs

## Les sessions coopératives

- explicite sur invitation de l'initiateur
- explicite sur initiative des participants
- implicite sur partage du même objet
- implicite sur partage de la même pièce

### Les problèmes à résoudre :

- gestion des retardataires
- persistance des sessions
  - ✓ la session persiste lorsqu'aucun participant n'est actif; les utilisateurs peuvent rejoindre et quitter la session à leur guise
  - ✓ la session commence lorsque le premier participant la lance et se termine lorsque le dernier participant la quitte

# **Des systèmes coopératifs typiques**

Les systèmes de gestion de processus (workflows)

Les systèmes de gestion d'espace de travail partagé (produits groupware)



# **Les workflows**

## **Qu'est-ce qu'un workflow ?**

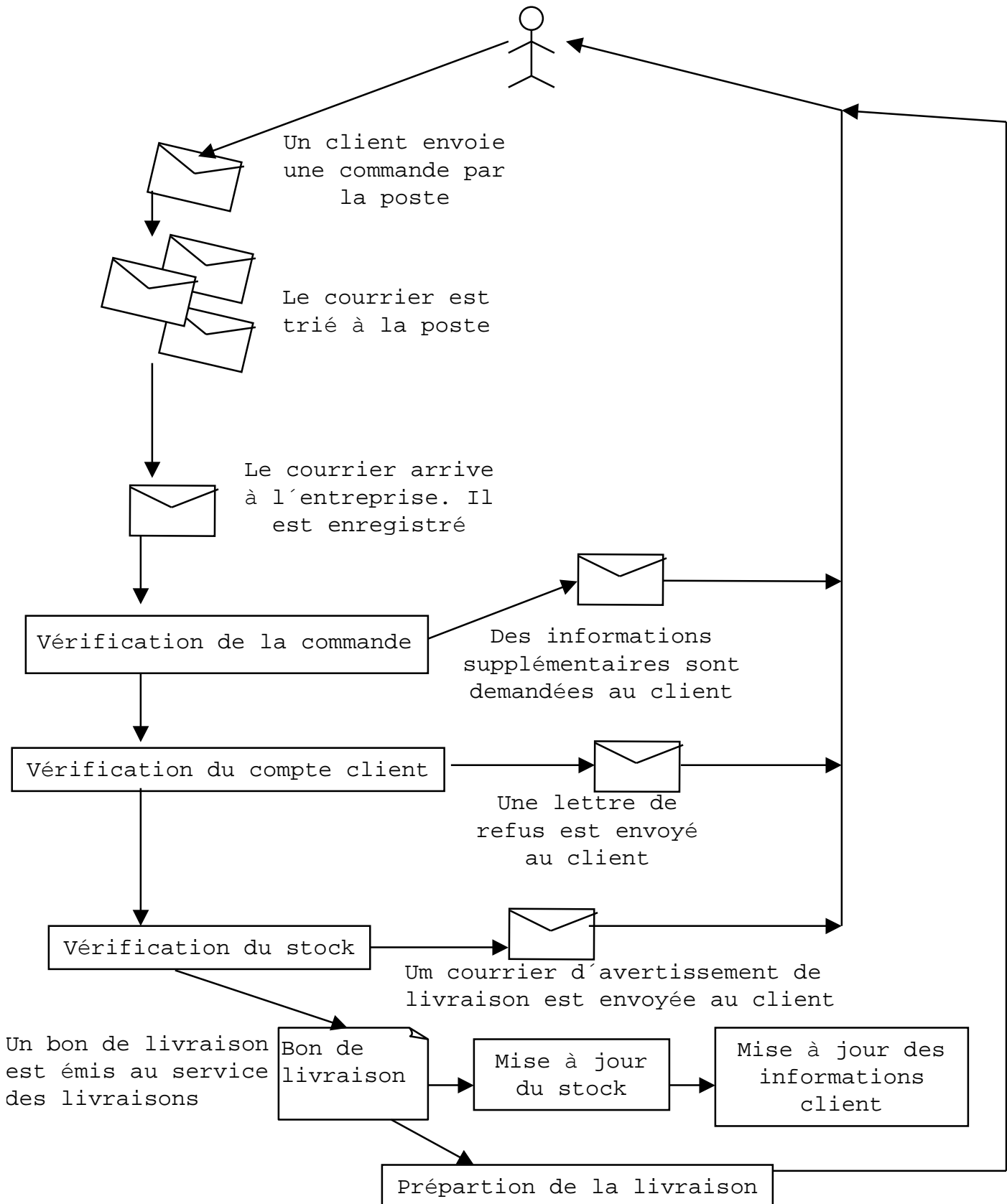
Un workflow est l'automatisation de tout ou partie d'un processus durant lequel des documents, des informations ou des tâches sont transférés d'un participant à un autre en vue d'actions réalisées conformément à des règles pré-établies.

## **Qu'est-ce qu'un système de gestion de workflow (workflow management system, WFMS)**

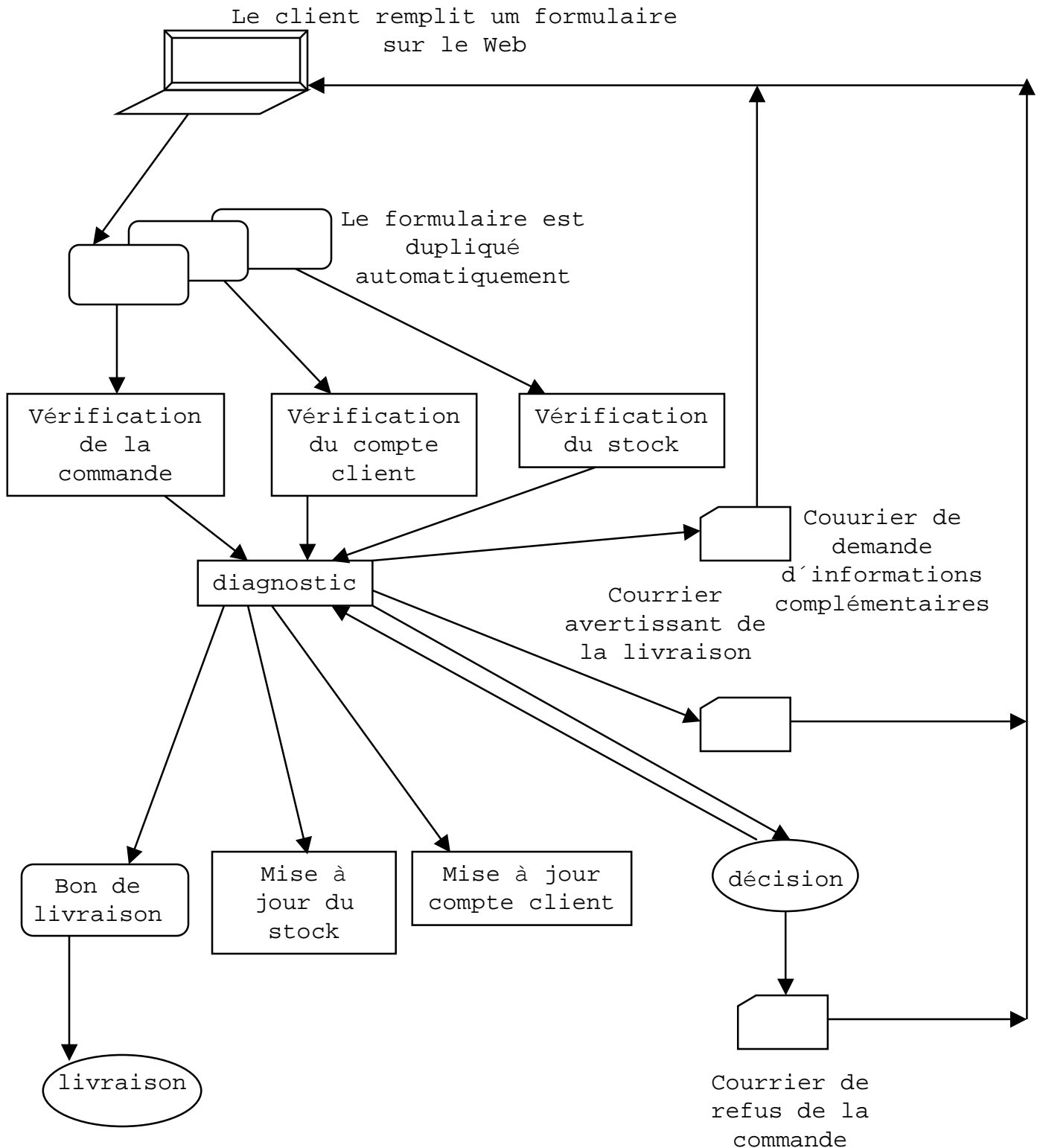
Un système de gestion de workflow est un système informatique qui permet de définir, créer et contrôler l'exécution de workflow. Il est capable d'interpréter la définition du processus, interagir avec les participants et si besoin est invoquer des applications spécifiques.

# Exemple de processus

client



# Workflow



## Différents types de workflow

- workflow administratif

peu de processus mais utilisé souvent

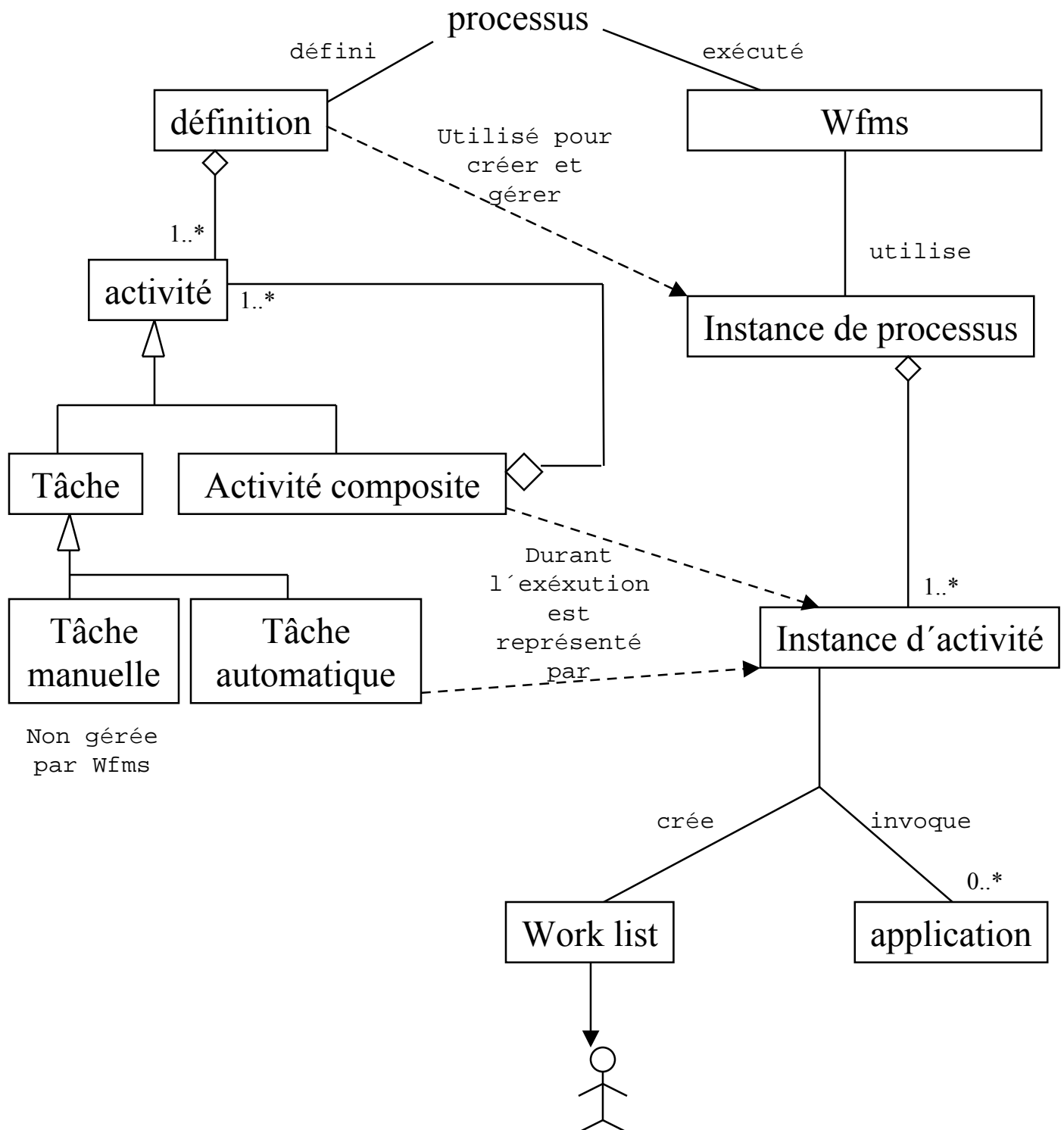
- workflow coopératif

implique la participation de plusieurs équipes délocalisées qui doivent communiquer

- workflow ad-hoc

definitions nombreuses de processus et facilement modifiables

# Principes généraux des systèmes de workflow



## **Workflow administratif**

Informatisation de la succession de traitements liés à des formulaires administratifs,

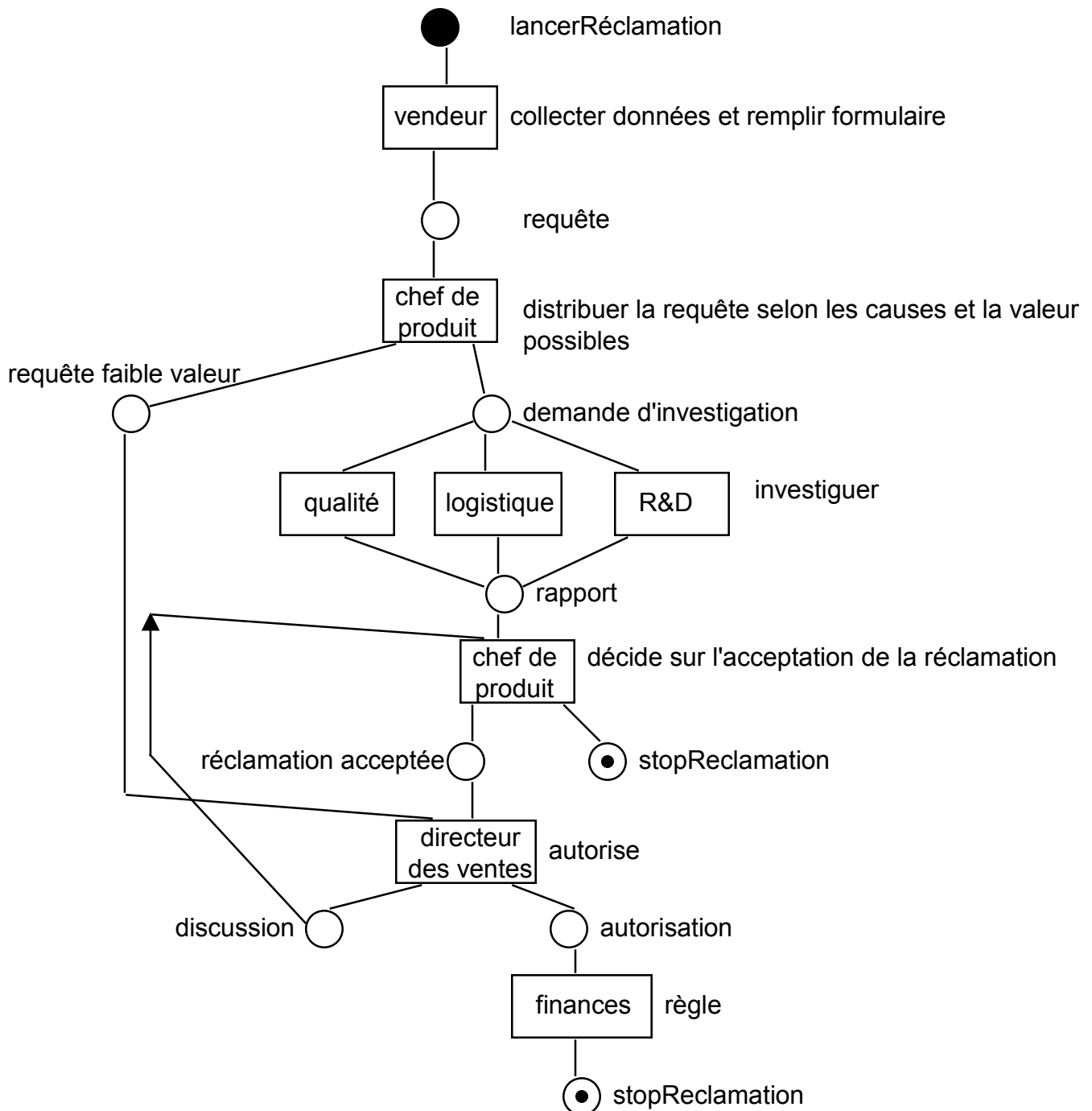
Par exemple, les traitements liés à l'arrivée d'une demande de réclamation

Avantages :

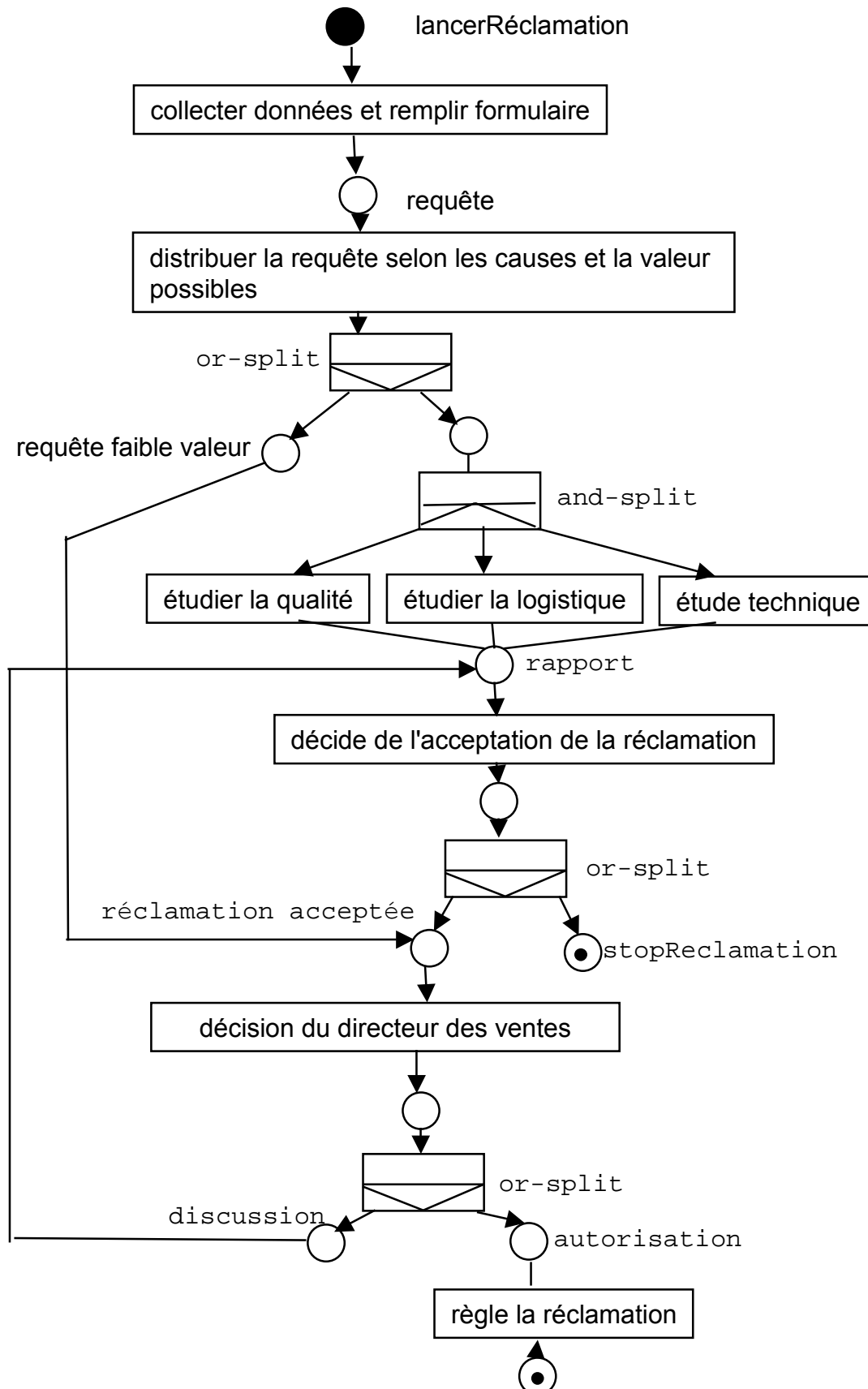
- accélération des traitements
- meilleure gestion des formulaires (plus de papier)
- plus grande facilité de remplissage des formulaires
- plus grande facilité de validation
- retour rapide d'informations à l'émetteur
- possibilité d'alimenter automatiquement les différents systèmes d'information de l'entreprise
- plus de risque de perte de données

# Workflow administratif

## Modélisation des procédures par un réseau de Pétri simple



## Modélisation des procédures par un réseau de Pétri enrichi





## Modélisation par un langage spécifique

**CoPlan** (Co-ordinated Procedure Language) [T. Kreifelts, U. Licht, G. Woetzel, *DOMINO: A System for the Specification and Automation of Cooperative Office Processes*. In proceedings EUROMICRO'84, North Holland, Amsterdam:33-41]

PROCEDURE: ReclamationClient  
REQUIRES: lancerReclamation  
PRODUCES: stopReclamation

ACTOR-A action 1: vendeurCollecteDonneesEtRemplitFormulaire  
REQUIRES: lancerReclamation  
PRODUCES: requête

FORM1: requête  
ATTRIBUTE1: nomClient  
ATTRIBUTE2: designationProduit  
ATTRIBUTE3: valeur

.....  
ENDFORM1

ACTOR-B action2:chefDeProduitDistribuerRequeteSelonCauseEtValeurPossibles  
REQUIRES: requête  
IF valeur FROM requête CONDITION inférieurà 100%  
PRODUCES: requeteFaibleValeur  
ELSE  
PRODUCES: demandeInvestigation

.....

## **Fontionnalités orientées objet dans les logiciels de workflow**

Le workflow est une classe :

Chaque exécution du workflow est une instance de la classe possédant ses propres valeurs d'attributs (par exemple son état, l'historique d'exécution)

L'héritage et la spécialisation de workflow

La composition de workflow

Les objets gérés par le workflow sont aussi des instances de classes :

L'héritage

La composition

# Modélisation des processus métier

## Modèle *langage-action*

fondé sur la théorie de **l'acte de parole** [J.L. Austin, *How to do things with words*, Harvard University Press, Cambridge, Massachusetts], [J.R. Searle, *A Taxonomy of Illocutionary Acts*, in K. Gunderson (ed.) *Language, Mind and Knowledge*, University of Minnesota, Minneapolis: 344-369]

### Cinq catégories d'énonciations illocutoires :

- assertive : lelocuteur s'engage dans ce qu'il affirme
- directive : le locuteur fait une demane à son intrelocuteur
- commissive : le locuteur promet
- déclaration : le locuteur prononce un fait ou une réalité
- expressive : le locuteur exprime un état psychologique

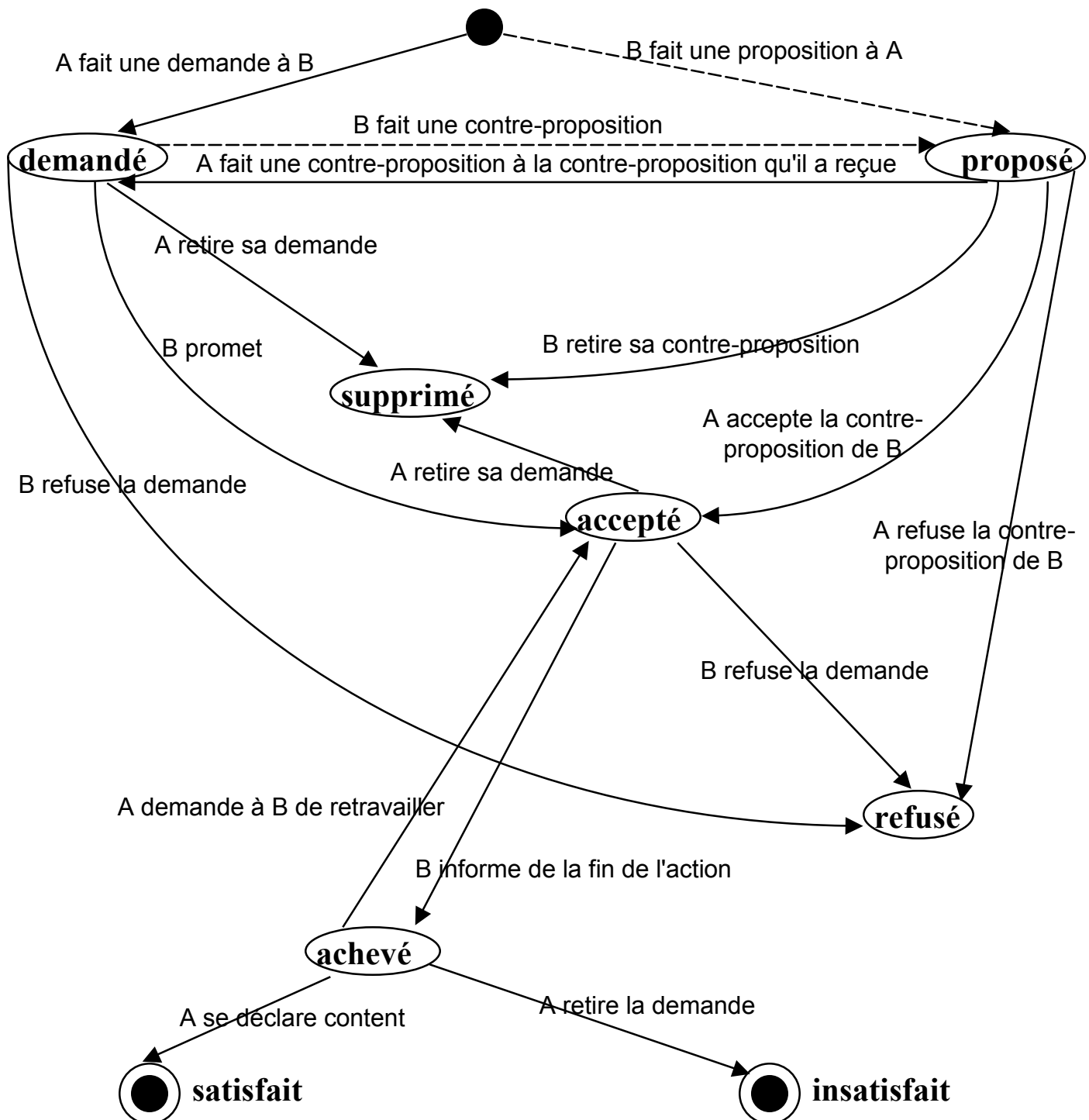
Une conversation = des actes de paroles échangés entre deux partenaires

### Quatre types de conversations :

- ***la conversation pour action*** : les deux partenaires négocient une action que l'un des deux fera
- ***la conversation pour possibilités*** : les deux partenaires négocient une modification du contexte dans lequel ils agissent, le but étant de générer une conversation pour action
- la conversation pour clarification : les partenaires gèrent ou anticipent des blocages concernant une conversation pour action
- la conversation pour orientation : les partenaires essaient de créer un contexte pouvant servir de base à des conversations futures

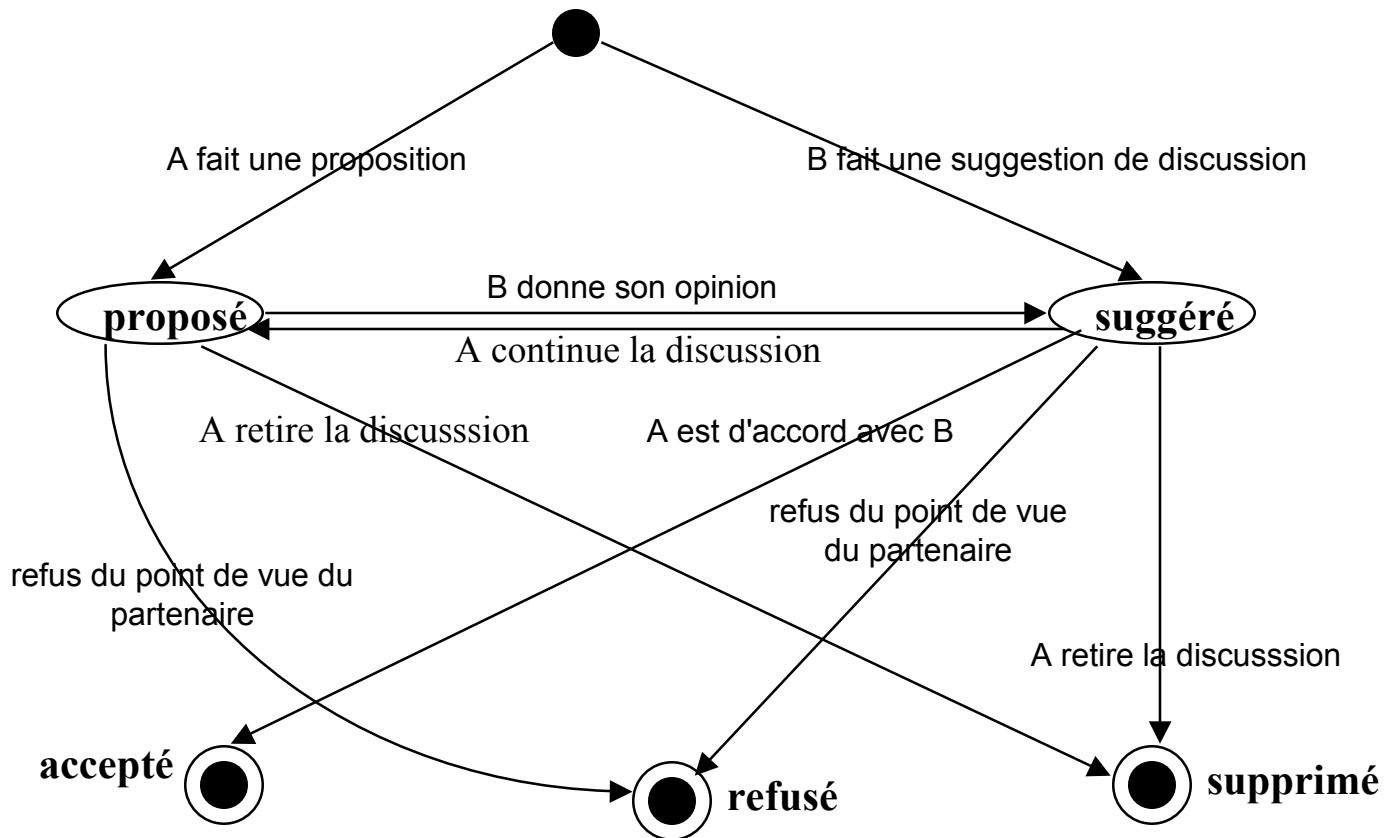
# Conversation pour action

Diagramme d'états-transitions (d'après T. Schael, *Théorie et pratique du workflow*, Springer)



# Conversation pour possibilités

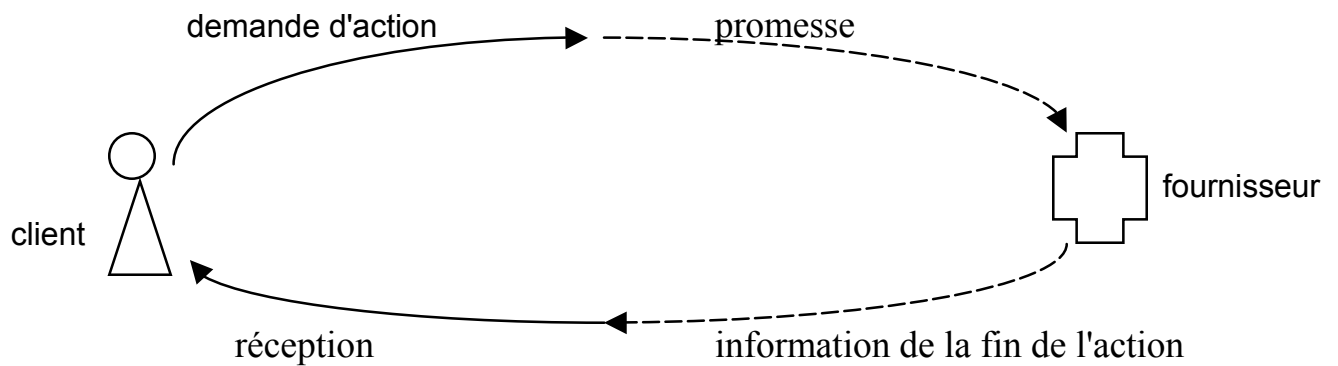
Diagramme d'états-transitions (d'après T. Schael, *Théorie et pratique du workflow*, Springer)



# Application à la modélisation des workflows coopératifs

## Modèles de base

### Le workflow d'action



contient implicitement tous les états d'une conversation pour action

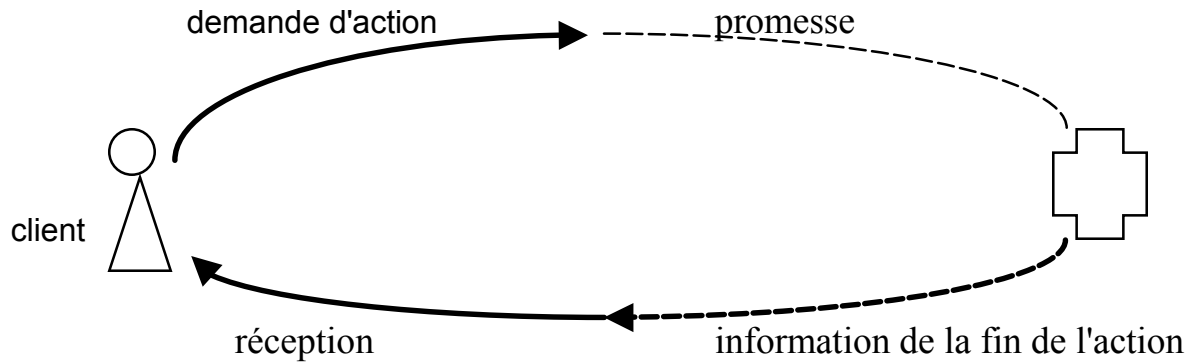
### Le workflow de déclaration



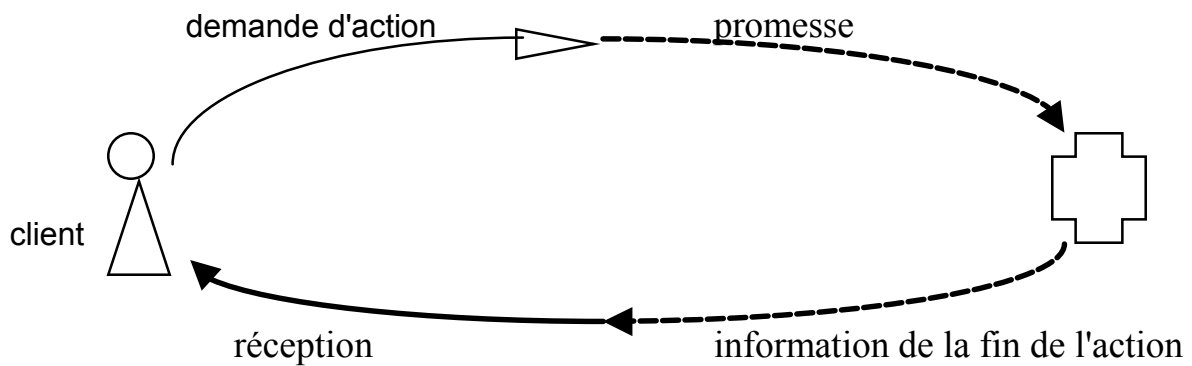
contient implicitement tous les états d'une conversation pour possibilités

# Variantes

## workflow d'action sans phase d'engagement



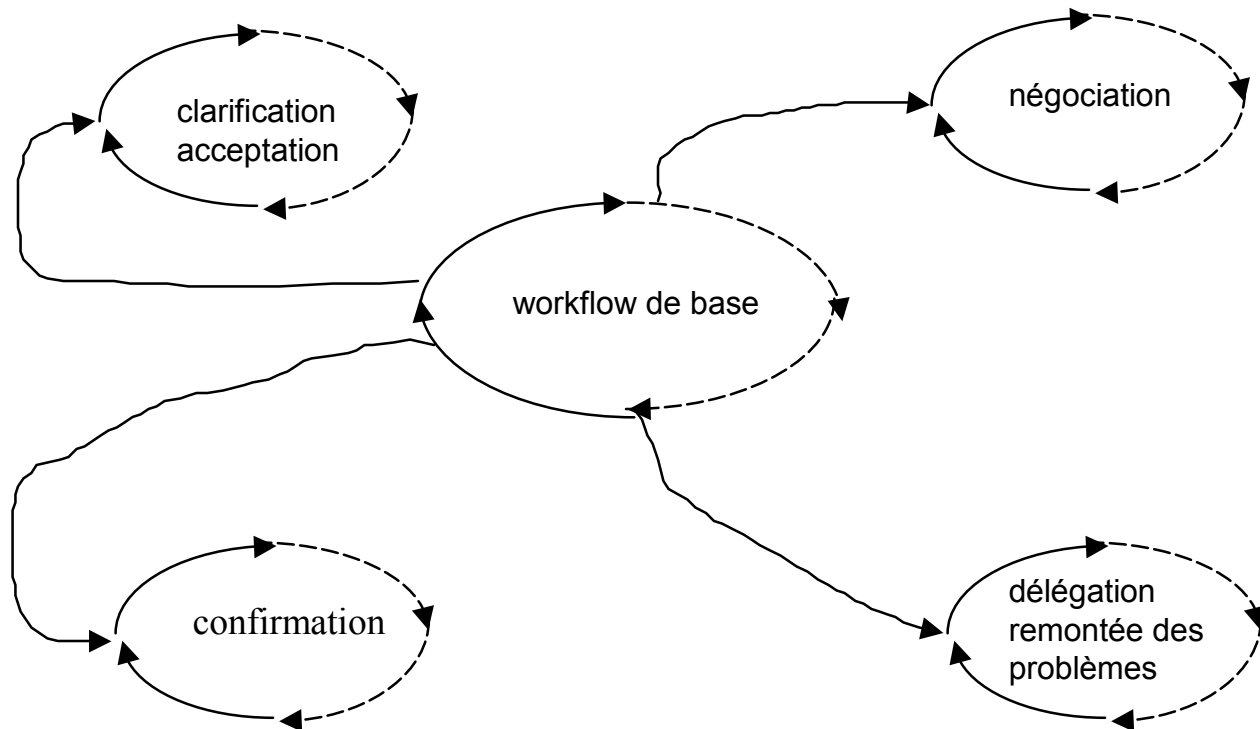
## workflow d'action avec phase de requête automatisée



## workflow de déclaration sans phase de continuation



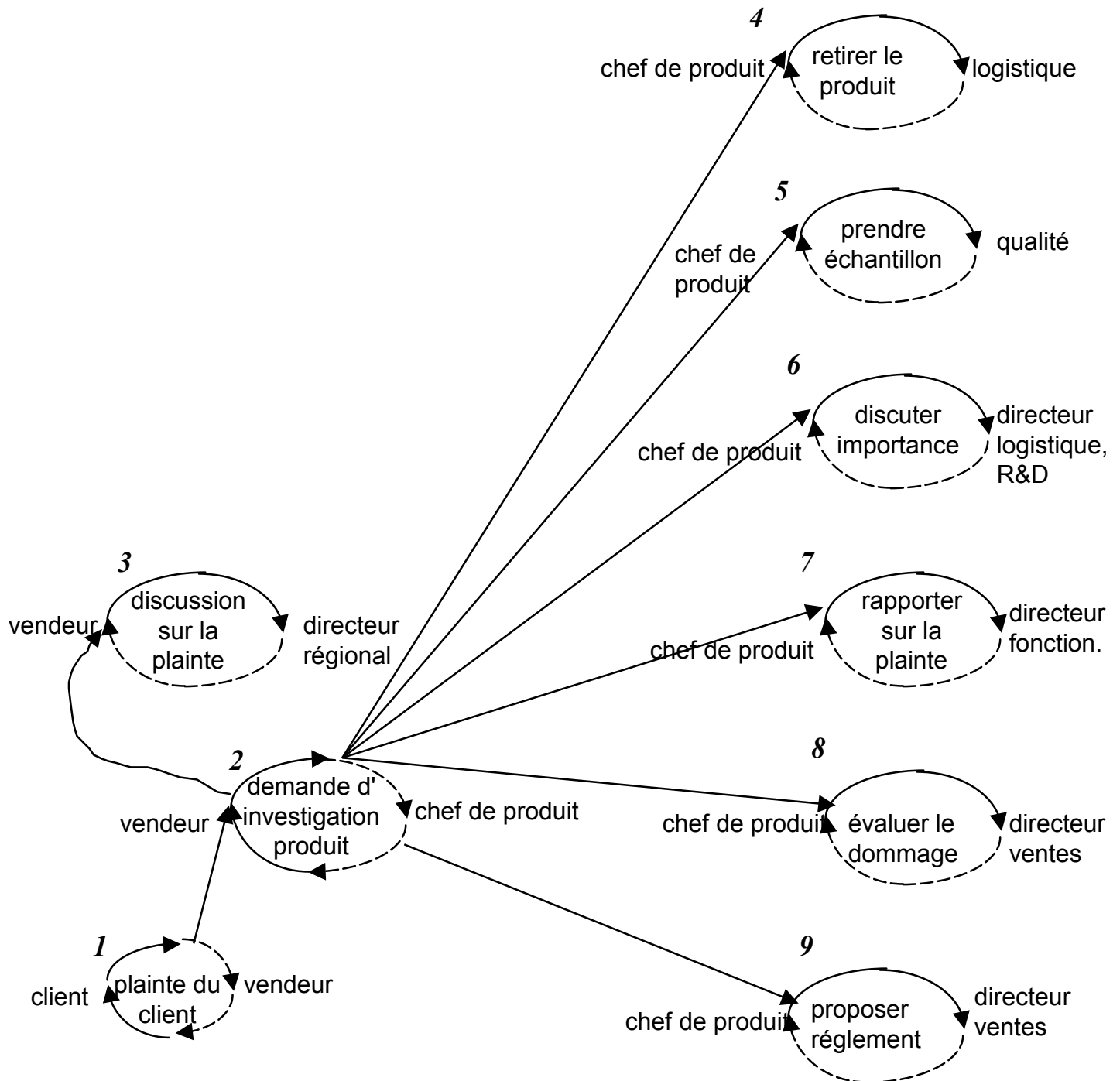
# Modèle de workflow articulé





# Exemple

## Gestion des réclamations des clients



## La modélisation orientée règles

exemple [Coordination in workflow management systems – a rule-based approach, G. Kappel, S. Rausch-Schott, W. Retschitzegger, Coordination technology for Collaborative applications – Organizations, Processes, and Agents, LNCS 1364, Springer Verlag, 1997, p : 99-120] :

```
define rule ReConception
  on post (activity, perform :actFolder) [trgObj==analyseRapports]
do
  if (((actFolder docNamed:'Rapport') errortype='conceptual'
and: (actFolder docNamed:'Rapport') percentage > 10 then
    execute ReConception notifyAgent:actFolder
end rule ReConception
```

```
define rule Nouvellesinstructions
  on post (activity, perform :actFolder) [trgObj==analyseRapports]
do
  if (((actFolder docNamed:'Rapport') errortype='conceptual'
and: (actFolder docNamed:'Rapport') percentage <= 10 then
    execute NouvellesInstructions notifyAgent:actFolder
end rule NouvellesInstructions
```

```
define rule CreationRapport
  on post (activity, perform :actFolder) [trgObj==analyseRapports]
do
  if (((actFolder docNamed:'Rapport') errortype='other' then
    execute CreationRapport notifyAgent:actFolder
end rule CreationRapport
```

# **WMFC : Workflow Management Coalition**

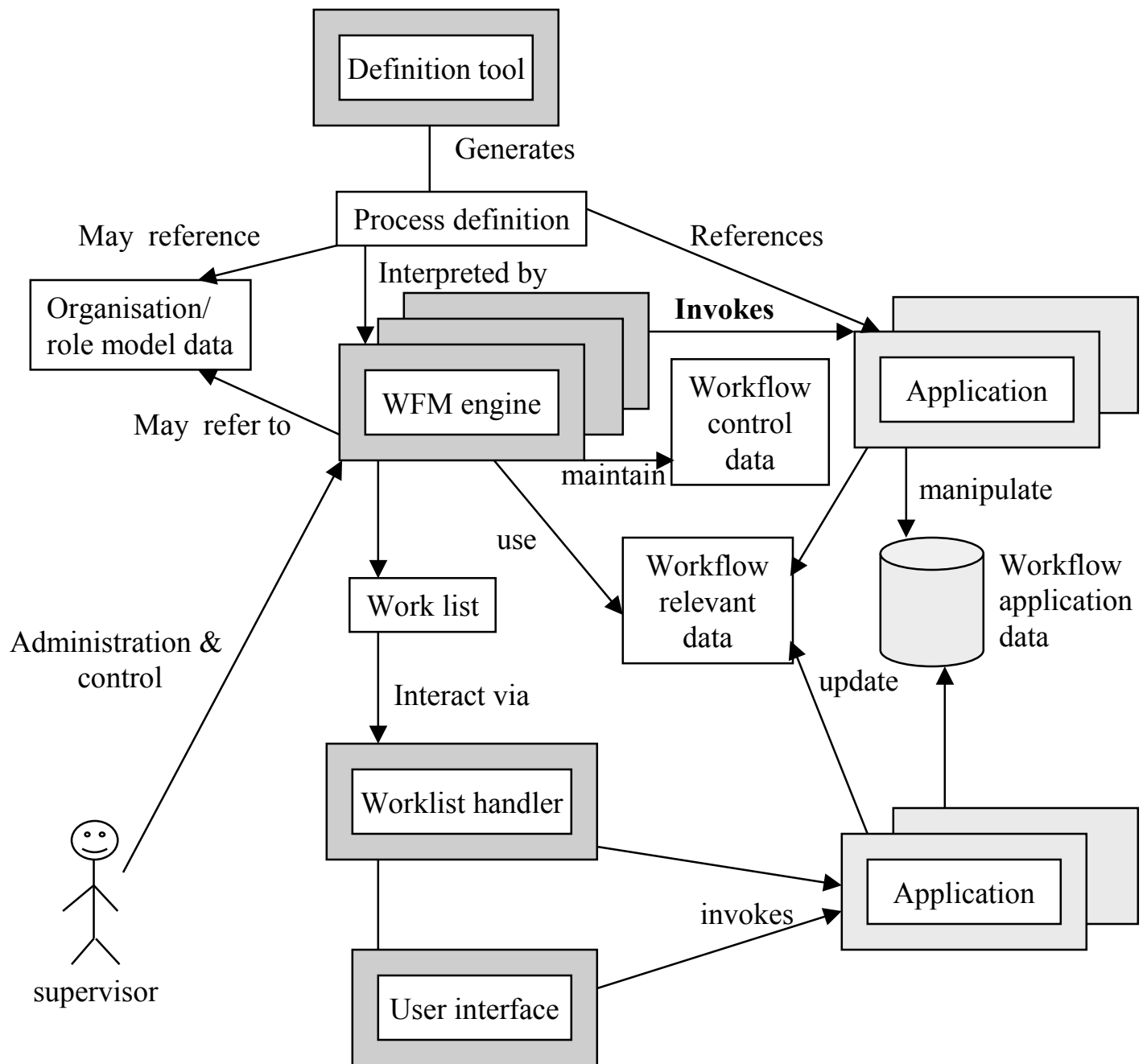
Fondé en 1993

organisation internationale non lucrative de vendeurs, utilisateurs, concepteurs et chercheurs de systèmes de workflow

sa mission : établir des standards pour la terminologie et l'interopérabilité de produits de workflow

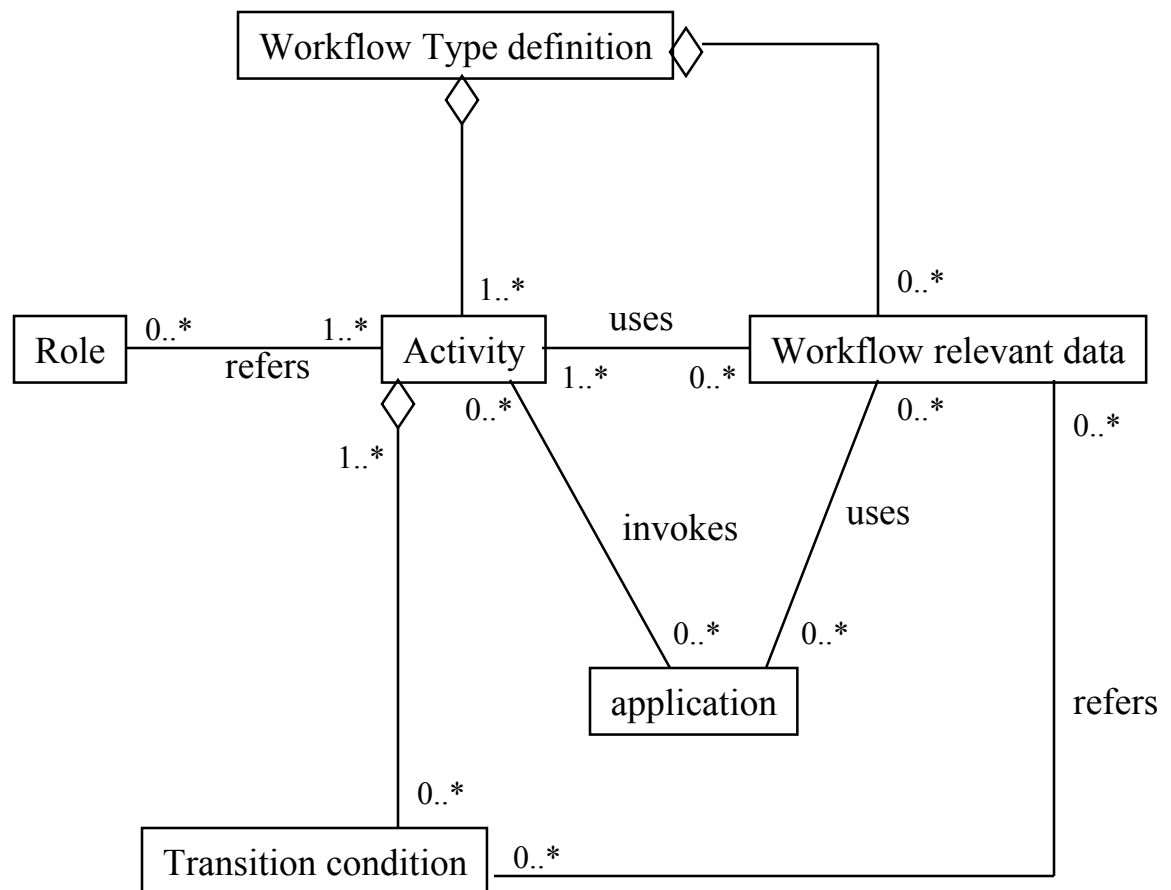
# Architecture standard d'un WFMS (workflow management system)

(extrait de "Workflow Reference Model" de Wfmc)



Workflow Enactment Service : Organisation/role model data + WFM engine + Work List + Workflow Control data + Workflow relevant data

# Basic Process Definition Meta-model



Transition condition : entry | iteration | exception

## **Le langage standard de définition de workflow basé sur XML (XPDL)**

[http://www.wfmc.org/standards/docs/TC-1025\\_10\\_xpdl\\_102502.pdf](http://www.wfmc.org/standards/docs/TC-1025_10_xpdl_102502.pdf)

Une définition de workflow est principalement composée de la définition :

- ProcessHeader
- formalParameters
- Datafields
- Participants
- Applications
- ActivitySets
- Activities
- Transitions
- AccessLevel

Les définitions de workflows peuvent être structurées en packages.

## Package (ensemble de définition d'éléments)

```
<Package Id="find-reviewer-for-paper">
  <PackageHeader>
    <XPDLVersion>0.03</XPDLVersion>
    <Vendor>Lamsade</Vendor>
    <Created>21 juillet 2003</Created>
  </PackageHeader>
  .
  .
  .
</Package>
```

## Workflow (process)

```
<WorkflowProcesses>
  <WorkflowProcess Id="1" name="find-reviewer-for-paper"
    AccessLevel="PUBLIC">
    <ProcessHeader>
      <description> processus de recherche d'un reviewer
        pour un article
      </description>
      <created> 22 juillet 2003 </created>
    </ProcessHeader>

    définition des paramètres formels du workflow
    .
    définition des données (relevant data)

    définition des participants au workflow

    définition des applications invoquées par le workflow

    définition des groupes d'activités
  </WorkflowProcess>

  <WorkflowProcess>
    .
    .
    .
  </WorkflowProcess>
</WorkflowProcesses>
```

## RedefinableHeader

```
<WorkflowProcess Id="1" name="find-reviewer-for-paper"
  AccessLevel="PUBLIC">

  <ProcessHeader>
    <description> processus de recherche d'un reviewer
    pour un article
    </description>
    <created> 22 juillet 2003 </created>
  </ProcessHeader>

  <RedefinableHeader>
    <Author>MJ. Blin</Author>
    <Version>1.0</Version>
  </RedefinableHeader>
```

## Données utilisées par le workflow pour les conditions et le passage de paramètres (relevant data)

```
<DataFields>
  <Datafield Id="1" name=nombreMinDeReviewers IsArray="False">
    <DataType>
      <BasicDataType> Type="Integer"/>
    </DataType>
    <InitialValue>2</InitialValue>
    <Length>1</Length>
  </DataField>
```

## Participants

```
<Participants>
  <Participant Id="ResponsableSession">
    <ParticipantType Type="Personne" />
    <Description>Personne du comité de programme président
    d'une session</Description>
  </Participant>

  <Participant ID="Membre">
    <ParticipantType Type="Personne" />
    <Description>membre du comité de programme
  </Description>
</Participant>
```



## Application (applications ou outils invoqués par le workflow)

```
<Application Id="pop.task">
```

définition des paramètres formels de l'application

```
    <ExternalReference>/home/workflow/pop.task</ExternalReference>
  </Application>
```

## Paramètres formels (pour un workflow ou une application)

```
<FormalParameters>
```

```
  <FormalParameter Id="accept" Index="1" Mode="OUT">
    <DataType>
      <BasicDataType type="BOOLEAN"/>
    </DataType>
  </FormalParameter>
```

```
  <FormalParameter Id="paper" Index="2" Mode="IN">
    <DataType>
      <BasicDataType type="STRING"/>
    </DataType>
  </FormalParameter>
```

```
  <FormalParameter Id="list-of-reviewers" Index="3"
Mode="INOUT">
    <DataType>
      <ListType>
        <complexType>
          <group ref="STRING" />
        </complexType>
      </ListType>
    </DataType>
  </FormalParameter>
```

```
</FormalParameters>
```

## Groupes d'activités

```
<ActivitySet Id="1">
  <Activities>
    .
    .
    .
  </Activities>
  <Transitions>
    .
    .
    .
  </Transitions>
</ActivitySet>
```

```
<Activities>
  <!-- cette activité appelle une application définie dans le workflow
  -->
  <Activity Id="activite1" name="reviewerSuivant">
    <Implementation>
      <Tool Id="pop.task" type="APPLICATION"></Tool>
      <ActualParameters>empty-list, reviewer, list-of-
        reviewers</ActualParameters>
    </Implementation>
  </Activity>

  <!-- Cette activite appelle un autre workflow défini dans le meme
  package à
  <Activity Id="activite2" name="recherche2Reviewers">
    <Implementation>
      <SubFlow Id="find-reviewer-for-paper" Execution="ASYNCHR">
        <ActualParameters>ok2, nok2, paper, rev2, list-of-
          reviewers
        </ActualParameters>
      </SubFlow>
    </Implementation>
  </Activity>

  <!-- Ces deux activites sont définies simplement pour le routage à
  <Activity Id="activite3" name="routage">
    <Implementation>
      <Route></Route>
    </Implementation>
    <TransitionRestriction>
      <Join Type="XOR">
      </Join>
    </TransitionRestriction>
  </Activity>

  <Activity Id="activite4" name="routage">
    <Implementation>
      <Route></Route>
    </Implementation>
    <TransitionRestriction>
      <Split Type="AND">
      <TransitionRefs>
        <TransitionRef>activite1</TransitionRef>
        <TransitionRef>activite2</TransitionRef>
      </TransitionRefs>
      </Split>
    </TransitionRestrictions>
  </Activity>
```

## Les transitions entre activités

<Transitions>

```
<Transition Id="T1-2-1" From="activitel" To="activite2">  
  <Condition type="Condition">empty-list=false</Condition>  
  <Description>On execute l'activite 2 apres l'activite 1 si le  
    parametre de retour de l'application pop.task appelee par  
    l'activite 1 empty-list est faux</Description>  
</Transition>
```

```
<Transition Id="T1-3" From= "activitel" To="activite3">  
  <Condition Type="Otherwise" />  
  <Description> Sinon, on passe a l'activite 4</Description>  
</Transition>
```

```
<Transition Id="T2-3" From="activite2" To="activite3">  
  <Description>On execute l'activite 3 apres l'activite 2 sans  
    condition </Description>  
</Transition>
```

```
<Transition Id=T4-1 From= "activitel" To="activite4 ">  
  <Condition Type="Condition">nombreDeBoucles<4>/Condition>  
  <Description> On boucle de l'activite 4 a l'activite 1, 3 fois, la  
    variable NombreDeBoucles est declaree en datafield</Description>  
</Transition>
```

</Transitions>

## Récapitulation

```

<Package Id="find-reviewer-for-paper">
  <PackageHeader>
    <XPDLVersion>0.03</XPDLVersion>
    <Vendor>Lamsade</Vendor>
    <Created>21 juillet 2003</Created>
  </PackageHeader>

  <WorkflowProcesses>
    <WorkflowProcess Id="1" name="find-reviewer-for-paper"
      AccessLevel="PUBLIC">
      <ProcessHeader>
        <description> processus de recherche d'un
reviewer      pour un article</description>
        <created> 22 juillet 2003 </created>
      </ProcessHeader>

      <RedefinableHeader>
        <Author>MJ. Blin</Author>
        <Version>1.0</Version>
      </RedefinableHeader>

      <DataFields>
        <Datafield Id="1" name=nombreMinDeReviewers
          IsArray="False">
          <DataType>
            <BasicDataType> Type="Integer"/>
          </DataType>
          <InitialValue>2</InitialValue>
          <Length>1</Length>
        </DataField>
      </Datafields>

    <Participants>
      <Participant Id="ResponsableSession">
        <ParticipantType Type="Personne" />
        <Description>Personne du comité de programme
          président d'une session</Description>
      </Participant>
      <Participant ID="Membre">
        <ParticipantType Type="Personne" />
        <Description>membre du comité de programme
          </Description>
      </Participant>
    </Participants>
  </WorkflowProcesses>
</Package>

```

```

    <Application Id="pop.task">

définition des paramètres formels de l'application

    <ExternalReference>/home/workflow/pop.task
    </ExternalReference>
</Application>

<FormalParameters>
    <FormalParameter Id="accept" Index="1"
Mode="OUT">
        <DataType>
        <BasicDataType type="BOOLEAN"/>
        </DataType>
    </FormalParameter>

    <FormalParameter Id="paper" Index="2" Mode="IN">
        <DataType>
            <BasicDataType type="STRING"/>
        </DataType>
    </FormalParameter>

    <FormalParameter Id="list-of-reviewers"
Index="3"
Mode="INOUT">
        <DataType>
            <ListType>
                <complexType>
                    <group ref="STRING" />
                </complexType>
            </ListType>
        </DataType>
    </FormalParameter>
</FormalParameters>

    <ActivitySet Id="1">
        <Activities>
            <!-- cette activité appelle une application
            définie dans le worflow -->
            <Activity Id="activite1"
name="reviewerSuivant">
                <Implementation>
                    <Tool Id="pop.task" type
                    ="APPLICATION"></Tool>
                <ActualParameters>empty-list, reviewer,
                    list-of-reviewers</ActualParameters>
                </Implementation>

```

```

    <!-- Cette activite appelle un autre workflow défini
dans le
meme package à
    <Activity Id="activite2" name="recherche2Reviewers">
        <Implementation>
            <SubFlow Id="find-reviewer-for-paper"
            Execution="ASYNCHR">
                <ActualParameters>ok2, nok2, paper, rev2,
list-
                    of-reviewers
                </ActualParameters>
            </SubFlow>
        </Implementation>
    </Activity>

    <!-- Ces deux activites sont définies simplement pour le
routage à
    <Activity Id="activite3" name="routage">
        <Implementation>
            <Route></Route>
        </Implementation>
        <TransitionRestriction>
        <Join Type="XOR">
        </Join>
        </TransitionRestrictions>
    </Activity>

    <Activity Id="activite4" name="routage">
        <Implementation>
            <Route></Route>
        </Implementation>
        <TransitionRestrictions>
        <Split Type="AND">
            <TransitionRefs>
                <TransitionRef>activite1</TransitionRef>
                <TransitionRef>activite2</TransitionRef>
            </TransitionRefs>
        </Split>
        </TransitionRestrictions>
    </Activity>
</Activities>

```

```
<Transitions>
```

```
  <Transition Id="T1-2-1" From="activite1"
  To="activite2">
    <Condition type="Condition">empty-
    list=false</Condition>
    <Description>On execute l'activite 2 apres
    l'activite 1 si le parametre de retour de
    l'application pop.task appelee par
    l'activite 1 empty-list est faux</Description>
  </Transition>
```

```
  <Transition Id="T1-3" From= "activite1"
  To="activite3">
    <Condition Type="Otherwise" />
    <Description> Sinon, on passe a l'activite
    4</Description>
  </Transition>
```

```
  <Transition Id="T2-3" From="activite2" To="activite3">
    <Description>On execute l'activite 3 apres
    l'activite 2 sans condition </Description>
  </Transition>
```

```
  <Transition Id=T4-1 From= "activite1" To="activite4 >
    <Condition
    Type="Condition">nombreDeBoucles<4>/Condition>
    <Description> On boucle de l'activite 4 a
    l'activite 1, 3 fois, la variable
    NombreDeBoucles est declaree en
    datafield</Description>
  </Transition>
```

```
</Transitions>
```

```
</ActivitySet>
```

```
</WorkflowProcess>
```

```
</WorkflowProcesses>
```

```
</Package>
```



# **Les qualités d'un système de gestion de workflows**

## **La flexibilité**

- la gestion des exceptions
- l'adaptation

## **L'interopérabilité**

## La gestion des exceptions

- liées aux systèmes sous-jacents
- liées aux applications associées
- celles prévues dans la définition du processus
- celles qui sont imprévues

### Leur gestion

- la détection : manuelle, gestionnaire d'exceptions, règles
- la prise en charge :
  - ✓ la ré-exécution
  - ✓ l'annulation
  - ✓ le retour en arrière
  - ✓ la compensation
  - ✓ la délégation
  - ✓ la relaxation des contraintes
  - ✓ l'ignorance

## L'adaptation

plus ou moins complexe selon qu'elle concerne :

une instance particulière du processus  
toutes les instances en cours  
le modèle

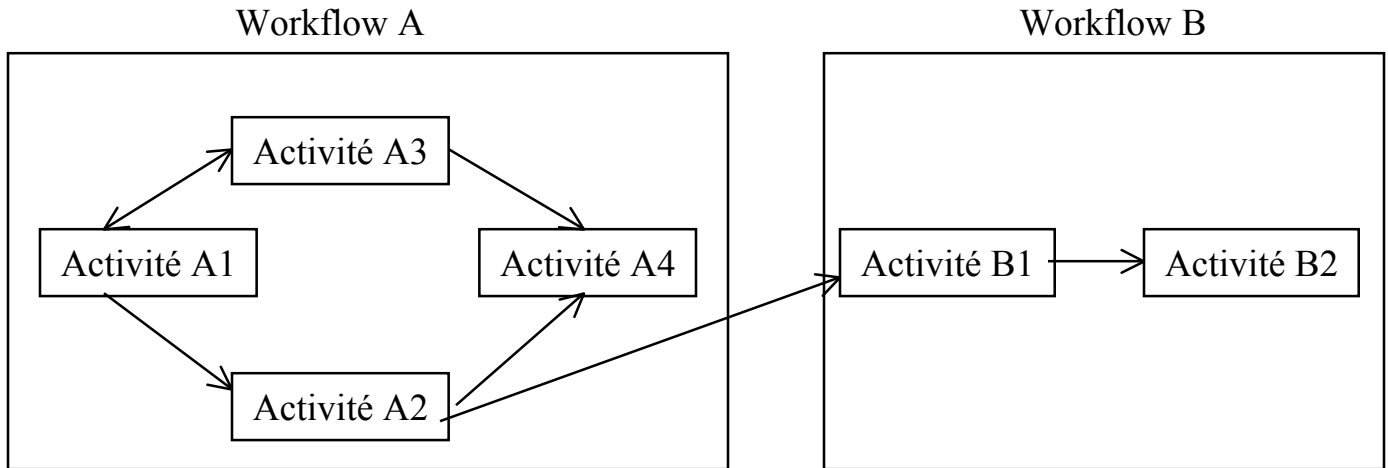
### Les techniques :

la sélection dynamique manuelle d'un comportement  
l'attribution automatique de comportement  
l'affinage dynamique et la modélisation tardive (permet de modéliser en cours d'exécution des parties manquantes)

## L'interopérabilité entre workflows

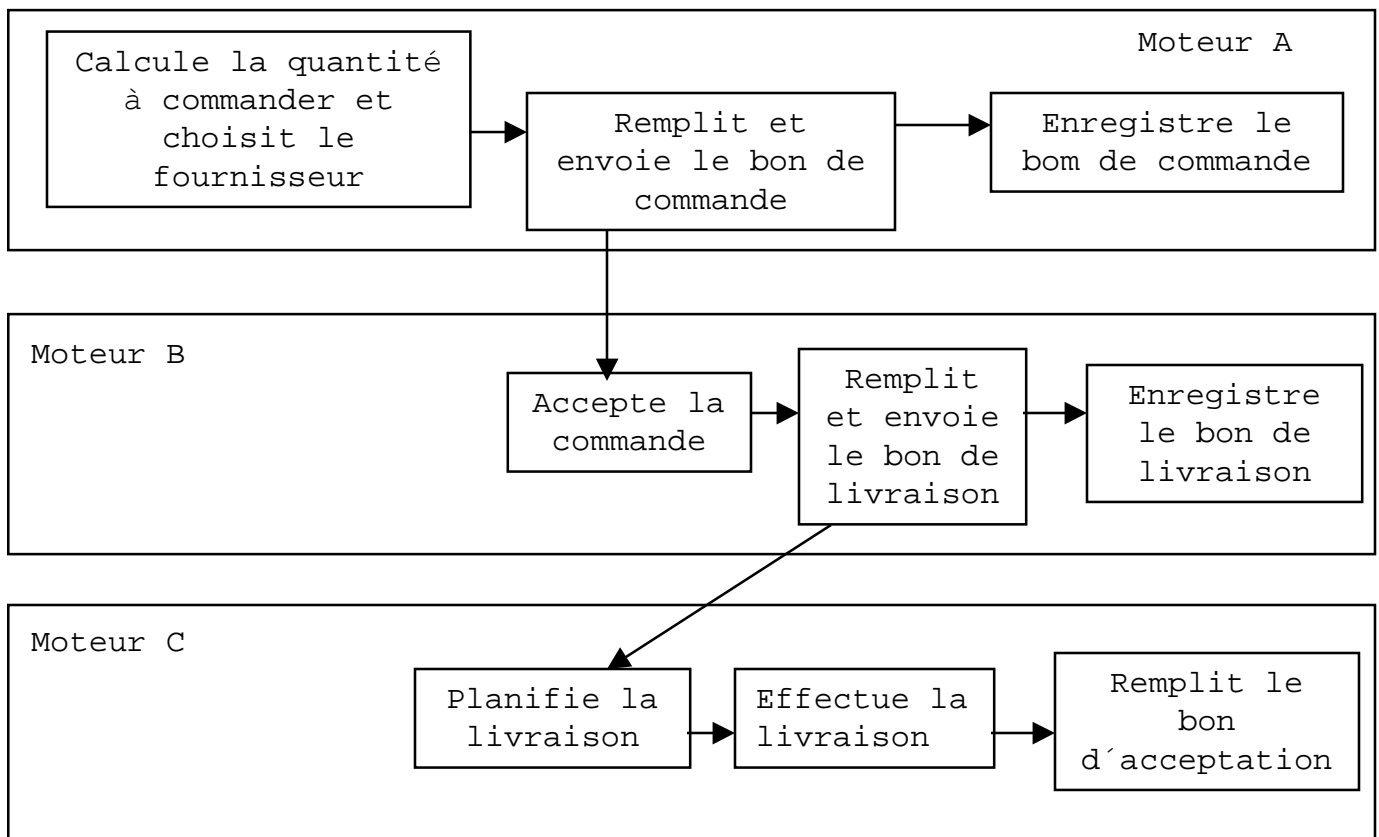
### Modèles et standards du WFMC

#### Workflows chaînés

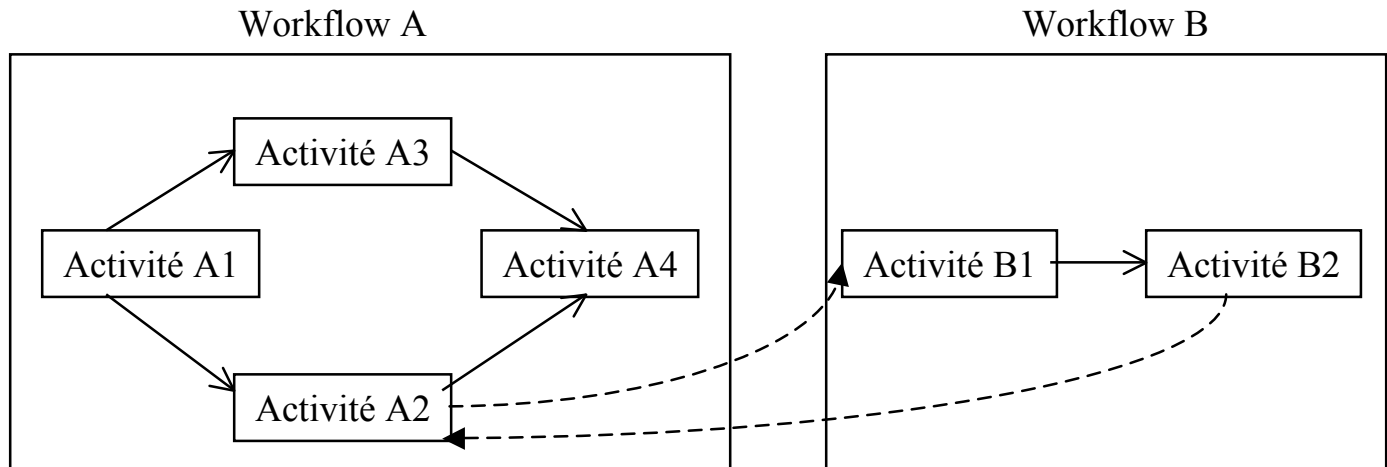


Une fois le workflow B lancé, les deux workflows s'exécutent en parallèle

#### Exemple

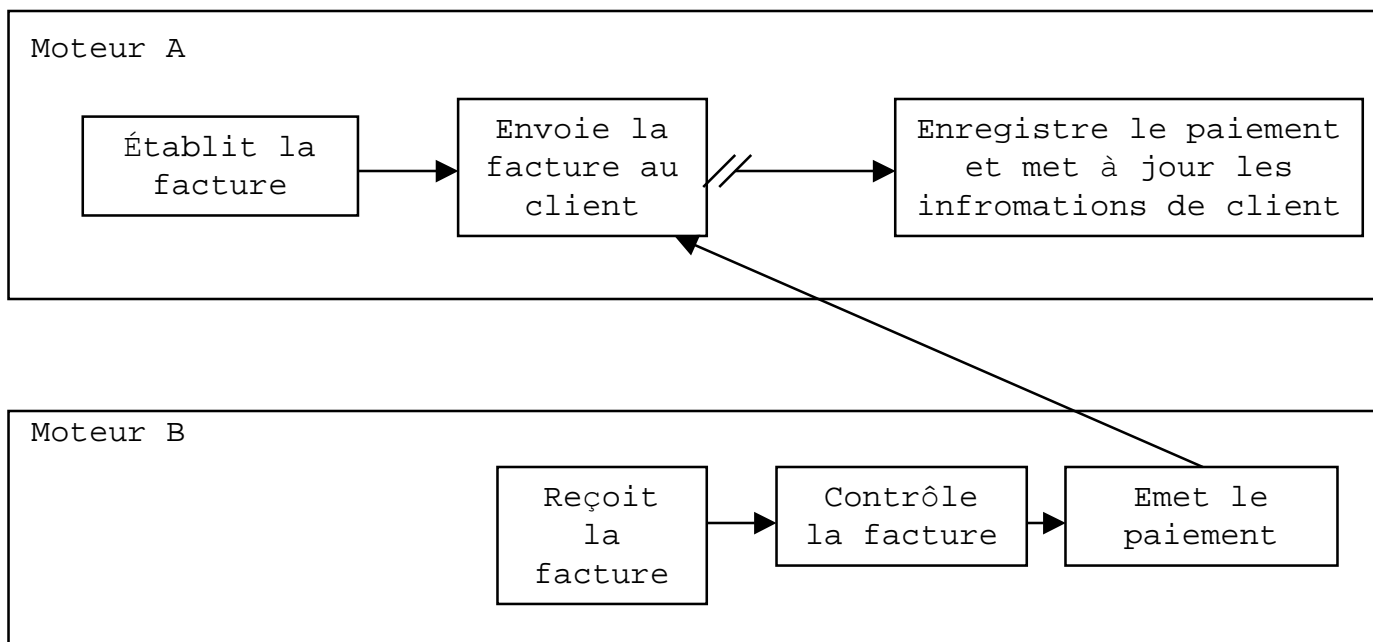


## Workflows imbriqués

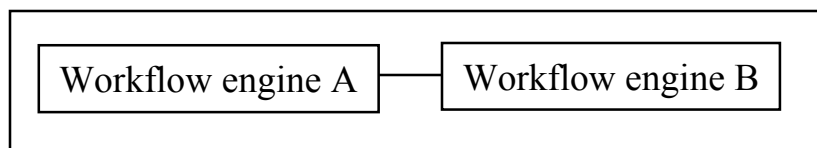
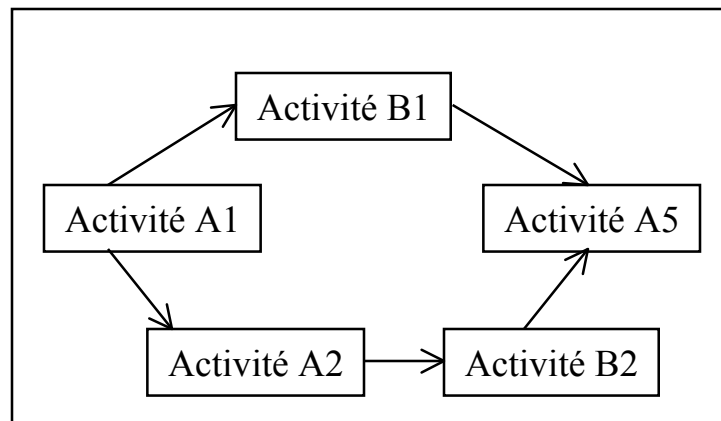


L'activité A2 transfère le contrôle au workflow B. Quand l'exécution de ce dernier est terminée, l'activité A2 est finie aussi.

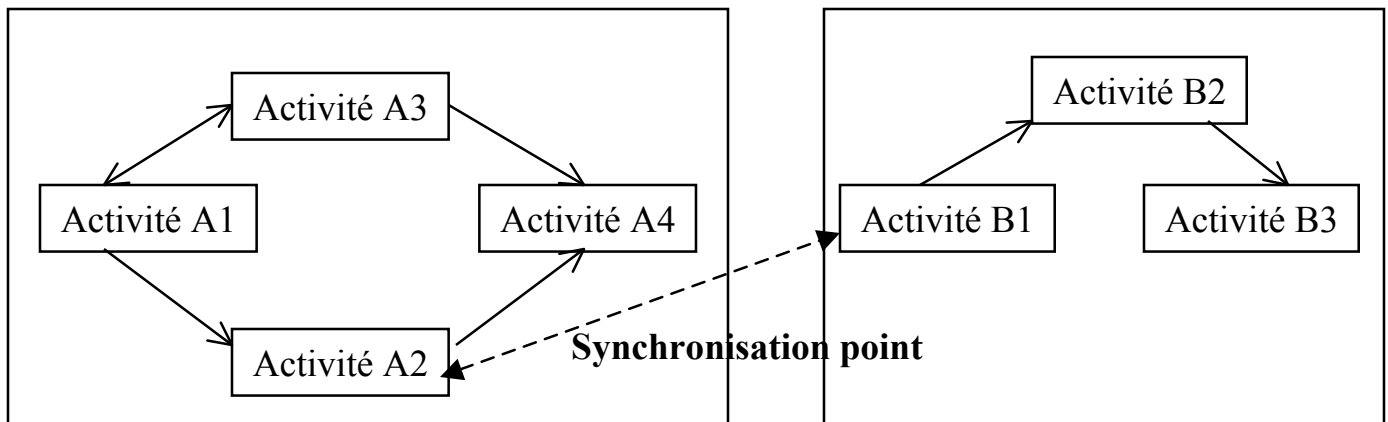
## Exemple



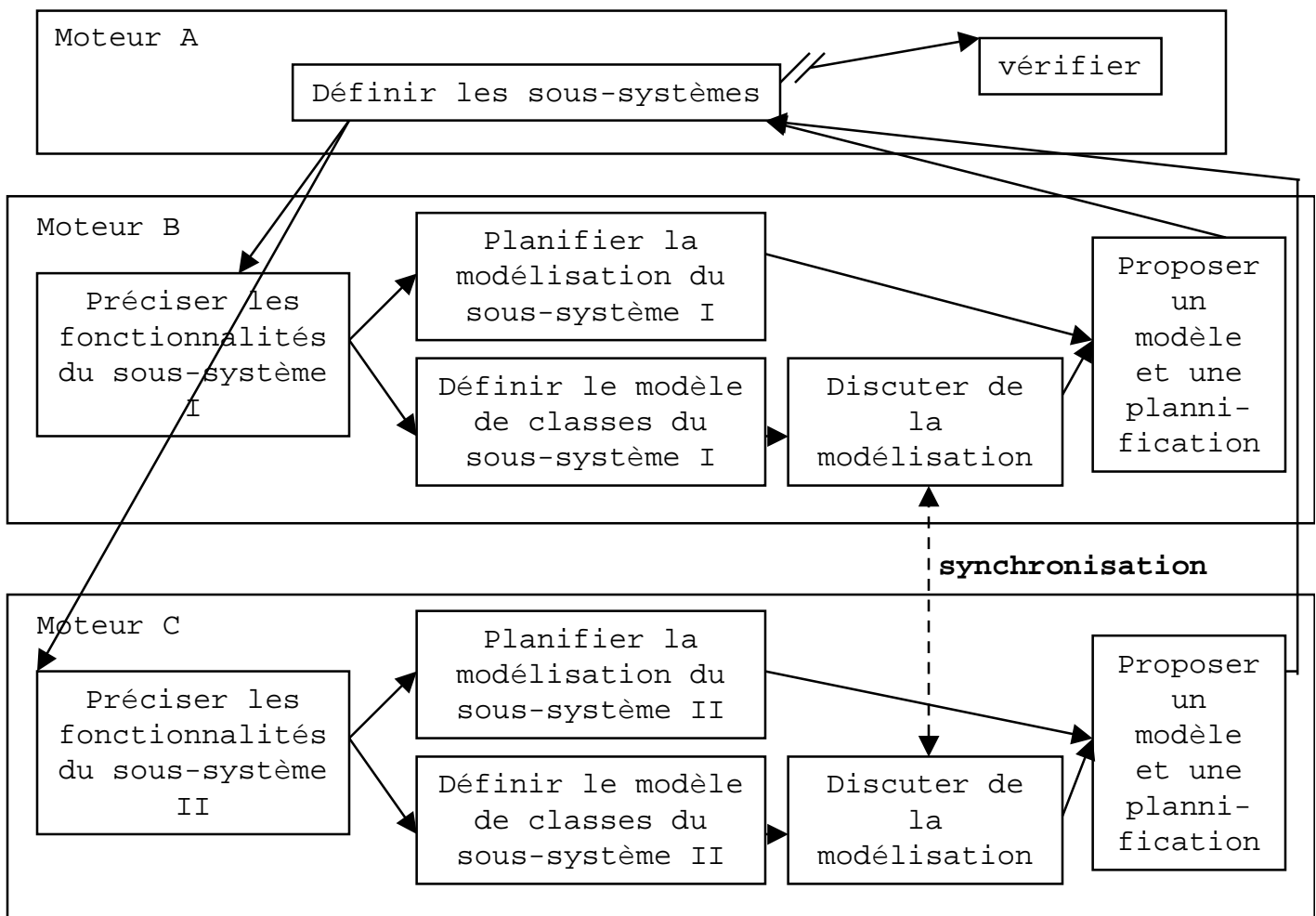
## Workflows connectés



## Workflows synchronisés



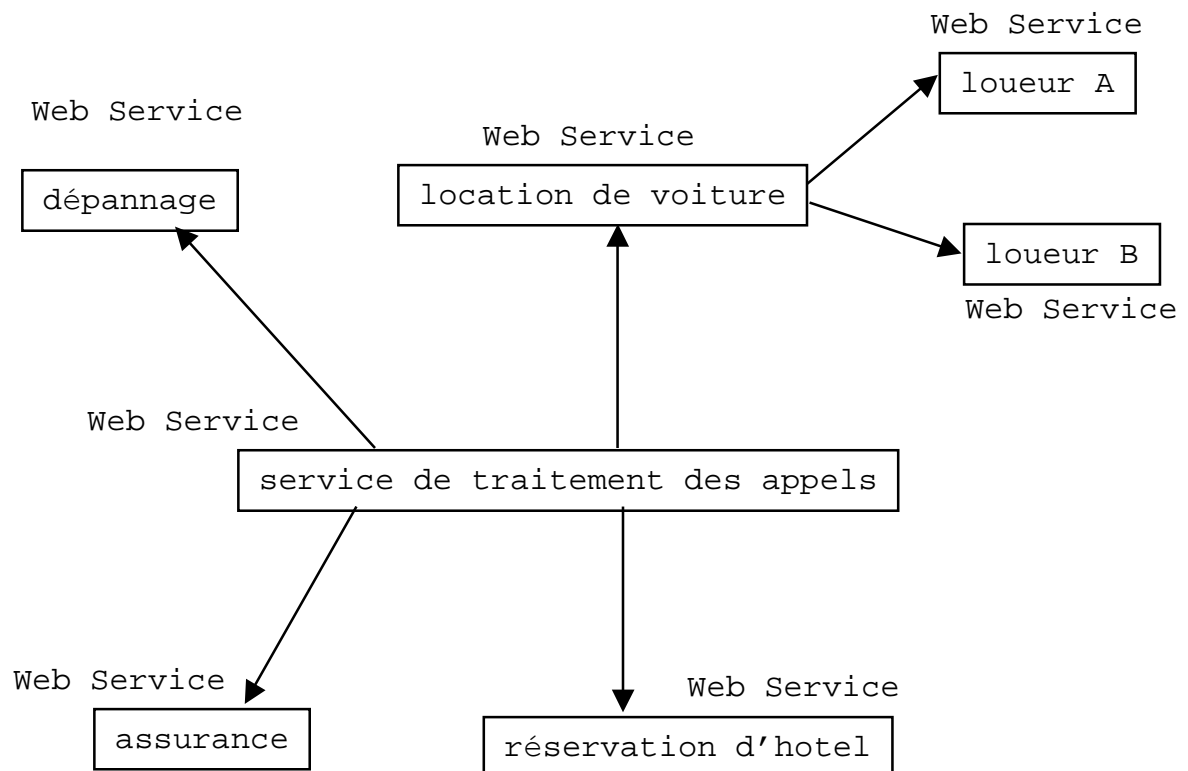
## Exemple



# Les Web services

utiliser le web comme infrastructure pour les systèmes de gestion de workflows distribués et hétérogènes

exemple :



Un Web service = une application modulaire basée sur les protocoles Internet, qui fournit un service spécifique, qui respecte le format d'échange de données XML

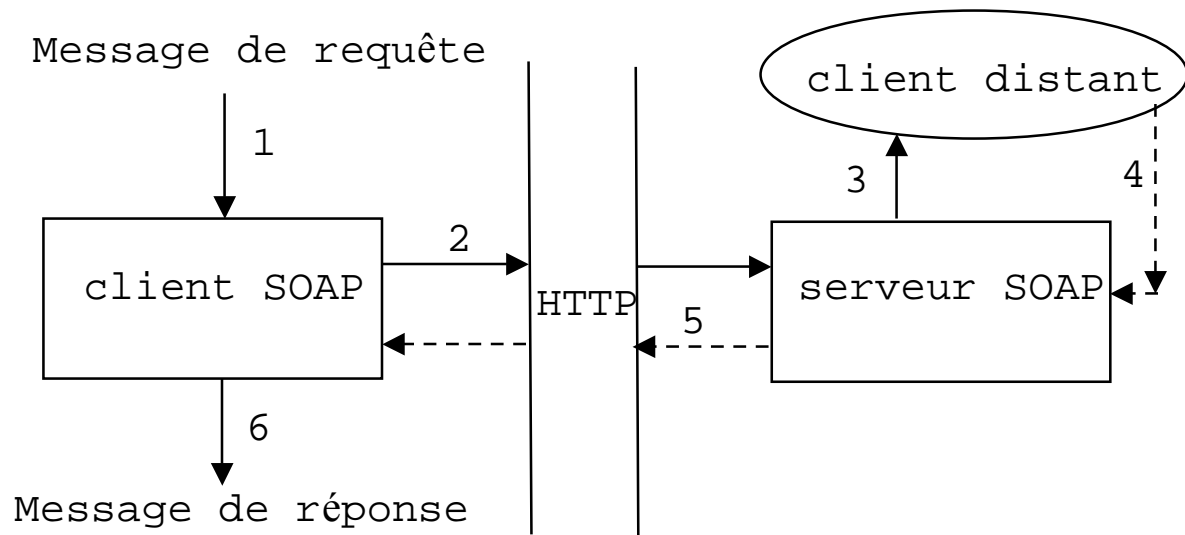
## Outils permettant de construire des services Web et des compositions de services

flots de service	WSFL, XLANG (langage de (définition de processus)
publication, découverte	UDDI (Universal Description, Discovery and Integration)
description de services	WSDL (Web service description language)
messages XML	SOAP(échange de messages XML au-dessus de HTTP)
réseau	HTTP, FTP, IIOP

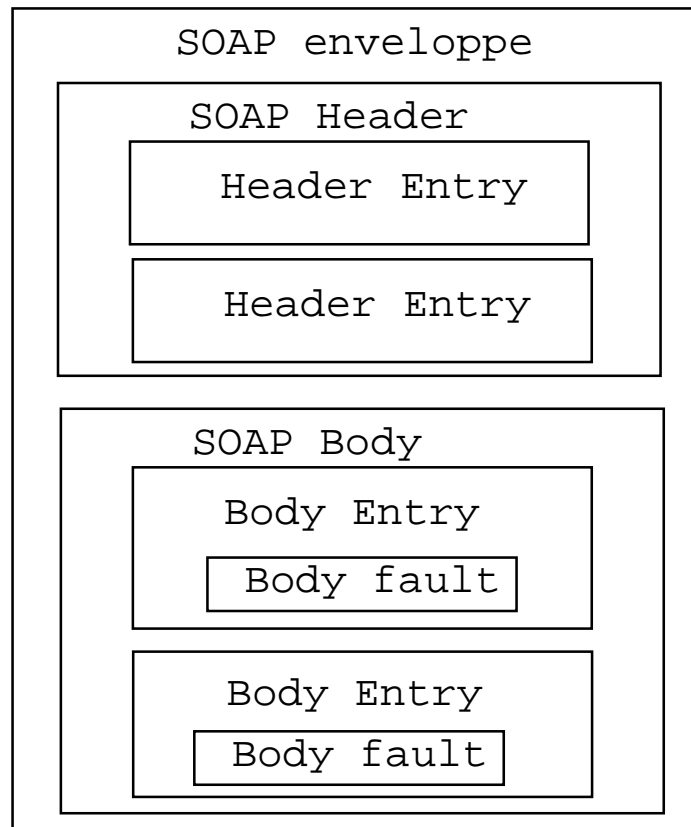


## SOAP (Simple Object Access Protocol)

Standard W3C de définition d'un protocole assurant des appels de procédures à distance (RPC) s'appuyant principalement sur XML et HTTP



## Les messages SOAP



```

<SOAP-ENV: Envelope
  xmlns: SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <SOAP-ENV:Header>
        <t:Transaction >
          SOAP-ENV:mustUnderstand="1"
        </t:Transaction>
      </SOAP-ENV:Header>
      <SOAP-ENV:Body>
        <m:ValiderCodePostal>
          <Code>75775</Code>
        </m:ValiderCodePostal>
      </SOAP-ENV:Body>
    </SOAP-ENV: Envelope>
  
```

## Réponses

```
<SOAP-ENV: Envelope
  xmlns: SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:
    encodingStyle==http://schemas.xmlsoap.org/soap/encoding/>
      <SOAP-ENV:Body>
        <m:ValiderCodePostalReponse >
          <Valide>Oui</Valide>
        </m:ValiderCodePostalReponse>
      </SOAP-ENV:Body>
</SOAP-ENV: Envelope>
```

**ou**

```
<SOAP-ENV: Envelope
  xmlns: SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:
    encodingStyle==http://schemas.xmlsoap.org/soap/encoding/>
      <SOAP-ENV:Header>
        <f:NonCompris qname = "t:Transaction"/>
      </SOAP-ENV:Header>
      <SOAP-ENV:Body>
        <SOAP-ENV:Fault>
          <fauteCode> Doit etre compris </fauteCode>
          <fauteTexte> Un header obligatoire n'est pas compris </fauteTexte>
        </SOAP-ENV:Fault>
      </SOAP-ENV:Body>
</SOAP-ENV: Envelope>
```

## Exemple d'une requête SOAP sur HTTP

POST /Application HTTP/1.1

Host : <http://www.HoteURI.com>

Content-Type: text/xml; charset="utf-8"

Content-Lenght: nnnn

SOAPAction : "CodePostal-service"

<SOAP-ENV: Envelope

xmlns: SOAP-ENV="<http://schemas.xmlsoap.org/soap/envelope/>"

SOAP-ENV:

encodingStyle==<http://schemas.xmlsoap.org/soap/encoding/>>

<SOAP-ENV:Header>

<t:Transaction>

SOAP-ENV:mustUnderstand="1"

</t:Transaction>

</SOAP-ENV:Header>

<SOAP-ENV:Body>

<m:ValiderCodePostal >

<Code>75775<Code>

</m:ValiderCodePostal>

</SOAP-ENV:Body>

<SOAP-ENV: Envelope>

## Réponse sur HTTP

HTTP/1.1 200 OK

Content-Type: text/xml; charset="utf-8"

Content-Lenght: nnnn

<SOAP-ENV: Envelope

xmlns: SOAP-ENV="<http://schemas.xmlsoap.org/soap/envelope/>"

SOAP-ENV:

encodingStyle==<http://schemas.xmlsoap.org/soap/encoding/>>

<SOAP-ENV:Body>

<m:ValiderCodePostalReponse>

<Valide>Oui</Valide>

</m:ValiderCodePostalReponse>

</SOAP-ENV:Body>

<SOAP-ENV: Envelope>

## Requête SOAP contenant des données non xml

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/

  <SOAP:Header>
    <n:Manifest xmlns:n=http://example.org/manifest>
      <n:Reference n:id="image1" xlink:href="fichierImage1">
        <n:Description>Photo numero 1</n:Description>
      </n:Reference>
    </n:Manifest>
  </SOAP:Header>

  <SOAP:Body>
    .....
  </SOAP:Body>
</SOAP-ENV>
```

## WSDL (Web Services Description Language)

Definitions

Types

Messages

Port Types

Binding

Service

## Definitions

```
<definitions name = "CodePostal"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd=http://www.w3.org/2000/10/XMLSchema/>
```

## Types

```
<types>
  <schema xmlns="http://www.w3.org/2000/10/XMLSchema/">
    <xsd:element name="Code" type="xsd:decimal">
    <xsd:element name="Valide" type="xsd:string">
  </schema>
</types>
```

## Messages

```
<message name = "ValiderCodePostal">
  <part name="body" type="xsd:decimal"/>
</message>

<message name = "ValiderCodePostalReponse">
  <part name="body" type="xsd:string"/>
</message>
```

## Port types

```
<portType name = "CodePostal_ServiceType">
  <operation name "CodePostal">
    <input message="ValiderCodePostal"/>
    <output message="ValiderCodePostanReponse"/>
  </operation>
</portType>
```

## Binding

```
<binding name="ValiderCodePostal_ServiceBinding"
  type="CodePostal_ServiceType">
  <soap:binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="CodePostal">
    <soap:operation soapAction="CodePostal-Service"/>
    <input>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        use="literal"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        use="literal"/>
    </output>
  </operation>
</binding>
```

## Service

```
<service name="CodePostalService">
  <documentation>définition du Web Service validant un code postal
</documentation>
  <port name="CodePostalPort"
    binding="ValiderCodePostal_ServiceBinding">
    <soap:address
      location="http://webServices.dauphine.fr/CodePostal"/>
    </port>
  </service>
```



```
<definitions name = "CodePostal" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns=http://schemas.xmlsoap.org/wsdl/ xmlns:xsd=http://www.w3.org/2000/10/XMLSchema>

  <types>
    <schema xmlns="http://www.w3.org/2000/10/XMLSchema">
      <xsd:element name="Code" type="xsd:decimal">
      <xsd:element name="Valide" type="xsd:string">
    </schema>
  </types>

  <message name="ValiderCodePostal">
    <part name="body" type="xsd:decimal"/>
  </message>
  <message name="ValiderCodePostalReponse">
    <part name="body" type="xsd:string"/>
  </message>

  <portType name="CodePostal_ServiceType">
    <operation name "CodePostal">
      <input message="ValiderCodePostal"/>
      <output message="ValiderCodePostanReponse"/>
    </operation>
  </portType>

  <binding name="ValiderCodePostal_ServiceBinding" type="CodePostal_ServiceType">
    <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="CodePostal">
      <soap:operation soapAction="CodePostal-Service"/>
      <input>
        <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          use="encoded"/>
      </input>
      <output>
        <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          use="encoded"/>
      </output>
    </operation>
  </binding>

  <service name="CodePostalService">
    <documentation>définition du Web Service validant un code postal</documentation>
    <port name="CodePostalPort" binding="ValiderCodePostal_ServiceBinding">
      <soap:address location="http://webServices.dauphine.fr/CodePostal/"/>
    </port>
  </service>
</definitions>
```

## Utilisation de l'API JAVA de génération d'une description WSDL

```
/* l'API est WSDL4J à télécharger à partir du site d'IBM "WSDL4J-Bin-1.4" */

import java.util.*;
import java.io.*;
import javax.xml.namespace.QName;
import javax.xml.parsers.*;
import javax.wsdl.WSDLException;
import javax.wsdl.extensions.UnknownExtensibilityElement;
import org.w3c.dom.*;
import com.ibm.wsdl.*;
import com.ibm.wsdl.factory.WSDLFactoryImpl;
import com.ibm.wsdl.extensions.soap.*;

public class GenerationWSDL {
public static void main (String [] args) throws javax.wsdl.WSDLException
{
    javax.wsdl.factory.WSDLFactory factory=
    javax.wsdl.factory.WSDLFactory.newInstance();
    javax.wsdl.Definition def=factory.newDefinition ();
    String xsd="http://www.w3c.org/2001/XMLSchema";
    javax.wsdl.Part part1=def.createPart();
    javax.wsdl.Part part2=def.createPart();
    javax.wsdl.Message msg1=def.createMessage();
    javax.wsdl.Message msg2=def.createMessage();
    javax.wsdl.Input input=def.createInput();
    javax.wsdl.Output output=def.createOutput();
    javax.wsdl.Operation operation=def.createOperation();
    javax.wsdl.PortType portType=def.createPortType();

    def.setQName(new QName("CodePostal"));
    def.addNamespace("xsd", xsd);

    part1.setName("body");
    part1.setTypeName(new javax.xml.namespace.QName(xsd, "decimal"));
    msg1.setQName (new javax.xml.namespace.QName("ValiderCodePostal"));
    msg1.addPart(part1);
    msg1.setUndefined(false);
    def.addMessage(msg1);
}
```

```
part2.setName("body");
part2.setTypeName(new javax.xml.namespace.QName(xsd, "string"));
msg2.setQName (new
    javax.xml.namespace.QName("ValiderCodePostalReponse"));
msg2.addPart(part2);
msg2.setUndefined(false);
def.addMessage(msg2);

input.setMessage(msg1);
output.setMessage(msg2);
operation.setName("CodePostal");
operation.setInput(input);
operation.setOutput(output);
operation.setUndefined(false);
portType.setQName(new
    javax.xml.namespace.QName("CodePostal_ServiceType"));
portType.addOperation(operation);
portType.setUndefined(false);
def.addPortType(portType);

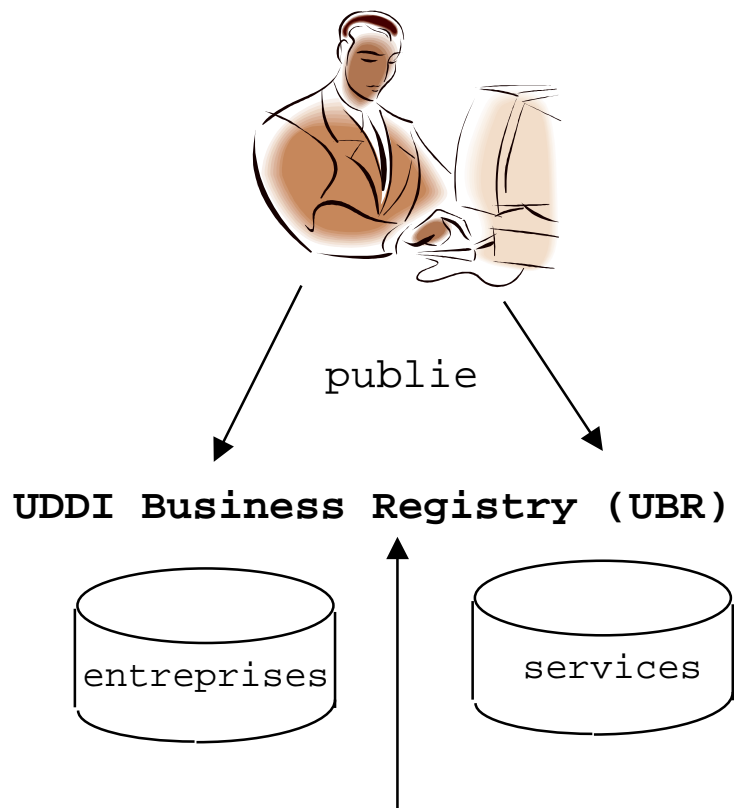
javax.wsdl.xml.WSDLWriter writer=factory.newWSDLWriter();
writer.writeWSDL(def, System.out);
} // fin du main
} // fin de la classe
```

## Résultats

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="CodePostal"
    xmlns:xsd="http://www.w3c.org/2001/XMLSchema"
    xmlns="http://schemas.xmlsoap.org/wsdl/">
    <message name="ValiderCodePostalReponse">
        <part name="body" type="xsd:string"/>
    </message>
    <message name="ValiderCodePostal">
        <part name="body" type="xsd:decimal"/>
    </message>
    <portType name="CodePostal_ServiceType">
        <operation name="CodePostal">
            <input message="ValiderCodePostal"/>
            <output message="ValiderCodePostalReponse"/>
        </operation>
    </portType>
</definitions>
```

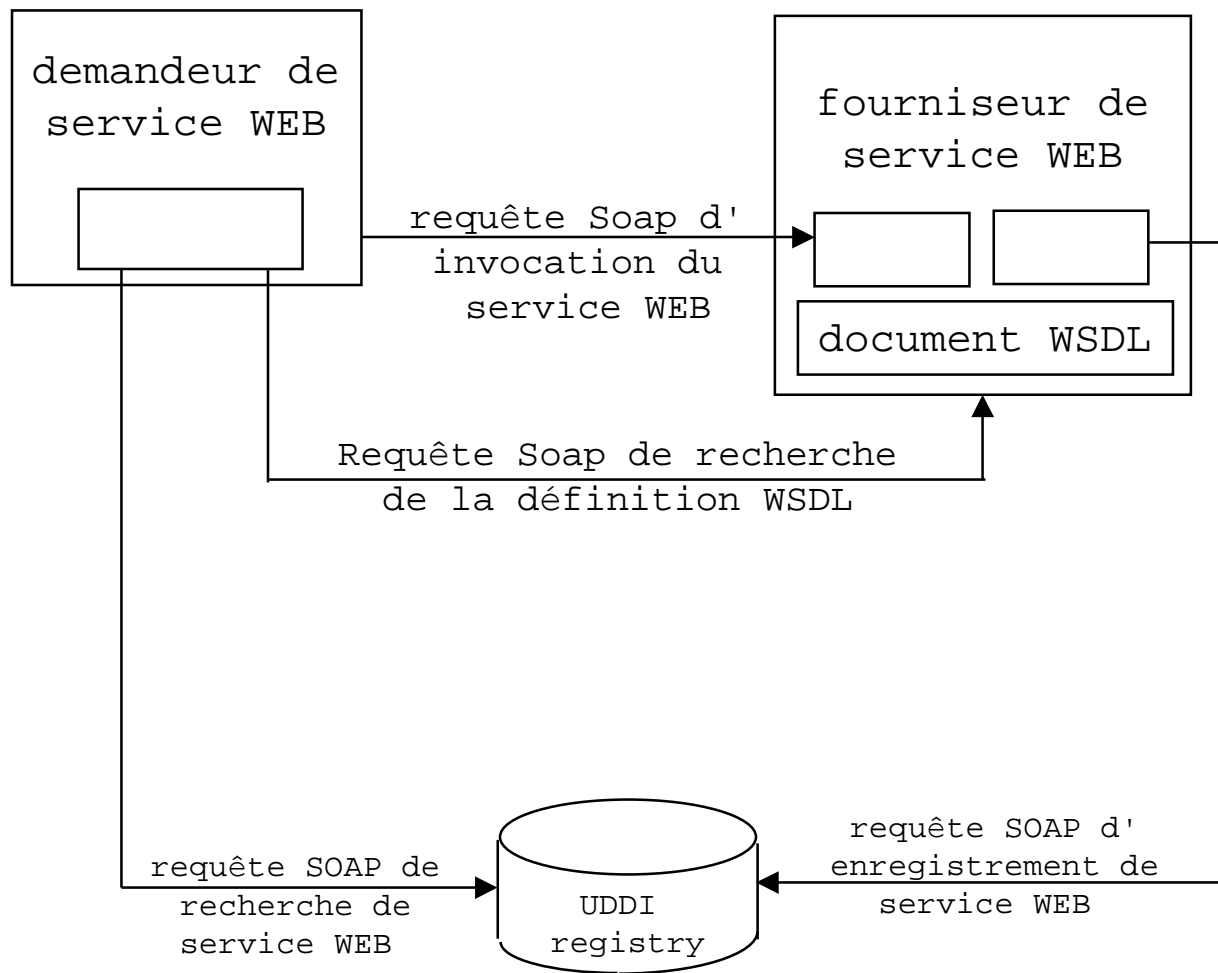
## UDDI (Universal Description Discovery and Integration)

Moyen de publier et de rechercher des informations sur des entreprises et des services



interrogent le UBR pour découvrir des entreprises ou/et des services



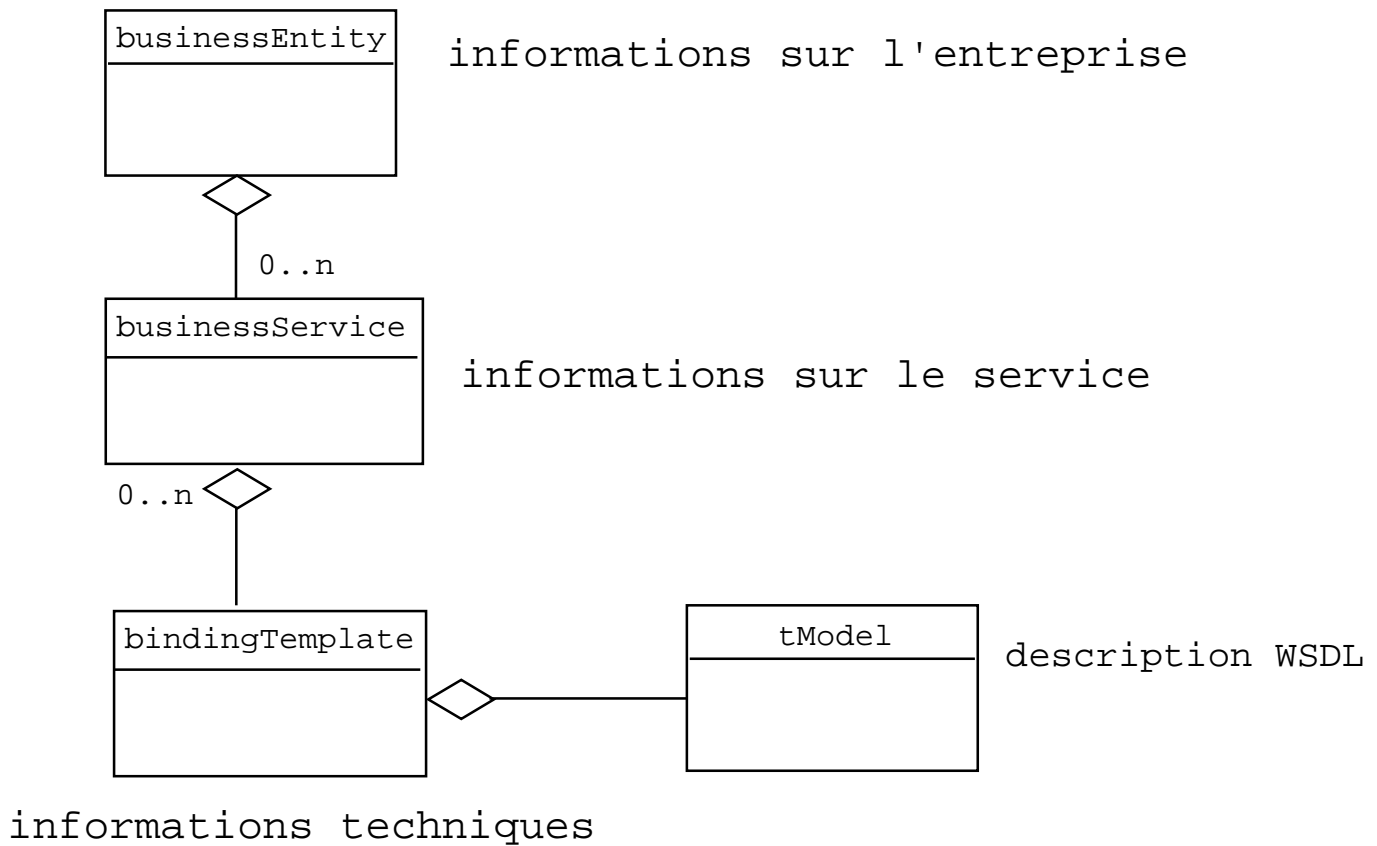


### Annuaire UDDI :

- pages blanches : informations sur les entreprises
- pages jaunes : description WSDL des services WEB
- pages vertes : fournit de informations techniques détaillés sur les services

## Modèle de données UDDI

Modèle XML comportant 5 structures de données



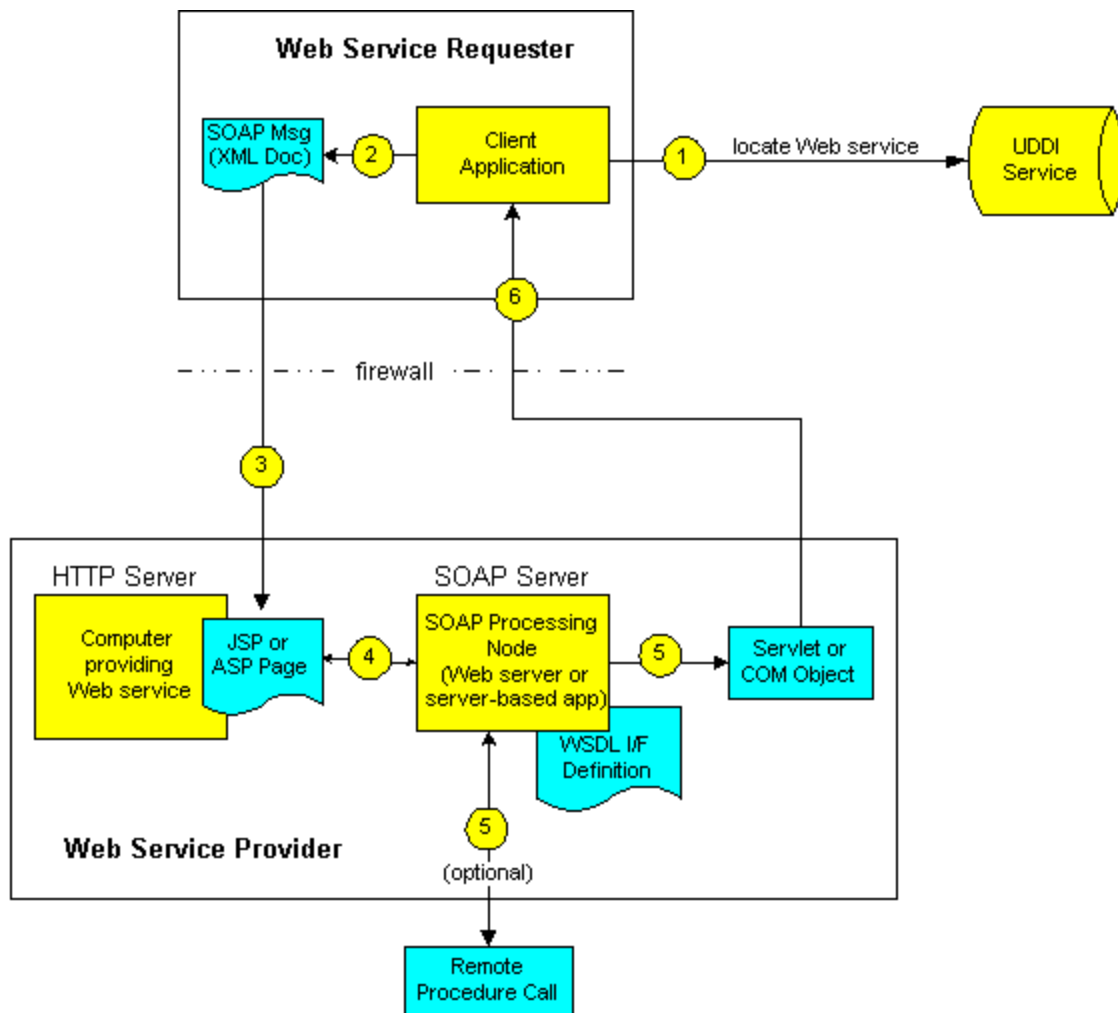
### API d'interrogation du UDDI registry

messages XML encapsulés dans des enveloppes SOAP permettant d'interroger les données de l'annuaire

### API de publication dans l'annuaire

pour supprimer ou ajouter des données

## Utilisation d'un service Web



## WSFL (Web Services Flow Language)

description d'un processus comme une succession d'appels à des opérations de services WEB décrits en WSDL

spécification de la circulation des données d'un appel à l'autre

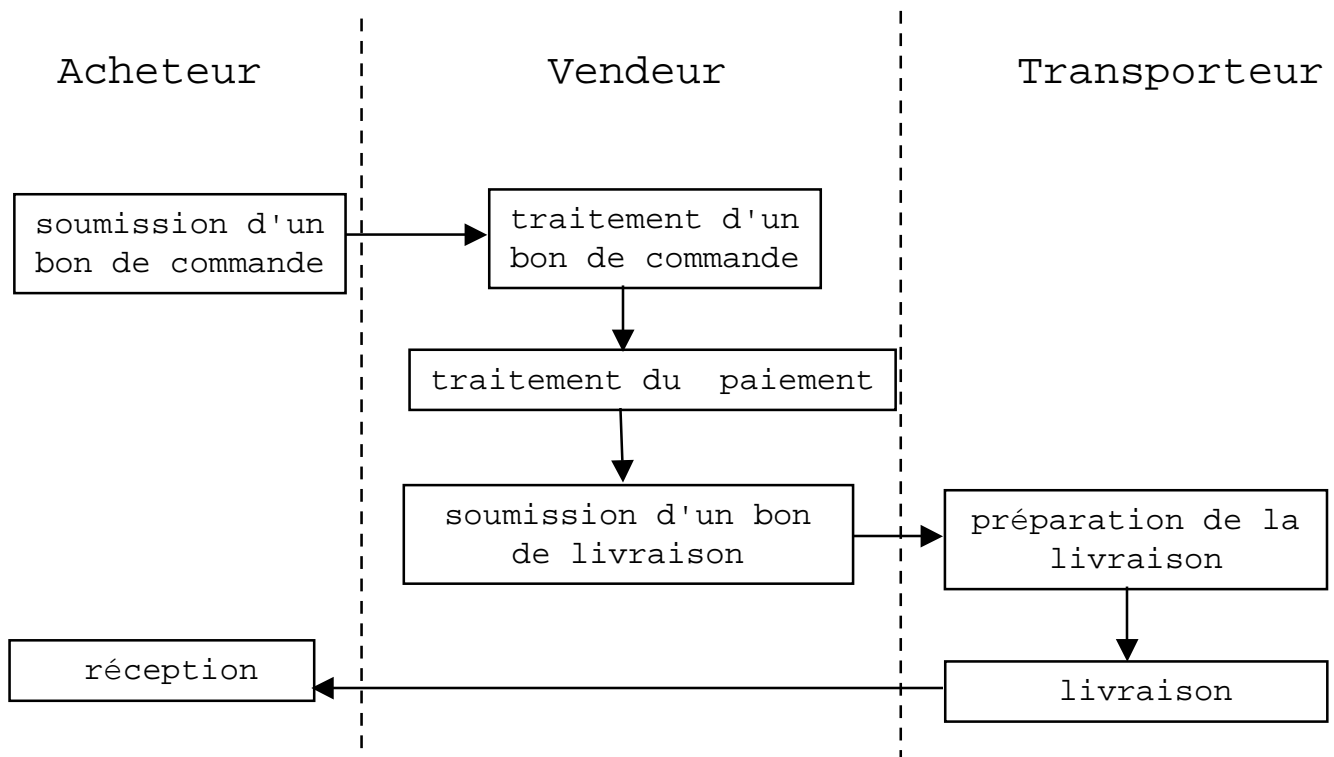
existence de primitives de contrôle (boucles, tests,...)

### deux styles de composition de services :

modélisation de processus métier

modélisation d'interaction de services pris deux à deux (contrat)

### Modélisation d'un processus métier en WSFL





```
<flowModel name="CommandeFlow" serviceProviderType="Commande">
```

```
<--! Déclaration des fournisseurs de services utilisés et leurs  
adresses -->
```

```
    <serviceProvider name="monVendeur" type="vendeur">  
        <locator type="static" service="Fournir.com"/>  
    </serviceProvider>  
    <serviceProvider name="monTransporteur" type="transporteur">  
        <locator type="static" service="Transport.com"/>  
    </serviceProvider>
```

```
<--! définitions des activités-->
```

```
<--! 1ere activité exécutée par l'opération "envoiOrdreTraitement"  
du service "TraiterCommandePT" du fournisseur de services  
"monVendeur"-->
```

```
<activity name "accepterCommande">  
    <performedBy serviceProvider="monVendeur"/>  
    <implement>  
        <export>  
            <target portType="TraiterCommandePT" operation =  
                "envoiOrdreTraitement"/>  
        </export>  
    </implement>  
</activity>
```

```
<--! 2eme activité exécutée par l'opération "envoiOrdreTransport" du  
service "TraiterCommandePT" du fournisseur de services  
"monTransporteur"-->
```

```
<activity name "traiterBonLivraison">  
    <performedBy serviceProvider="monTransporteur"/>  
    <implement>  
        <export>  
            <target portType="TraiterCommandePT" operation =  
                "envoiOrdreTransport"/>  
        </export>  
    </implement>  
</activity>
```

```
<--! 3eme activité exécutée par l'opération "envoiPaielement" du
service "TraiterCommandePT" du fournisseur de services
"monVendeur" -->
```

```
<activity name "traiterPaielement">
  <performedBy serviceProvider="monVendeur"/>
  <implement>
    <export>
      <target portType="TraiterCommandePT" operation =
        "envoiPaielement"/>
    </export>
  </implement>
</activity>
```

```
<--!définitions des contraintes de séquence dans l'exécution de
activités -->
```

```
<--!la 2ème activité ne peut s'exécuter qu'après la 1ère activité -->
```

```
<controlLink source="accepterCommande" target="traiterBonLivraison"/>
```

```
<--!définitions du flow de données entre les activités -->
```

```
<dataLink source="accepterCommande" target="traiterBonLivraison">
  <map sourceMessage="bonLivraison" targetMessage="BL"/>
</dataLink>
```

```
</flowModel>
```

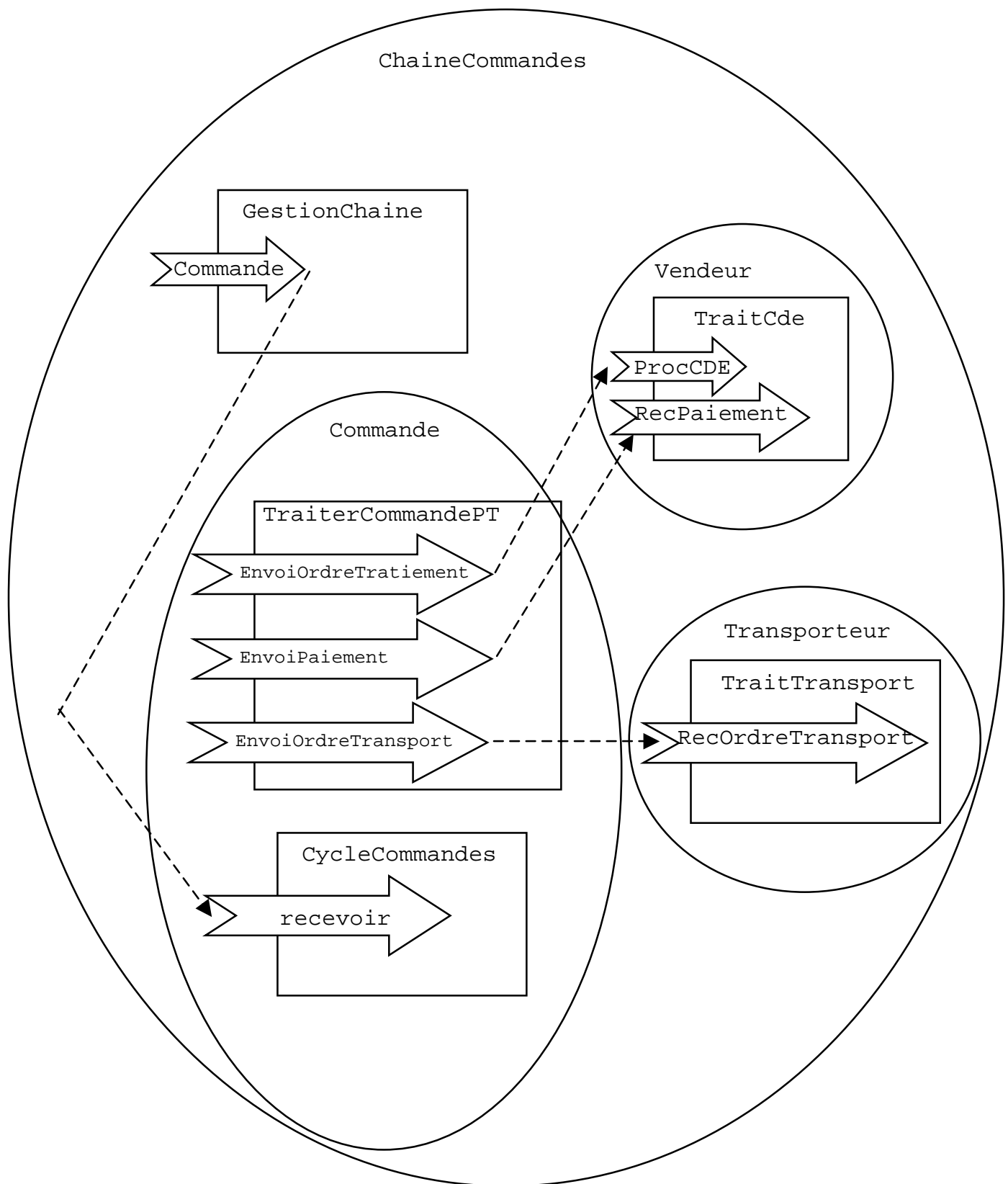
```
<globalModel name "maChaineCommande"
  serviceProviderType="ChaineCommandes">
  <serviceProvider name ="monVendeur" type="Vendeur"/>
  <serviceProvider name ="monTransporteur" type="Transporteur"/>
  <serviceProvider name="maCommande" type="Commande">
    <export>
      <source portType="CycleCommandes" operation="recevoir"/>
      <target portType="GestionChaine" operation="Commande"/>
    </export>
  </serviceProvider>

  <plugLink>
    <source serviceProvider="maCommande" portType="TraiterCommandePT"
      operation="envoiOrdreTraitement"/>
    <target serviceProvider="monVendeur" portType="TraitCde"
      operation="ProcCde"/>
  </plugLink>

  <plugLink>
    <source serviceProvider="maCommande" portType="TraiterCommandePT"
      operation="envoiPaieement"/>
    <target serviceProvider="monVendeur" portType="TraitCde"
      operation="RecPaieement"/>
  </plugLink>

  <plugLink>
    <source serviceProvider="maCommande" portType="TraiterCommandePT"
      operation="envoiOrdreTransport"/>
    <target serviceProvider="monTransporteur" portType="TraitTransport"
      operation="RecOrdreTransport"/>
  </plugLink>

</globalModel>
```

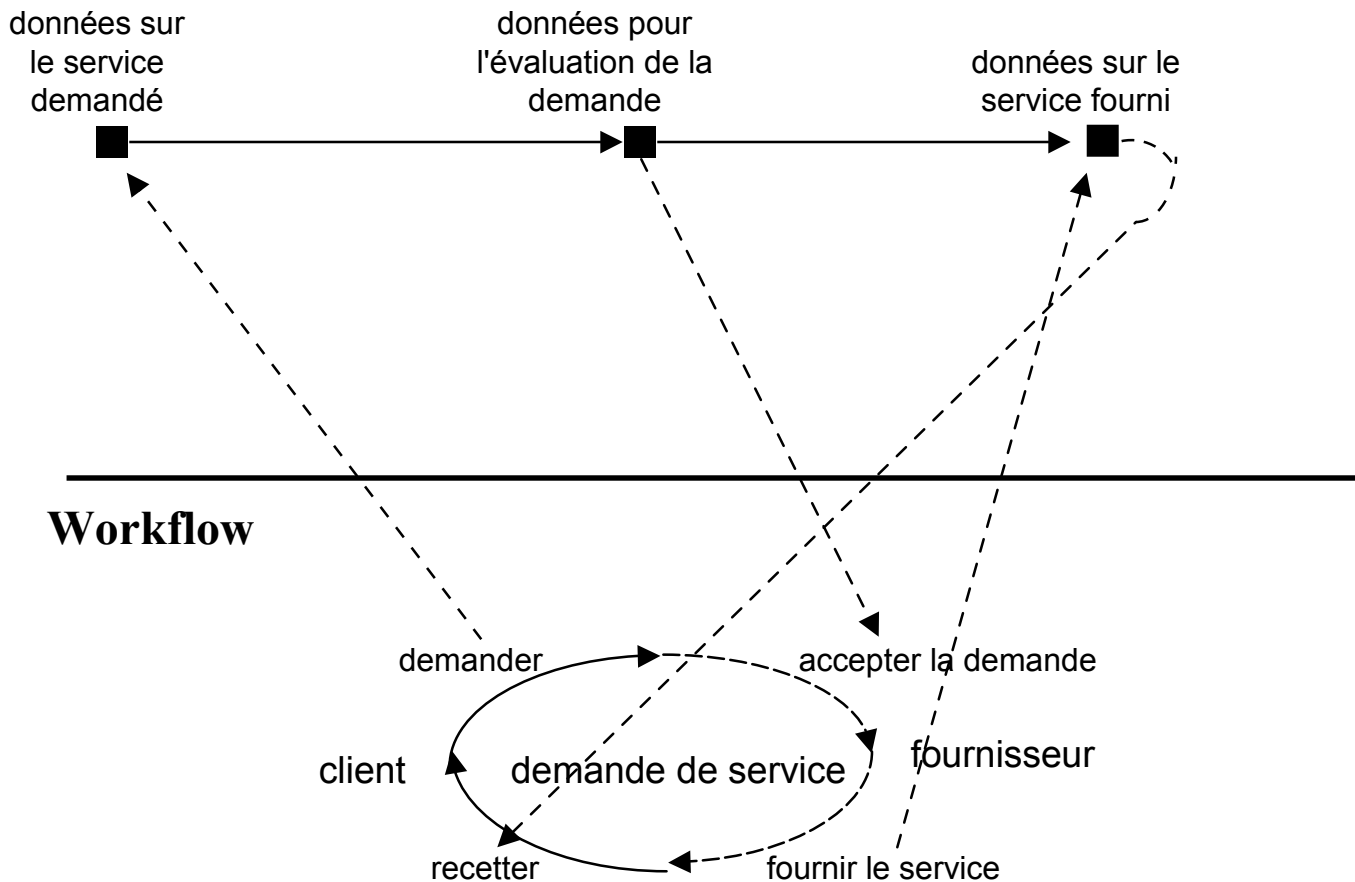


# **Perspectives pour les workflow : besoins fonctionnels**

- développement d'applications personnalisables dans un environnement graphique utilisateur
- générateur de diagrammes de flux pour définir les étapes, le routage et les conditions
- outil de simulation
- réutilisation de workflows
- zoom de processus
- définition partielle de workflow
- modification dynamique de workflow
- affectation souple des actions et des rôles
- négociation et gestion d'échéanciers
- audit de processus

# Liens entre le traitement des données et les étapes du processus de travail

## Traitement des données



# **Réalisation des liens : les transactions imbriquées**

## **Principes :**

- **rendre les données de la base de données accessibles pendant une transaction longue (transaction qui peut durer plusieurs jours voire plus)**
- **laisser la base dans un état cohérent à la fin de la transaction qu'elle se termine normalement ou non**

## **Moyen : les transactions imbriquées**

Une transaction est formée de sous-transactions, chaque sous-transaction pouvant elle-même être composée de sous-sous-transactions (formant ainsi un arbre de transactions)

le résultat d'une sous-transaction est visible aux autres sous-transactions de l'arbre de transaction ou parfois même des transactions extérieures

des dépendances entre transactions (ou sous-transactions) peuvent être spécifiées

les états acceptables de fin normale d'une transaction quand des sous-transactions ont été abortées, peuvent être spécifiées

des transactions de compensation sont associées aux sous-transactions pour laisser la base dans un état cohérent quand un conflit ou une panne intervient



## Exemples :

**Sagas** [H. Garcia-Molina, D. Gawlick, J. Klein, K. Kleissner, K. Salem, 1991, *Modeling Long-Running Activities as Nested Sagas*, In Proceedings IEEE Spring Comcon]

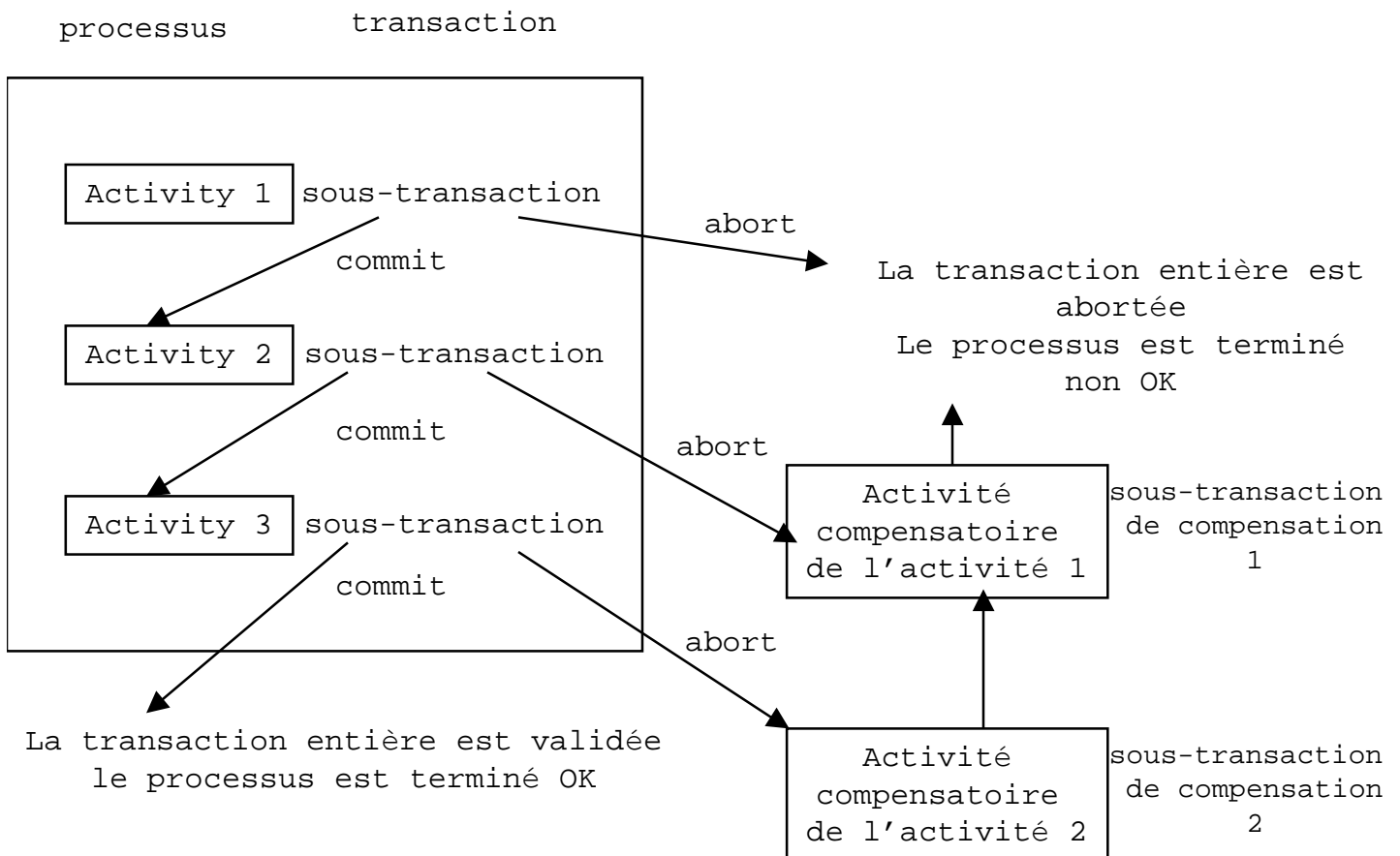
**Transactions Flexibles** [A. Zhang, M. Nodine, B. Bhargava, O. Bukhres, 1994, *Ensuring Relaxed Atomicity for Flexible Transactions in Multidatabase Systems*, in Proc. SIGMOD International Conference on Management of Data]

**Split-Join Transactions** [C. Pu, 1988, *Superdatabases for Composition of Heterogeneous Databases*, IEEE Proceedings of The Fourth Conference on Very Large Data Bases ]

**Acta** [P. Chrysanthis, K. Ramamritham, 1991, *A Formalism for Extended Transaction Model*, Proceedings of the Seventeenth International Conference on Very Large Data Bases]

**Contract** [H. Waechter, A. Reuter, 1992, *The ConTract Model*, in Database Transaction Models for Advanced Applications, A. K. Elmagarmid, editor, Morgan Kaufmann Publishers]

# Utilisation de Sagas dans la définition d'un workflow



Les sous-transactions de compensation sont toujours validées

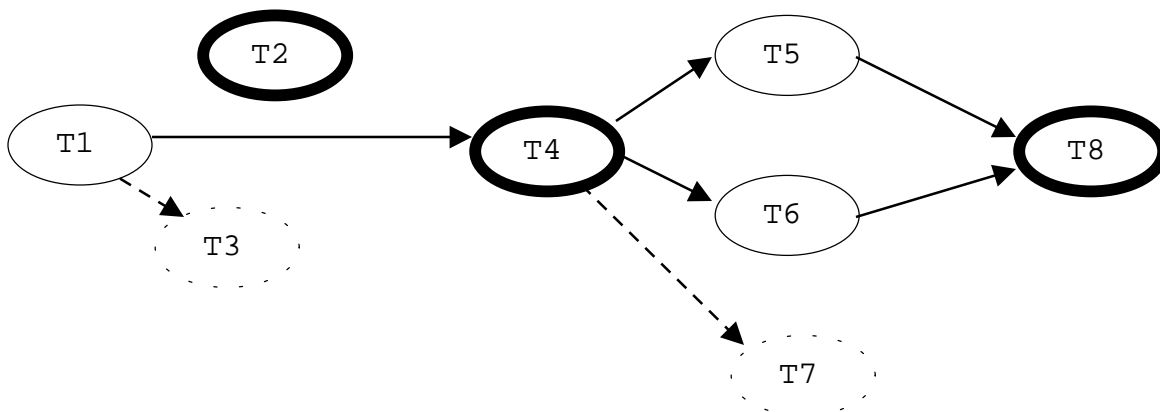
# Transactions flexibles


Contexte : base de données multiples et hétérogènes  
 chaque base de données travaille indépendamment des autres  
 une base de données peut décider unilatéralement de ne pas valider une sous-transaction

Propriétés d'une transaction flexible :

- fournit des chemins alternatifs de sous-transactions
- est validée si un chemin est validé
- une sous-transaction peut-être compensable, réessayable ou pivot

Exemple :



 trans. pivot
  trans. compensable
  trans. réessayable

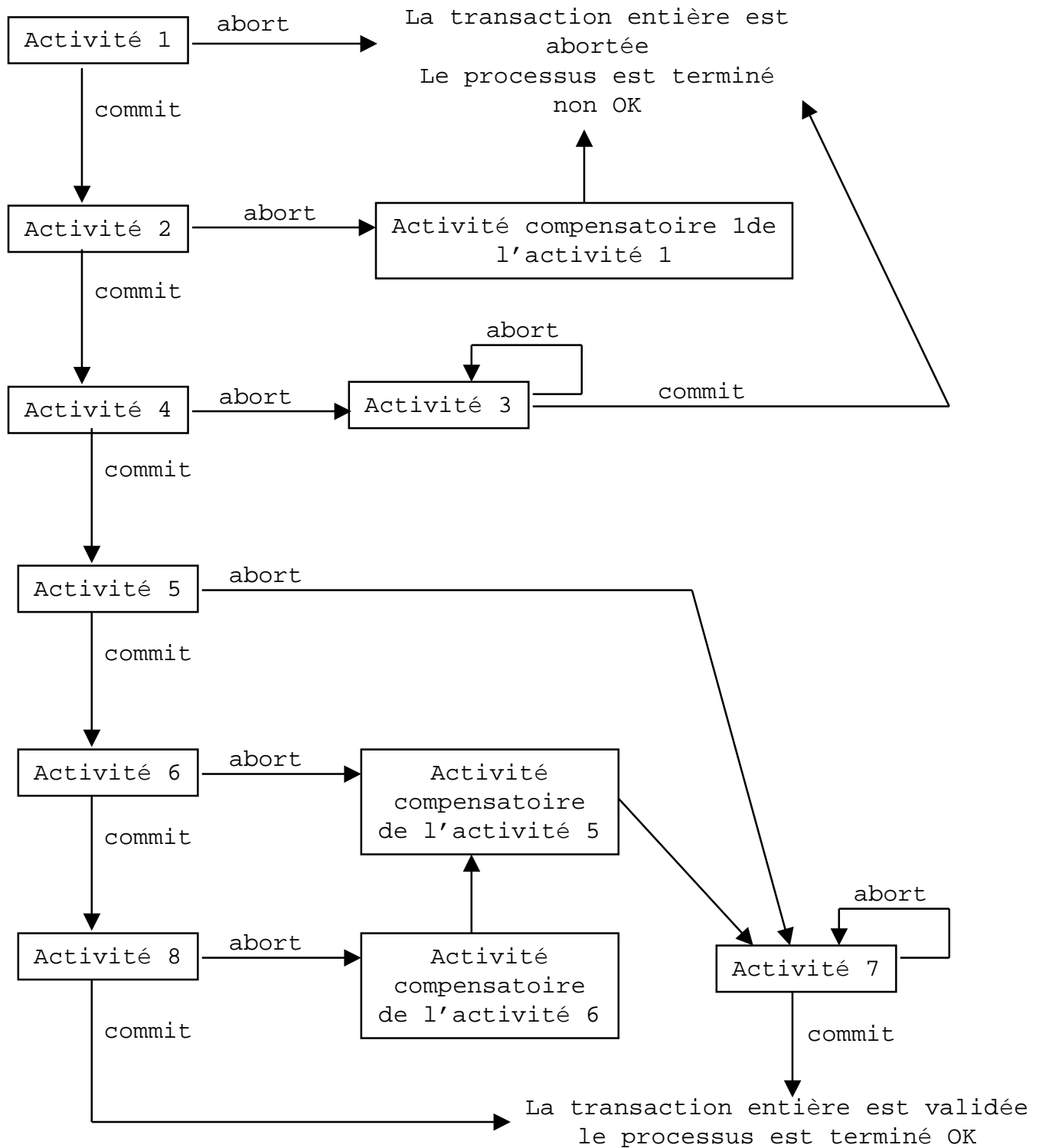
chemin 1 : {T1, T2, T4, T5, T6, T8}

chemin 2 : {T1, T2, T4, T7}

chemin 3 : {T1, T2, T3}

ordre de préférence : chemin 1, chemin2, chemin 3

## Utilisation des transactions flexibles dans la définition d'un workflow



# Les produits groupware

**des boîtes à outils comprenant tout ou partie des outils suivants :**

base de donnée partagée avec réplication ou non sur les postes clients

messaging sophistiquée (échange de messages avec priorités et notifications, archivage de messages, échange de messages multimédia, traitement de formulaires et validation par signature)

agenda de groupe

bibliothèque électronique avec classeurs

gestionnaire de tâches

forums

éditeur de formulaires

modélisation de processus

langage pour créer ses applications personnalisées

# **Gestion de l'espace partagé**

La gestion de la concurrence

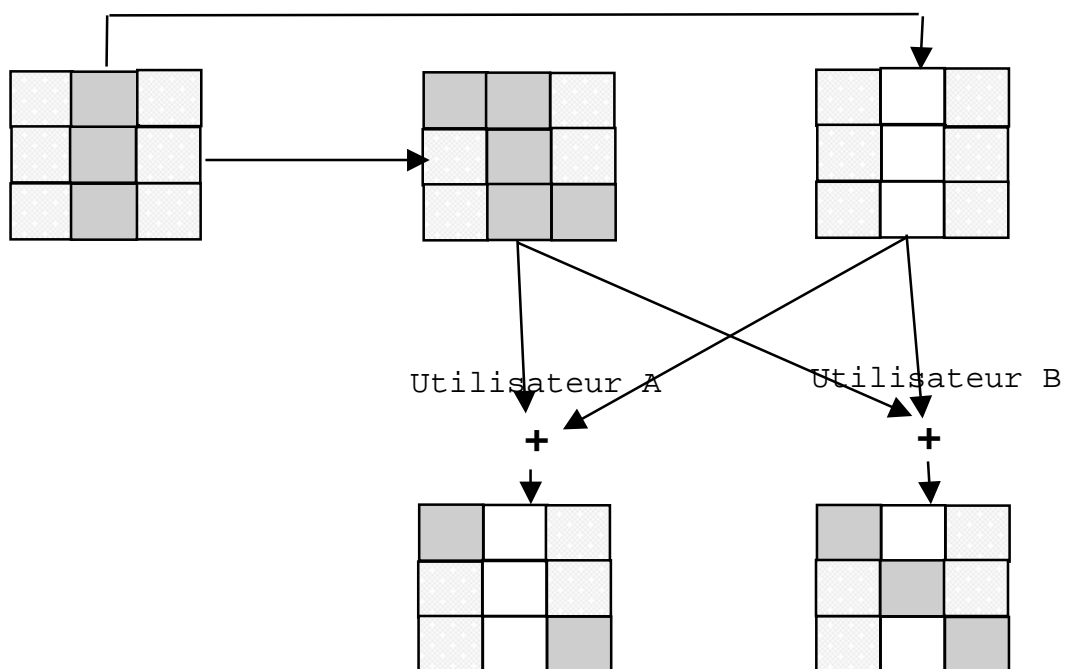
Le contrôle d'accès

# La gestion de la concurrence

## Différents modèles :

- L'ignorance de l'incohérence
- L'évitement de l'incohérence :
  - ▼ Les transactions longues
  - ▼ Le tour de parole
  - ▼ Le verrouillage d'objet
  - ▼ L'exécution réversible
  - ▼ les algorithmes de transformation d'opérations
- La gestion de l'incohérence
  - ▼ versions
  - ▼ configurations

## Ignorance de l'incohérence

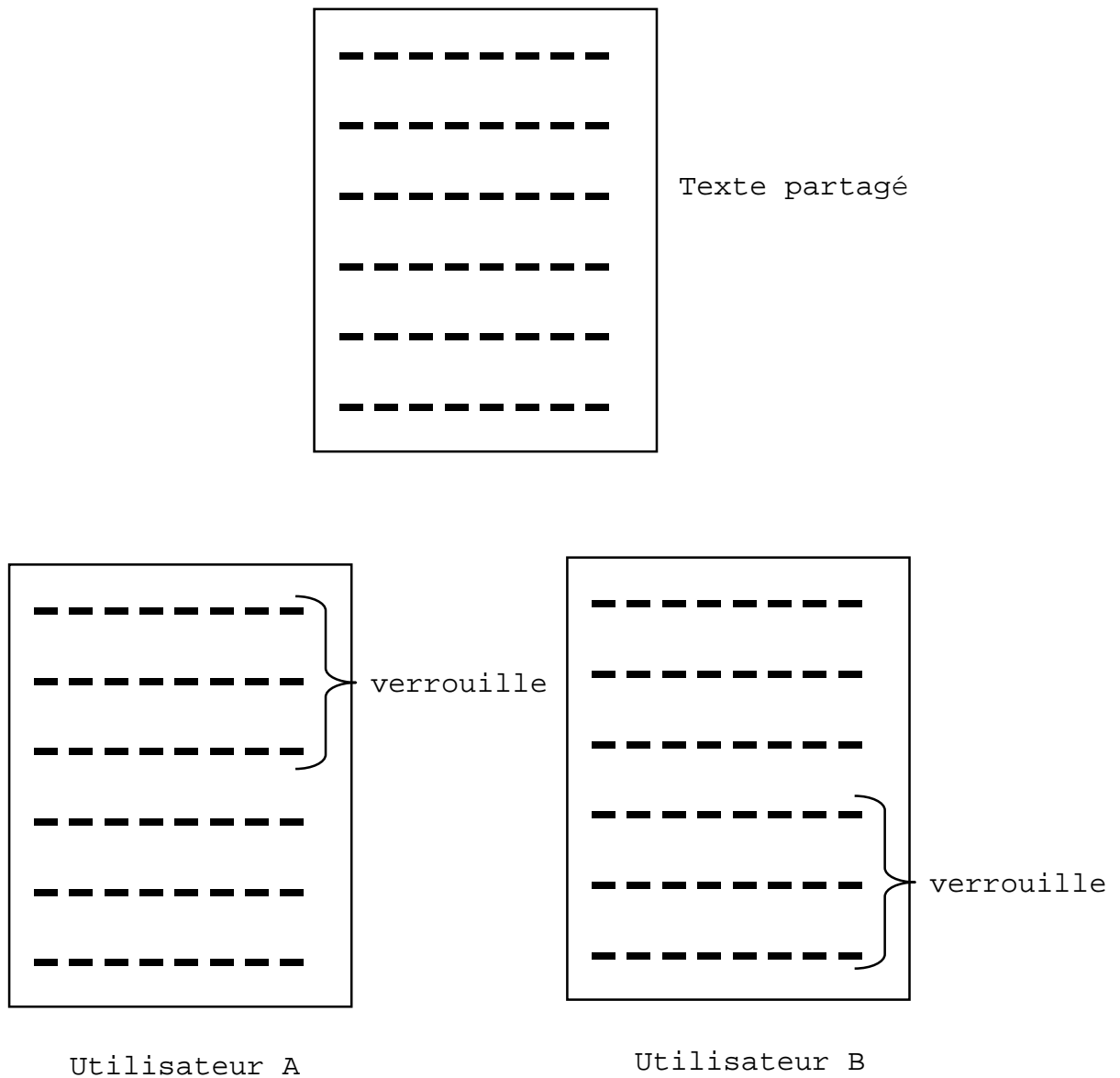




## Le tour de parole

- Un seul utilisateur n'est autorisé à interagir avec le système à un instant  $t$  (jeton circulant, demande explicite,...)

## Le verrouillage d'objet



## **L'exécution réversible**

Chaque site maintient deux listes d'opérations :

Une liste d'historique d'exécution H

Une liste d'opération en attente d'exécution Q

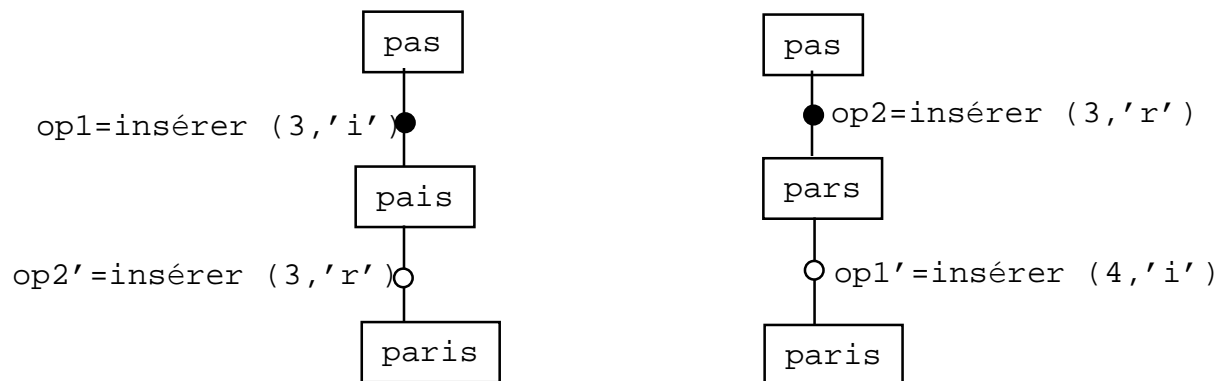
Toute opération  $o$  effectuée par un utilisateur A est exécutée localement et envoyée aux autres sites (B par exemple) accompagnée d'une horloge. Le traitement sur le site B est :

1.  $o$  s'applique à un objet non encore créé sur le site B,  $o$  est mis dans Q (B)
2.  $o$  s'applique à un objet n'existant plus sur B,  $o$  est ajoutée à H (B)
3.  $o$  s'applique à un objet existant sur B et horloge ( $o$ )  $\geq$  horloge (B),  $o$  est immédiatement exécutée et ajoutée à H (B)
4.  $o$  s'applique à un objet existant sur B et horloge ( $o$ )  $<$  horloge(B), toutes les opérations dans H (B) plus récentes que  $o$  sont défaites,  $o$  est exécutée, puis toutes les opérations défaites sont refaites.

## Les algorithmes de transformation d'opérations

Application dans les éditeurs partagés [Copies convergence in a distributed real-time collaborative environment, M. Vidot, M. Cart, J. Ferrié, M. Suleiman, ACM Computer-Supported Cooperative Work (CSCW 2000), Philadelphie, ACM Press, 2000, p : 171-190]

Exemple :



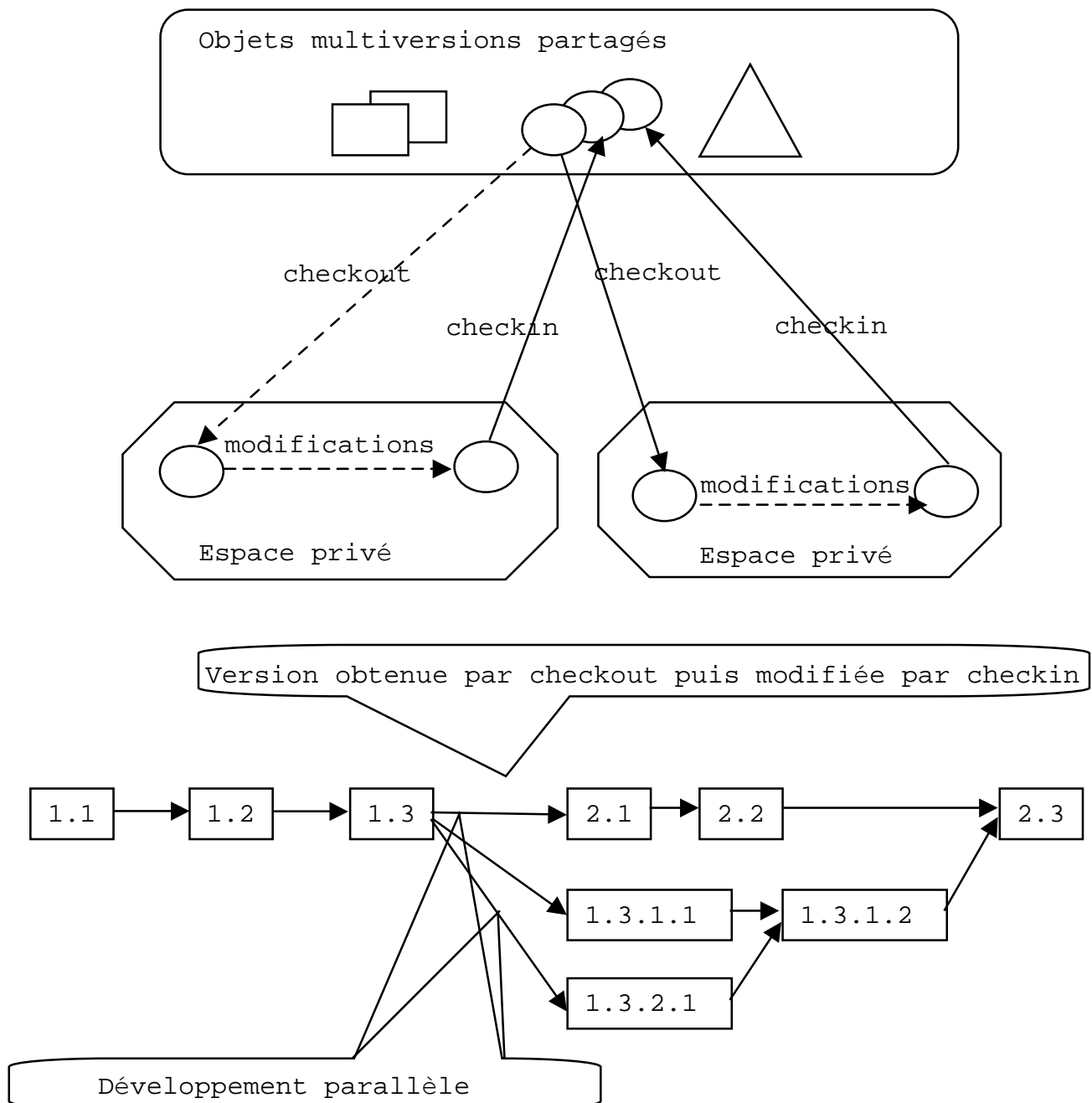
## La gestion de l'incohérence

**Le modèle checkin/checkout** (CVS, [RCS-a system for version control, W.F. Tichy, Software Practice and experience, 15, 7, 1989, p : 637-654])

Verrouillage des versions en modification

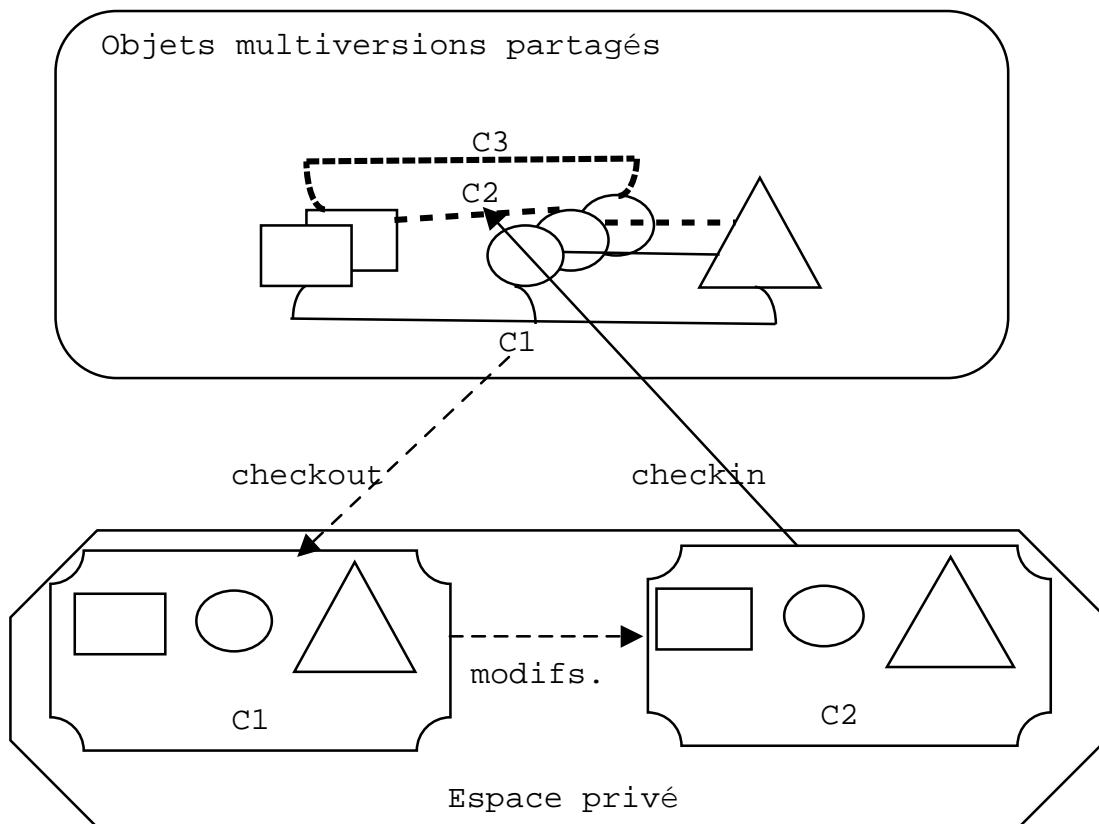
Gestion de versions des objets

Développement parallèle



## La gestion de configurations (Clearcase)

Application du modèle checkin/checkout à des configurations (ensembles nommés, cohérents et monoversionnés d'objets)



Développements parallèles : Soit les opérations suivantes :

1. L'utilisateur A exécute un checkout de la configuration C1
2. L'utilisateur B exécute un checkout de la configuration C1
3. L'utilisateur A crée une nouvelle configuration C2 puis exécute un checkin
4. L'utilisateur B crée une nouvelle configuration C'2 puis désire exécuter un checkin, il devra exécuter un checkout de la configuration C2, effectuer une fusion de C2 avec C'2 et exécuter un checkin

# Le contrôle d'accès

## Les droits d'accès :

- exprimés en termes d'opérations sur des objets et en fonction des individus
- liés aux rôles que jouent les individus
- droits de manipulation des objets, de visibilité, de couplage de vues,...
- évolutifs en fonction de la composition du groupe et de l'avancement de l'activité

exemple :

association de deux listes d'accès à chaque objet

	u1	u2
o1	lire/crire	lire

- une liste dédiée aux actions
- une liste dédiée au partage (propagation des résultats aux autres utilisateurs)

# La conscience de groupe

Une compréhension de l'activité des autres, qui procure un cadre pour sa propre activité et qui permet de s'assurer que sa contribution s'insèrent dans l'activité globale du groupe

- Conscience de l'activité des autres
- Conscience de la disponibilité des autres
- Conscience du processus commun
- Conscience des perspectives
- Conscience de l'environnement

Avec qui travaillons-nous ?

Que font les autres ?

Où travaillent-ils ?

Quand se produisent les différents évènements ?

Comment se produisent les différents évènements ?

## Les paradigmes

- le paradigme publier/souscrire : les composants rendent public leurs changements via des événements à destination des composants qui ont manifesté un intérêt pour cette classe d'évènements
  - ✓ souscription basée sur le sujet du message
  - ✓ souscription basée sur le contenu du message
- le paradigme spatial qui exploite les positions et les distances des participants par rapport aux objets touchés

## La mise en oeuvre

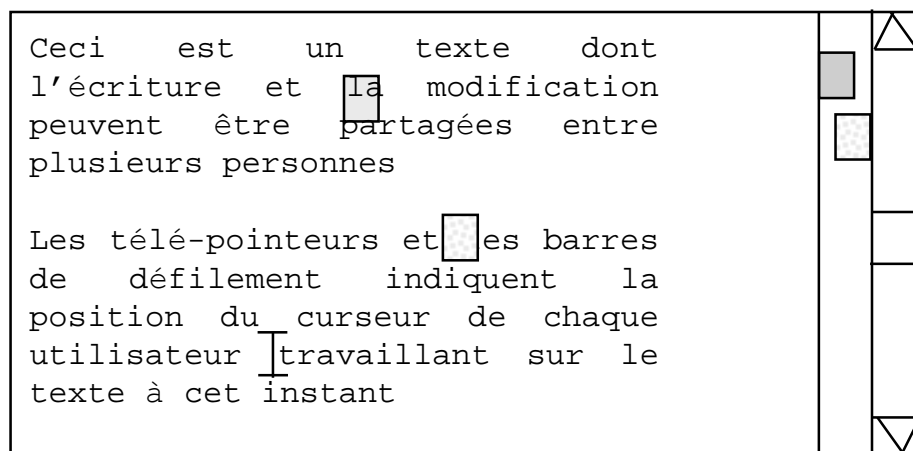
### Les principes

- la non distraction
- la non interférence fonctionnelle
- la différenciation entre les rôles joués par les participants et leur degré d'expertise
- l'adaptabilité
- la liaison de l'information avec des actions possibles

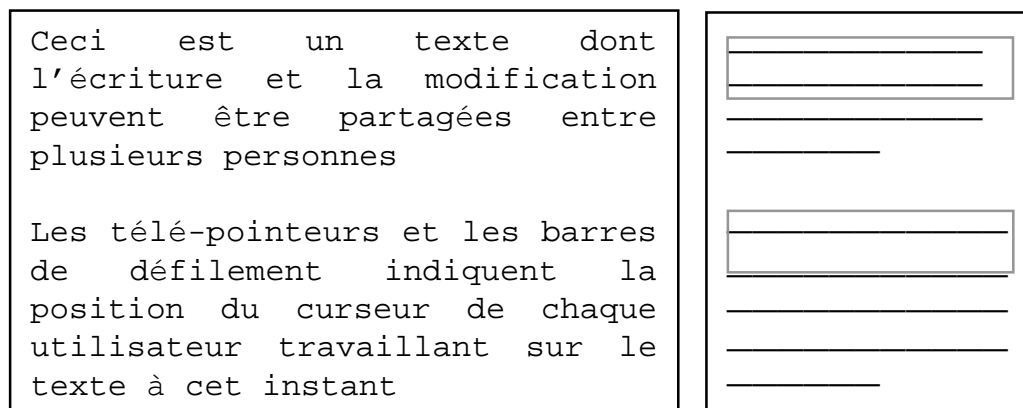


## Les moyens de mise en oeuvre

- notification asynchrone (par exemple par la messagerie)
- notification asynchrone (par exemple par la messagerie)
- indication de la présence et de l'état des participants dans un environnement partagé (par exemple, une page web)
- utilisation de la métaphore de la pièce
- télé-pointeurs et barres défilement multiples



- fenêtre supplémentaire ou vue radar



## Les moyens de mise en oeuvre (fin)

- intégrer le mécanisme dans le texte principal par exemple par un halo coloré, une transformation de texte ou un loupe

Ceci est un texte dont  
l'écriture et la modification  
peuvent être partagées entre  
plusieurs personnes

Les télé-pointeurs et les barres  
de défilement indiquent la  
position du curseur de chaque  
utilisateur travaillant sur le  
texte à cet instant

Ceci est un texte dont l'écriture et la  
modification peuvent être partagées entre plusieurs  
personnes

Les télé-pointeurs et les barres  
de défilement indiquent la position du  
curseur de chaque utilisateur  
travaillant sur le texte à cet  
instant

l'écriture et la modification  
peuvent être partagées entre

restitution visuelle viable pour un petit groupe de participants

## **Exemple d'outil de groupware : BSCW**

- concept de base
- actions sur les workspaces
- partage d'un workspace, d'un objet
- évènements associés à un objet
- documents versionnés
- gestion des réunions et des agendas
- application

## Concept de base

l'espace de travail partageable (workspace)

un workspace = un dossier (folder) qui peut contenir :

- des documents non versionnés
- des documents versionnés
- des URL
- des notes
- des résultats de recherche
- des utilisateurs
- des dossiers des réunions
- des discussions
- des groupes
- un calendrier de réunions
- un carnet d'adresse
- un presse-papier
- une poubelle
- un espace public

## **Actions sur un workspace**

- création d'un dossier dans le dossier courant
- création d'un dossier "réunion"
- création d'un dossier "discussion"
- création d'une note dans le dossier "discussion" courant
- création d'un résultat de recherche, avec la spécification d'une zone de recherche, d'un moteur de recherche et d'une requête
- création d'un URL dans le dossier ou dans le dossier "réunion" courant

chaque objet a des actions associées

## Partage d'un workspace, d'un objet

- ajout de membres ayant accès au workspace ou à l'objet
- définition des droits d'accès

par défaut tous les membres d'un workspace ont tous les droits sur les objets du workspace sauf les trois droits suivants réservés au propriétaire des objets :

changer les droits  
supprimer l'objet  
modifier une note

les membres anonymes ont seulement les droits en lecture

les droits par défaut peuvent être modifiés

- espace public

## **Evènements associés à un objet** (notifications configurables par utilisateur)

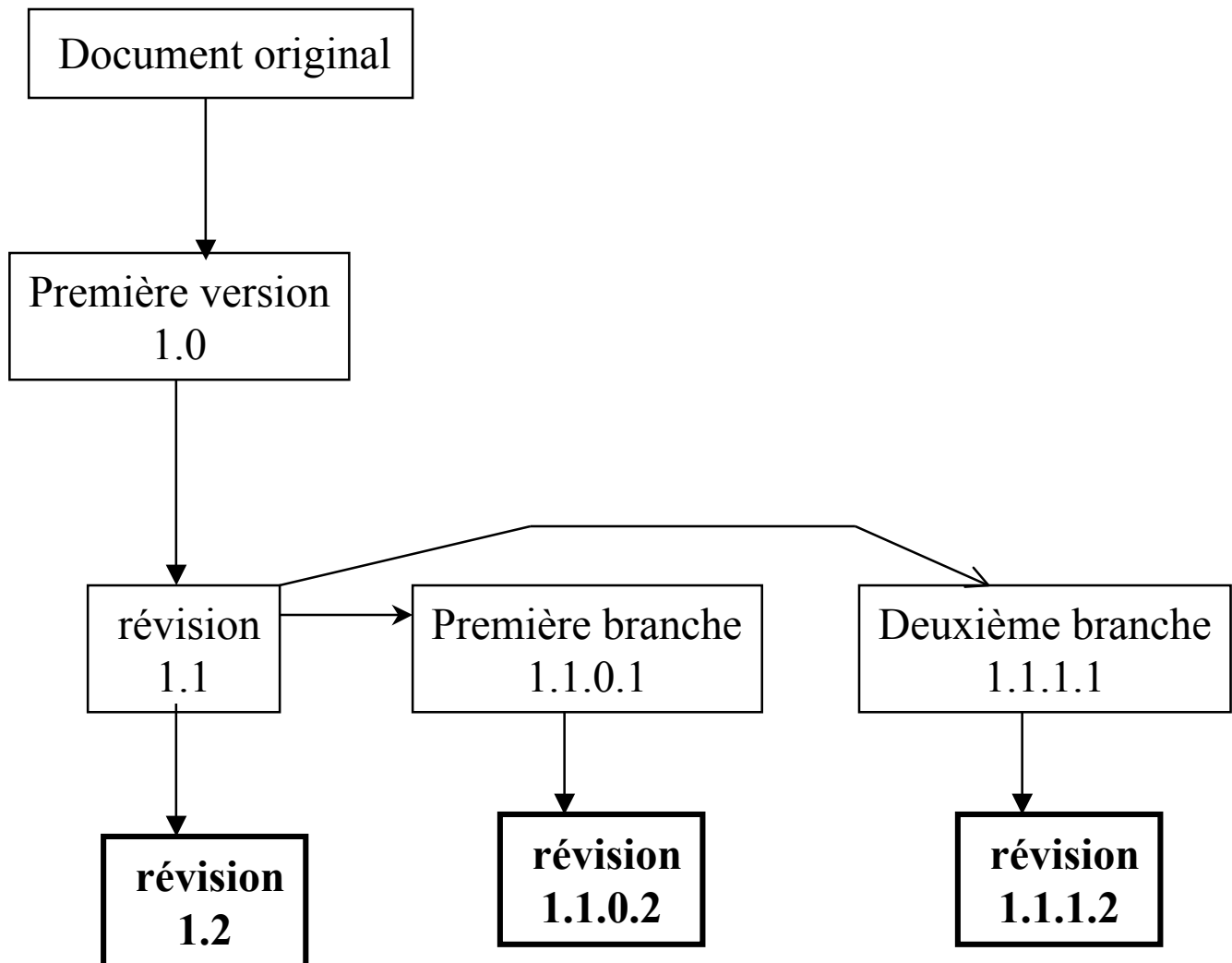
- nouveautés
- changements
- déplacements
- accès

+

- historique de tous les événements associés à un dossier
- modifications dans un dossier

# Documents versionnés

## Principes du versionnement



Seuls les derniers documents de chaque branche sont modifiables



## **Gestion des réunions et des agendas**

- création d'une réunion
- association de participants à une réunion
- notification par envoi d'un email
- consultation de son agenda
- réponse aux invitations de réunions

## **Application**

1. Utilisation de l'espace de travail blin
2. Création de votre espace avec un forum de discussion
3. Création d'un forum de discussion dans votre espace
4. Partageabilité de votre espace, ajout de membres, modification de leurs droits
5. Ajout de notes de discussion, lecture des réponses, réponse à une réponse
6. Traitement des évènements
7. Configuration des notifications d'évènements
8. Gestion du carnet d'adresses
9. Utilisation de documents versionnés, création de révisions et de variantes (les branches)
10. Gestion des réunions

# Le groupware temps réel

## Concepts :

- existence d'un groupe d'utilisateurs concerné par un projet commun
- interactions synchrones ou asynchrones distribuées
- un contexte partagé tel que, une action faite sur un objet par un utilisateur est visible simultanément par tous les autres utilisateurs
- groupe de fenêtres : les fenêtres de tous les utilisateurs sont liées tel que si, par exemple, un utilisateur dessine un cercle dans sa fenêtre, le cercle apparaît dans les fenêtres de tous les autres utilisateurs
- télécurseur : un curseur apparaît dans toutes les fenêtres, si un utilisateur le déplace dans sa fenêtre, il est déplacé dans toutes les autres fenêtres
- vue : visualisation d'une partie du contexte, représentations multiples
- rôles avec autorisations associées

## Questions encore à l'ordre du jour

### Fenêtres groupées :

- relaxation du mode What You See Is What I See  
application souple de ce concept sur l'emplacement des objets dans les fenêtres, le moment de mise à jour des fenêtres, les utilisateurs concernés par les mises à jour.

Exemple : 2 personnes A et B travaillent sur le même texte. Quand A modifie le texte, un nuage apparaît sur l'écran de B à l'emplacement de la modification. La modification n'apparaîtra à B que lorsqu'il deviendra inactif.

- Prolifération des fenêtres  
faut-il créer des sous-groupes de fenêtres ?
- Télécursseurs  
un télécursseur par sous-groupe de fenêtres ?

### Intégration de protocoles de travail dans l'outil

si oui, plus grande efficacité de travail mais plus de rigidité et difficulté pour les nouveaux utilisateurs

si non plus de souplesse pour traiter les exceptions, mais obligation d'instituer un protocole social

## Performances

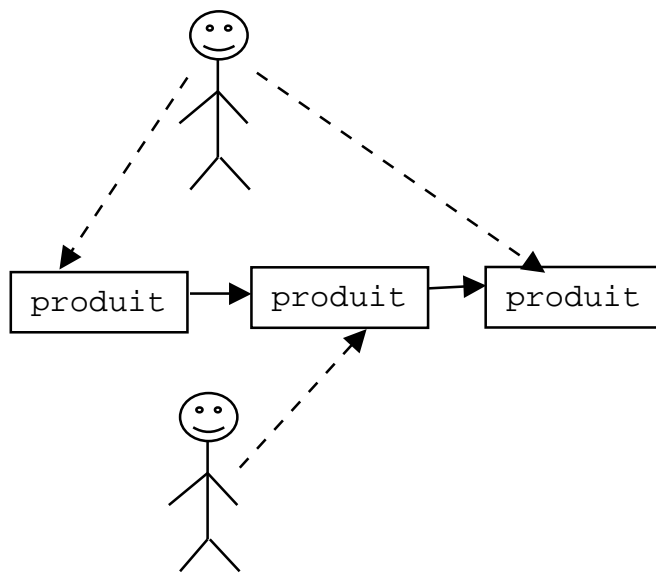
réplication des objets sur tous les sites, problème de concurrence

si simple système de verrouillage : quelle granularité de verrou ?  
quand poser les verroux ?  
quand les libérer ?

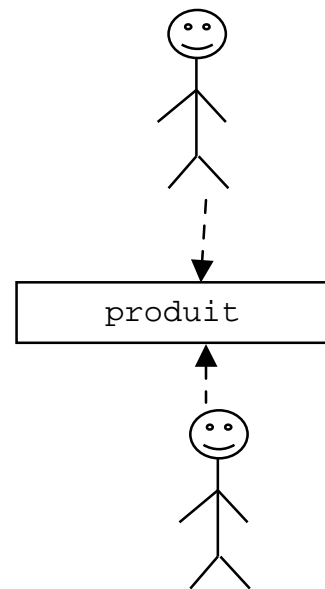
si système transactionnel : risque de verrouillage de trop longue durée

# Deux applications particulières : la conception et le développement collaboratif de logiciels

## La conception collaborative



*conception distribuée*



*co-conception*

choix des critères de décision, évaluation de leur importance

décision collective

production collective de la solution

intégration et confrontation des points de vue

capitalisation et réutilisation des connaissances de conception

## Les mémoires de projet

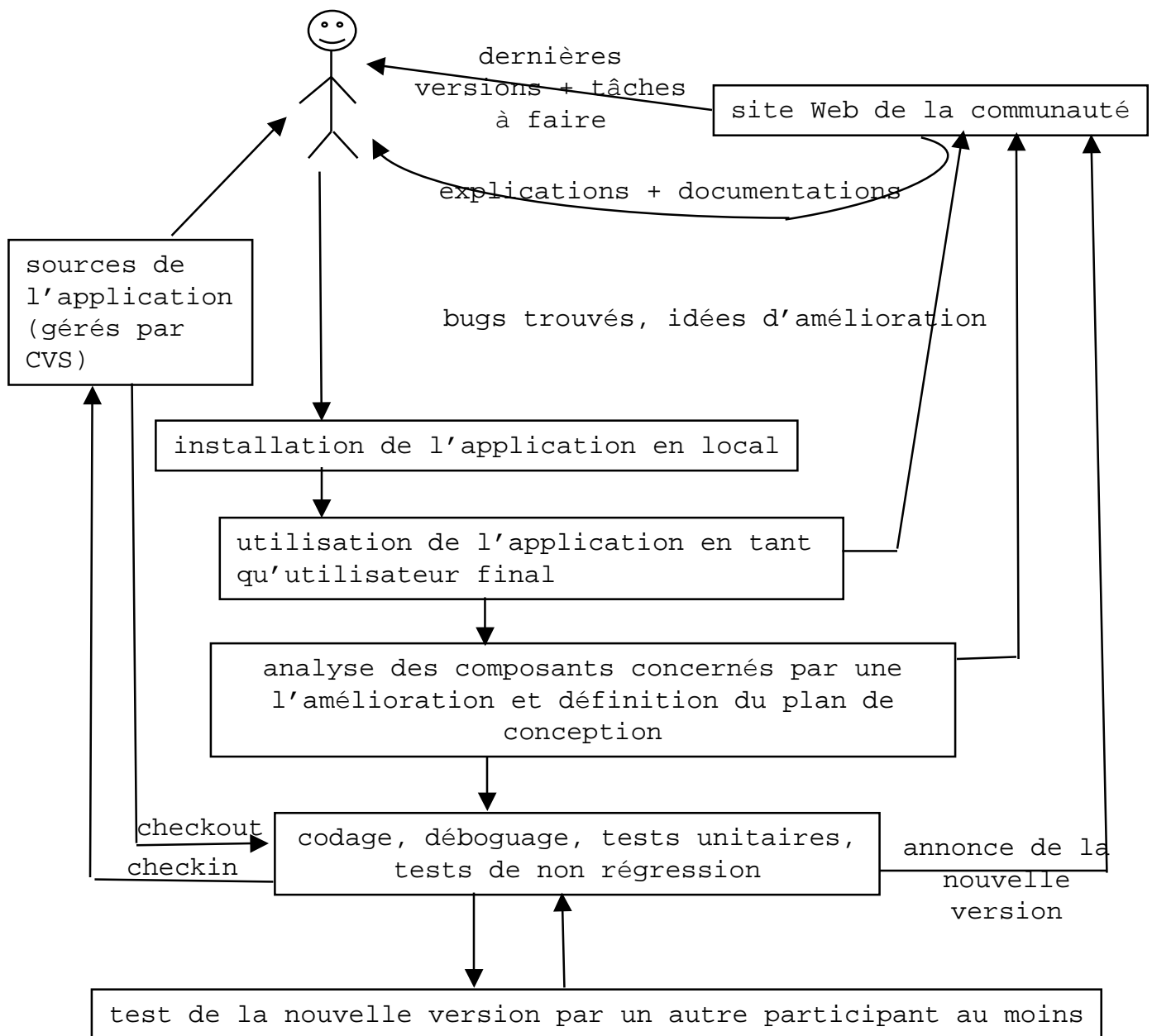
- problèmes rencontrés : nature, éléments, ...
- résolutions : participants, méthodes de résolution, choix potentiels, ...
- évaluations des solutions rejetées : arguments, avantages, inconvénients,...
- décisions : solution choisie, arguments, avantages, inconvénients, ...

Recueil des connaissances directement et dynamiquement au fil des activités :

- traçage de l'utilisation des fonctions,
- traçage des informations produites et échangées
- déduction d'informations de plus haut niveau par exemple sur la stratégie de conception suivie

# Le développement coopératif de logiciels open source

## Le processus



voir [www.sourceforge.org](http://www.sourceforge.org)



## **Caractéristiques de la coopération**

- engagements des participants non explicites
- les participants sont enregistrés et constituent des listes de diffusion
- travail spontané et rapide
- toute information est diffusée à tous les participants enregistrés
- chaque soumission est revue par au moins un pair
- chaque tâche n'est conduite que par un seul participant
- les communications se font exclusivement en mode asynchrone et par écrit
- peu de positions abstraites ou non réfléchies
- de contributions techniques précises, peu de bruit
- peu de structuration des échanges
- partage d'une forme de culture commune