

Table de matières

CHAPITRE 1. INTRODUCTION GENERALE	1
1.1 CONTEXTE	2
1.2 LES CONTRIBUTIONS DE CETTE THESE.....	2
1.3 PLAN DE LA THESE	4
CHAPITRE 2. LA PROBLEMATIQUE DE COORDINATION DES ACTIONS DANS LA GESTION DES SESSIONS COOPERATIVES : CARACTERISTIQUES ET BESOINS	5
2.1 INTRODUCTION	5
2.2 LES APPLICATIONS COOPERATIVES DISTRIBUEES	6
2.2.1 <i>Le travail coopératif assisté par ordinateur.....</i>	<i>6</i>
2.2.2 <i>Le collecticiel</i>	<i>6</i>
2.2.3 <i>Quelle est la différence entre TCAO et Collecticiel ?</i>	<i>7</i>
2.2.4 <i>Quelle est la différence entre les collecticiels et les systèmes multi-utilisateur ?</i>	<i>7</i>
2.2.5 <i>Classification du collecticiel</i>	<i>7</i>
2.2.6 <i>Les applications coopératives distribuées.....</i>	<i>8</i>
2.3 LA COORDINATION ET LES APPLICATIONS COOPERATIVES DISTRIBUEES	9
2.3.1 <i>Coordination, coopération et communication.....</i>	<i>9</i>
2.3.2 <i>Gestion de problèmes de coordination dans les collecticiels.....</i>	<i>11</i>
2.4 LA COORDINATION DE SESSIONS DANS LES APPLICATIONS COOPERATIVES DISTRIBUEES	14
2.4.1 <i>Définition de session coopérative.....</i>	<i>14</i>
2.4.2 <i>Formes de sessions coopératives</i>	<i>14</i>
2.4.3 <i>Types de sessions coopératives</i>	<i>16</i>
2.4.4 <i>La gestion de sessions coopératives.....</i>	<i>16</i>
2.4.5 <i>Problèmes de coordination dans la gestion des sessions coopératives.....</i>	<i>16</i>
2.4.6 <i>Systèmes de gestion de session.....</i>	<i>17</i>
2.4.7 <i>Standards et recommandations pour la gestion de sessions</i>	<i>23</i>
2.4.8 <i>Modèles pour la gestion de sessions</i>	<i>26</i>
2.5 SYNTHÈSE DE NOS CONTRIBUTIONS	31
2.5.1 <i>Application à un scénario de coordination en ingénierie coopérative distribuée... ..</i>	<i>32</i>
2.6 CONCLUSION	35
CHAPITRE 3. FORMALISATION DE LA GESTION DES SESSIONS MULTI- APPLICATIONS	37
3.1 INTRODUCTION	37
3.2 DESCRIPTION FONCTIONNELLE DE LA GESTION DES SESSIONS.....	38
3.2.1 <i>Responsibility and User Management Service RMS</i>	<i>39</i>
3.2.2 <i>Session Management Service SMS.....</i>	<i>42</i>
3.3 MODELE DE COORDINATION DE SESSIONS DE COOPERATION.....	47
3.3.1 <i>Modélisation de sessions de coopération</i>	<i>48</i>
3.3.2 <i>Modélisation des règles de coordination</i>	<i>49</i>
3.4 MODELISATION DES DEPENDANCES ENTRE EVENEMENTS	50
3.4.1 <i>L'énoncé de précedence</i>	<i>50</i>
3.4.2 <i>L'énoncé d'inhibition</i>	<i>50</i>
3.4.3 <i>L'énoncé de déclenchement.....</i>	<i>51</i>
3.5 APPLICATION DU MODELE : SPECIFICATION D'UNE SESSION DE COOPERATION AVEC DES REGLES DE COORDINATION DE BASE	52
3.5.1 <i>Définition des acteurs.....</i>	<i>52</i>
3.5.2 <i>Définition des actions.....</i>	<i>52</i>
3.5.3 <i>Règles de d'ordonnancement des événements contrôlant le changement d'état</i>	<i>54</i>
3.5.4 <i>Règles d'appartenance au groupe (membership rules).....</i>	<i>55</i>
3.5.5 <i>Règles de spécification de la portée des communications à l'intérieur et à l'extérieur du groupe.....</i>	<i>56</i>

3.5.6 Règles de consistance des rôles.....	56
3.6 CoDES : PROCESSUS COOPERATIF DE CONCEPTION ET D'ANALYSE DANS LE DOMAINE AEROSPATIALE.....	57
3.7 SCENARIO PDR.....	57
3.7.1 Objectifs de PDR.....	58
3.7.2 Acteurs.....	58
3.7.3 Modélisation de la revue PDR	59
3.8 REGLES DE COORDINATION POUR UN CAS DE L'INGENIERIE COOPERATIVE DISTRIBUEE (PDR).....	64
3.8.1 Définition des participants	64
3.8.2 Définition des applications.....	65
3.8.3 Règles de coordination des participants	65
3.8.4 Règles de coordination des applications.....	66
3.9 MISE EN ŒUVRE DU SCENARIO PDR EN UTILISANT UN OUTIL WORKFLOW	68
3.10 CONCLUSION	73
CHAPITRE 4. ENVIRONNEMENT D'AIDE A LA CONFIGURATION ET A LA GESTION DES SESSIONS MULTI-APPLICATIONS.....	75
4.1 INTRODUCTION	75
4.2 PREPARATION DES SESSIONS COOPERATIVES	76
4.2.1 Structuration de groupes ouverts	76
4.2.2 Diagramme des classes RMS	76
4.2.3 Configuration des sessions de coopération.....	78
4.3 GESTION DE SESSIONS COOPERATIVES MULTI-APPLICATIONS	79
4.3.1 Diagramme des classes SMS.....	79
4.3.2 Diagrammes de séquences UML du service SMS	86
4.3.3 Définition des règles de coordination par des contraintes OCL.....	94
4.4 ARCHITECTURE DES SERVICES DE GESTION DES SESSIONS COOPERATIVES MULTI-APPLICATIONS RMS ET SMS	97
4.4.1 CORBA.....	97
4.4.2 Java Mail.....	98
4.4.3 JDBC.....	98
4.4.4 JSDT.....	98
4.4.5 Java Web Start	99
4.4.6 Remote Method Invocation (RMI)	99
4.4.7 XML.....	100
4.4.8 Architecture élaborée.....	100
4.5 SYSTEMES ORIENTES EVENEMENTS.....	103
4.5.1 Systèmes distribués orientés événements.....	103
4.5.2 Définition d'un événement.....	104
4.5.3 Types d'architectures des systèmes distribués orientés événements	104
4.6 SERVICE DE GESTION DES SESSIONS COOPERATIVES ORIENTE EVENEMENTS	105
4.6.1 Définition des événements de l'état de la session.....	105
4.6.2 Définition des événements des participants.....	106
4.6.3 Définition des événements d'information.....	107
4.6.4 Définition des événements des applications	107
4.7 CONCLUSION	108
CHAPITRE 5. SCENARIO DE VALIDATION : APPLICATION DU GESTIONNAIRE DES SESSIONS DANS UN CAS DE L'INGENIERIE COOPERATIVE DISTRIBUEE	109
5.1 INTRODUCTION	109
5.2 LE PROJET DSE.....	109
5.2.1 Objectifs du projet.....	110
5.2.2 Le système : architecture et composants.....	112
5.3 SCÉNARIO CO-DES : PDR-ATV	114
5.3.1 Exécution de la revue PDR-ATV en utilisant l'environnement DSE.....	116

5.3.2	<i>Tâches PDR-ATV avec l'environnement DSE</i>	118
5.3.3	<i>Configuration et préparation de l'environnement DSE</i>	118
5.3.4	<i>Préparation de la revue préliminaire de conception</i>	121
5.3.5	<i>Les critiques analysent des documents de la base de référence et publient RID</i> ...	125
5.3.6	<i>L'équipe de projet répond aux RID</i>	125
5.3.7	<i>Exécution d'une session pour analyser un RID particulier</i>	125
5.4	ANALYSE DES BENEFICES DANS PDR-ATV EN UTILISANT L'ENVIRONNEMENT DSE.....	130
5.5	RESULTATS DE L'EVALUATION DES SERVICES RMS ET SMS	134
5.5.1	<i>Questionnaire d'évaluation de gestion de responsabilité</i>	135
5.5.2	<i>Questionnaire d'évaluation de gestion de session</i>	136
5.5.3	<i>Questionnaire d'évaluation des réunions distribuées</i>	137
5.6	CONCLUSION	138
CHAPITRE 6. CONCLUSION GENERALE		139
6.1	RAPPEL DES CONTRIBUTIONS	139
6.2	PERSPECTIVES.....	141
CHAPITRE 7. BIBLIOGRAPHIE DE L'AUTEUR		145
CHAPITRE 8. BIBLIOGRAPHIE		149

Chapitre 1. Introduction générale

Le développement des réseaux d'ordinateurs au cours des deux dernières décennies a permis de rendre la communication par ordinateur plus rapide, de meilleure qualité et moins chère. Dans cette nouvelle infrastructure de communication est née la possibilité de travailler en groupe et sont apparues de nouvelles applications distribuées. Ceci constitue le contexte de l'émergence du domaine de recherche de travail coopératif assisté par ordinateur TCAO (ou CSCW en anglais). Le TCAO est situé au carrefour des sciences informatiques, des sciences humaines et d'autres disciplines. Son objectif est d'étudier comment les personnes travaillent en groupe et comment la technologie peut les aider. C'est ainsi que de nouvelles applications informatiques surgissent, des applications capables de réunir plusieurs utilisateurs autour d'un même projet, d'une même activité, ou simplement d'un même document, afin de fondre leurs efforts et leurs compétences pour atteindre un but commun. Ces logiciels sont appelés collecticiels (ou groupware en anglais). Ils ont pour objectif de faciliter le travail de groupe en permettant la manipulation d'objets partagés et leur évolution.

Dans la famille des collecticiels, on trouve des applications qui possèdent comme caractéristiques principales la coopération et la distribution. Ce type d'applications soutient des activités qui sont à la fois coopératives et distribuées, comme notamment la téléformation et l'ingénierie concurrente. Ce type d'activités est caractérisé par des groupes d'utilisateurs physiquement (ou virtuellement) distribués qui coopèrent par des interactions et qui sont regroupés en sessions de travail. Ces sessions constituent les unités de base pour la coopération. Deux catégories de sessions de coopération existent : les sessions asynchrones et les sessions synchrones. Dans la première catégorie, les utilisateurs agissent selon des procédures séquentielles ou parallèles sur des données locales. La deuxième catégorie est constituée par les sessions synchrones au cours desquelles les participants agissent simultanément et depuis des points

d'accès distribués sur des objets partagés en suivant des règles de coopération et en utilisant un ensemble d'applications qui leur permet de progresser de façon coordonnée. Les travaux de cette thèse se situent dans cette deuxième catégorie.

Nous nous sommes intéressés aux différentes catégories de dépendances qui régissent les interactions entre participants et applications à l'intérieur des sessions synchrones de coopération. Le résultat effectif de la coopération lors d'une session est le produit des actions simultanées et concurrentes effectuées pendant la définition et l'exécution de la session. Ces actions nécessitent d'être coordonnées afin d'éviter les incohérences. Ceci constitue la motivation principale de nos travaux.

1.1 Contexte

Cette thèse s'inscrit dans l'ensemble des études menées au sein du projet européen DSE (Distributed Systems Engineering). DSE est un projet de recherche Européen (2000-2002) cofinancé par la Commission Européenne dans le cadre du Programme IST. Il a été réalisé par un consortium international comprenant deux laboratoires de recherche¹, trois sociétés fournissant des compétences technologiques² et trois sociétés industrielles ayant une forte implication dans des activités du domaine spatial³. Les principaux objectifs du projet sont concentrés sur l'amélioration du processus d'ingénierie coopérative en permettant à différentes équipes géographiquement éloignées d'agir réciproquement à partir de leur propre emplacement, ayant simultanément accès et contrôle sur des applications à distance, des répertoires de données globaux ou des fichiers. Le projet DSE a été prévu pour définir et construire un environnement modulaire capable de supporter des applications utilisées sur une plate-forme d'ingénierie coopérative dans l'industrie spatiale européenne.

1.2 Les contributions de cette thèse

Les principales contributions de notre travail sont les suivantes :

- L'identification des interdépendances entre actions lors de sessions de coopération multi-utilisateurs multi-applications. Sur la base des discussions avec les utilisateurs finals et en analysant les exigences de l'activité de coopération dans le domaine de l'ingénierie système distribuée, nous avons d'abord identifié trois types de modalités d'interdépendance : la précedence, l'inhibition et le déclenchement. Ces trois modalités répondent aux dépendances inter-applications, intra-applications, inter-participants et participants-applications identifiées. Les dépendances inter-applications permettent de coordonner des applications de type différent, e.g. la coordination entre le gestionnaire de droit de parole et l'application de vidéoconférence. Les dépendances intra-applications sont celles

¹ LAAS-CNRS et Université P&M Curie - Laboratoire (LIP6)

² SILOGIC, SOCIETA ITALIANA AVIONICA S.p.A. (SIA) et D3 GROUP Softwareentwicklung, Forschung

³ ALENIA AEROSPAZIO Divisione Spazio - Un'Azienda FINMECCANICA S.p.A., AEROSPATIALE MATRA Lanceurs Stratégiques & Spatiaux (EADS-LV) et INDUSTRIEANLAGEN-BETRIEBSGESELLSCHAFT GmbH (IABG)

qui impliquent les composants d'une même application, e.g. la coordination des entités dans un système distribué avec n serveurs, 1 proxy et n clients. Les dépendances inter-participants concernent la coordination des actions effectuées par les participants afin d'éviter l'annulation ou l'incohérence des actions, e.g. l'entrée des participants ordinaires doit être précédée par l'entrée du président de la session. Les dépendances entre participants et applications apparaissent lorsqu'une action relative à une application affecte le comportement des participants ou vice-versa. C'est le cas par exemple de l'interdiction de l'entrée de participants retardataires qui est spécifiée par la dépendance entre l'action d'ouverture de la session et l'action d'entrée des participants. Nous avons ensuite défini un ensemble de règles de coordination de base pour des sessions de coopération multi-utilisateurs multi-applications. Ces règles représentent des sessions qui comptent différents participants distribués qui exécutent un ensemble d'applications de différents types.

- La formalisation, la structuration et la généralisation des interdépendances par des ordres partiels étiquetés et des formules logiques. Ceci nous a permis de créer un cadre formel capable de spécifier de manière générale les règles de coordination. Cette formalisation constitue la synthèse de notre réflexion sur les interdépendances entre les différents acteurs et entités d'une session de coopération multi-applications multi-utilisateurs. Elle peut servir de modèle référence pour la vérification de nouvelles spécifications et pour le test des nouvelles implémentations des services de coordination.
- La conception et la mise en œuvre d'un service de gestion de session qui fournit les fonctions permettant aux participants de définir, d'initialiser, de chercher, de se joindre à, de quitter, d'ouvrir, de clore et de terminer une session. Deux services principaux ont été développés. Le premier permet de planifier une session de coopération. Le deuxième permet de gérer le déroulement d'une session programmée en commençant par les invitations, et en terminant par la fermeture de la session tout en passant par différents états de contrôle dont l'ouverture et la clôture de la session. Des interfaces graphiques conviviales sont mises à disposition des utilisateurs de notre service (administrateurs, animateurs de sessions ou simple participants). Des fonctions de mémoire de groupe (*awareness*) sont fournies pour les participants et pour l'animateur leur permettant ainsi de suivre ou de contrôler le déroulement d'une session de coopération.

L'expérimentation et la validation du service développé dans le cadre d'un cas d'utilisation réel à une échelle réduite d'un scénario de l'industrie spatiale dans le cadre de la phase de validation de l'environnement logiciel du projet DSE. L'implémentation et la validation se sont passées de façon incrémentale en deux étapes chacune correspondant à une version. Cette approche de développement incrémental a été mise en place afin de limiter les risques et de contrôler le processus de développement. La stratégie de « *versioning* » utilisée est basée sur l'identification, lors de la définition de l'architecture, des exigences satisfaites dans la version 1. Dans cette version, seules les demandes classifiées « obligatoires », ont été prises en compte. Les requêtes secondaires pour la VI ou non critiques, ainsi que les difficultés de développement rencontrées et portant sur des exigences non obligatoires, ont été traitées dans la version suivante.

La première version de nos services a été évaluée par un groupe d'ingénieurs du domaine spatial. Les résultats d'évaluation de la première version développée au milieu du projet ont été pris en considération lors du développement de la seconde version. La version 2 a apporté de nouvelles solutions et des améliorations au niveau de l'awareness (conscience du groupe) grâce à la gestion de la présence, et au niveau de la convivialité et de la maîtrise de la complexité des fenêtres de l'interface graphique, et ce pour chacun des modules de préparation et de gestion de session.

1.3 Plan de la thèse

Le manuscrit est organisé en six chapitres. Le chapitre 2 présente un triple objectif, d'abord il propose de définir la terminologie générale utilisée dans ce document, ensuite il essaie de dégager les caractéristiques de la coordination en générale et enfin il porte sur la caractérisation des fonctions de coordination nécessaires pour les sessions de coopération multi-applications. Le chapitre 3 présente d'abord les besoins fonctionnels des applications de gestion de sessions, suivi par la proposition d'un modèle de coordination des sessions qui est capable de spécifier les interdépendances entre les acteurs pendant une session coopérative. Le chapitre 4 présente la conception et la mise en œuvre de deux services pour la gestion des sessions de coopération multi-applications : le service de configuration des groupes et de préparation de sessions de collaboration appelé RMS et le service de gestion de sessions appelé SMS. Le chapitre 5 détaille le processus de validation des services RMS et SMS issus de cette thèse dans le cadre du projet européen DSE. Enfin, la conclusion et les perspectives de ce travail sont présentées dans le chapitre 6.

Chapitre 2. La problématique de coordination des actions dans la gestion des sessions coopératives : caractéristiques et besoins

2.1 Introduction

Ce chapitre vise à présenter les problèmes relatifs à la coordination des actions pour la gestion des sessions coopératives. On ne peut pas aborder de manière isolée ce type de problèmes, car ceux-ci sont dépendent des types d'applications exécutées pendant la session, du type et du nombre d'instances permises à un instant donné et de la définition des contraintes de coordination, c'est à dire les types de dépendances entre les actions effectuées par les acteurs de la session. Les travaux de cette thèse visent des sessions coopératives qui soutiennent les activités de l'ingénierie coopérative distribuée. Ces sessions impliquent des groupes de travail géographiquement distribués qui travaillent ensemble par des séquences d'interactions synchrones ou asynchrones, en utilisant un ensemble d'applications coopératives selon une planification définie à l'avance ou improvisée.

Ce chapitre est structuré en quatre parties :

Tout d'abord, comme il est dit ci-dessus, la première partie cherche à caractériser et définir le type d'applications utilisées par les sessions de coopération, à savoir, les applications coopératives distribuées. La seconde partie présente les problèmes de coordination rencontrés dans ce type d'applications et propose quelques définitions de termes de coordination, de coopération et de communication afin de les

distinguer dans cette thèse. La troisième partie traite de la coordination des sessions coopératives. Cette partie a un triple objectif : d'abord présenter la définition et la caractérisation de la gestion des sessions coopératives, ensuite exposer la problématique liée à la coordination des actions dans la gestion de sessions coopératives et finalement présenter un état de l'art sur les gestionnaires de sessions actuels. Enfin, la quatrième partie du chapitre est dédiée à notre propre travail et nous y présentons les problèmes de coordination des actions de gestion des sessions de coopération, en utilisant un cas d'application de l'ingénierie coopérative distribuée. Ce même scénario est utilisé pour la validation des nos travaux et amplement exposé dans le Chapitre 5 de cette thèse.

2.2 Les applications coopératives distribuées

Dans l'ample spectre du domaine de travail coopératif assisté par ordinateur (par la suite on utilisera le sigle TCAO) on trouve les applications coopératives distribuées. Elles relèvent de caractéristiques spécifiques qui les différencient des autres applications collecticielles. Jusqu'à aujourd'hui les notions de TCAO et de collecticiel n'ont pas toujours reçus de définition précise et consensuelle [Kli1991]. Il n'est pas donc facile de donner une réponse immédiate et précise à la question : que sont les applications coopératives distribuées ? Juste après, nous développerons le TCAO, le collecticiel et les applications coopératives distribuées, afin de donner une base aux termes utilisés dans cette thèse.

2.2.1 Le travail coopératif assisté par ordinateur

Le terme du travail coopératif assisté par ordinateur (Computer-Supported Cooperative Work ou CSCW en anglais) a été créé par Paul Cashman et Irene Grief en 1984 lors d'un forum dans lequel se sont réunis des spécialistes de différents domaines afin d'examiner comment les personnes travaillent en groupe et comment la technologie peut les aider [Gru1991]. Depuis la première conférence internationale du TCAO en 1986, des nombreuses conférences et forums se tiennent en Europe et aux Etats-Unis et le terme TCAO s'est banalisé dans le milieu de la recherche.

Le TCAO est un domaine fort prometteur, situé au carrefour des sciences humaines, des sciences informatiques et d'autres disciplines. En effet, le TCAO intègre aussi bien les domaines des systèmes répartis, des réseaux, des interfaces homme - machine et de l'intelligence artificielle d'une part que les domaines de la sociologie, de la psychologie, de l'ergonomie et de l'organisation du travail d'autre part.

2.2.2 Le collecticiel

Le terme collecticiel (groupware en anglais) a été créé par Peter et Trudy Johnson-Lens [Joh1982] en 1982 pour désigner l'ensemble des systèmes informatiques

et des processus sociaux des groupes assistés par le système. La définition de groupware de Peter et Trudy Johnson-Lens est résumée par la Figure 2-1.

GROUPWARE =
Intentional GROUP process and procedures to achieve specific purposes
+
softWARE tools for designed to support and facilitate the group's work

Figure 2-1 Définition du Groupware selon Johnson-Lens [Joh1982]

Plusieurs dictionnaires encyclopédiques en informatique [RRH2000, Hab1997] basent leur définition du collecticiel sur celle énoncée par Ellis, et. al. [EGR1991] : « un système à base d'ordinateurs qui supporte des groupes de personnes réalisant en commun une tâche ou un but et qui fournit une interface pour accéder à un environnement commun ».

2.2.3 Quelle est la différence entre TCAO et Collecticiel ?

Tandis que le collecticiel se réfère aux systèmes assistés par ordinateur réels, le TCAO est l'étude des outils et des techniques du collecticiel et de leurs effets psychologiques, sociaux et organisationnels.

Selon [TN1999] le but du TCAO est d'étudier comment les gens travaillent ensemble tant dans des petits groupes que dans de grandes organisations, comment les applications collecticiels influencent ces groupes et organisations et comment les applications collecticiels peuvent améliorer ou augmenter la communication entre les membres du groupe et la coordination à l'intérieur des organisations.

2.2.4 Quelle est la différence entre les collecticiels et les systèmes multi-utilisateur ?

Il y a une différence fondamentale entre les collecticiels et les systèmes multi-utilisateurs, e.g. les bases de données concurrentes. Le verrouillage des données dans une base de données veut donner à chaque utilisateur l'impression qu'il travaille seul sur une tâche individuelle. Il reste indépendant des autres et ignore leurs actions.

D'après [RRH2000] dans les systèmes collecticiels, il est fondamental au contraire de renforcer les processus du groupe et chaque utilisateur doit être conscient du contexte dans lequel ses tâches sont exécutées. Ceci implique la connaissance de l'existence des autres utilisateurs et de leurs actions.

2.2.5 Classification du collecticiel

Avant de développer les applications coopératives distribuées, il s'avère nécessaire d'identifier et classifier les principales caractéristiques des systèmes collecticiels. Une manière classique de classer les différents types de collecticiels est de les placer dans un tableau de deux dimensions : le lieu et le moment [Joha1988, EGR1991]. Cette convention est illustrée par le Tableau 2-1. Ainsi, les collecticiels sont des systèmes qui gèrent le travail en groupe de quatre manières :

1. Synchrones et coïncidant : même moment, même lieu.
2. Synchrones et déplacé : même moment, lieux différents.
3. Asynchrones et coïncidant : moments différents, même lieu.
4. Asynchrones et déplacé : moments différents, lieux différents.

		Même Lieu	Lieux Différents
		Face à face	Géographiquement distribuées
Même Moment	<i>Synchrone</i>	<ul style="list-style-type: none"> • PC et projecteur d'écran • PC à écran partagé • Salles électroniques • Outils de décision de groupe 	<ul style="list-style-type: none"> • Vidéoconférence (un-à-un, n-à-n) • Tableaux blancs • Partage d'applications
Moments Différents	<i>Asynchrone</i>	<ul style="list-style-type: none"> • Bases de données à accès concurrent • Systèmes de gestion de documents 	<ul style="list-style-type: none"> • Messagerie électronique • Systèmes workflow

Tableau 2-1 Classification des collecticiels

Cette classification espace-temps a le mérite de distinguer deux grandes classes de systèmes collecticiels : les synchrones et les asynchrones. Les collecticiels synchrones sont utilisés simultanément par l'ensemble d'utilisateurs afin d'effectuer les différentes tâches et sous-tâches d'un travail coopératif. Par opposition, les utilisateurs de collecticiels asynchrones réalisent ces tâches à des moments différents, ils interagissent avec le même système mais en moments différés. Dans chaque cas, les utilisateurs se trouvent à un même endroit ou en des endroits éloignés.

Grudin [Gru1995] affine ces dimensions en introduisant la notion d'incertitude ou d'imprévisibilité. Sur l'axe du temps, un travail coopératif peut s'accomplir au même moment, à des instants différents mais prévisibles, ou encore à des instants différents et imprévisibles. De même, le travail peut se faire dans le même lieu, en des endroits distincts mais prévisibles ou en des endroits distincts mais imprévisibles.

2.2.6 Les applications coopératives distribuées

Les applications coopératives distribuées (ACD) désignent un sous ensemble de systèmes collecticiels dont les caractéristiques principales sont la coopération et la distribution.

La caractéristique de coopération définit un ensemble d'applications dans lesquelles plusieurs utilisateurs interagissent. Le mode de collaboration peut être synchrone (simultané) ou asynchrone (différé). En effet, on ne peut pas regarder le processus de coopération comme une tâche qui se déroule complètement soit de façon asynchrone soit de façon synchrone. Plutôt, on adopte la définition de processus de coopération comme une succession d'activités asynchrones et synchrones [MD2002].

La distribution sert à situer la localisation des utilisateurs dans les ACD. La définition de distribution dans les ACD est différente de celle utilisée dans la dimension lieu présentée en 2.2.5. Dans ce cas la notion de lieu est associée à un emplacement géographique physique. Cela signifie que l'expression « même lieu » peut signifier la même salle de réunion ou le même bureau.

Dans le cas des ACDs, le lieu est un emplacement virtuel ou électronique. Cette définition considère que le lieu est un emplacement particulier dans un groupe virtuel, par exemple le poste de travail d'un participant dans une entreprise virtuelle. Ce qui peut renvoyer à une adresse physique, comme l'adresse IP d'un ordinateur sur Internet ou l'adresse d'une vidéoconférence multi-point.

En utilisant cette définition de lieu, on peut caractériser les ACD comme des applications collecticiels dans lesquelles chaque utilisateur est associé à une ou plusieurs adresses électroniques. La coopération est réalisée indépendamment de la distance de séparation entre les utilisateurs. La distance peut aller de quelques mètres (le même bureau, le bureau d'à côté) jusqu'à l'échelle planétaire (au bout du monde).

2.3 La coordination et les applications coopératives distribuées

2.3.1 Coordination, coopération et communication

Dans les modèles de TCAO et de collecticiel on trouve systématiquement les notions de coopération, coordination et communication. Mais il est difficile de donner une définition exacte de ces trois termes qui sont très liés entre eux car chaque auteur donne sa propre définition selon le domaine ou le niveau d'application. Kling [Kli1991] avait soulevé la difficulté de définir ces termes, il a identifié aussi la complexité entre les relations et la polyvalence dans la définition de ce qu'il appelle les « mots c » : coopération, conflit, convivialité, compétition, collaboration, contrôle, contrainte, coordination et combat. Avant de présenter les problèmes liés à la coordination dans les applications coopératives distribuées, on essaie de donner une définition à ces termes, toujours dans le contexte des ACD et en réalisant une synthèse des définitions trouvées dans la littérature du TCAO et du collecticiel.

2.3.1.1 Communication.

Parmi ces trois termes, celui-ci est le plus facile à définir car il se rapporte principalement à la transmission de données. Les modèles de coordination de Malone [MC1990, MC1994], de Klein [Kle1998] ainsi que les modèles de collectif de Salber [SC+1995] et d'Ellis [EGR1991, EW1994, Ell1998] s'accordent sur l'identification de ce premier composant fondamental. L'étude de ces modèles montre que la spécificité principale de la communication est la capacité d'échange d'information entre les utilisateurs. Ainsi on peut définir la communication comme ce qui permet aux utilisateurs l'échange d'information.

2.3.1.2 Coopération.

Comme le rappelle [Mar1867], on trouve la première définition de coopération dans *Le Capital* de Marx en 1867. Il donne les définitions suivantes: « quand plusieurs travailleurs fonctionnent ensemble en vue d'un but commun dans le même procès de production ou dans des procès différents mais connexes, leur travail prend la forme coopérative » et « en général, des hommes ne peuvent pas travailler en commun sans être réunis. Leur rassemblement est la condition même de leur coopération. ». Dans ces définitions on trouve deux composants de la coopération : d'abord la notion de groupe et ensuite la création collective commune qui implique autant un but commun que l'existence d'interdépendances.

Selon Schmidt [Sch1991] la coopération peut être augmentative, intégrative ou débative :

La coopération augmentative sert à combler les capacités limitées de l'être humain. Elle permet d'augmenter les capacités cognitives de traitement de l'information et les capacités physiques des personnes.

La coopération intégrative concerne la coopération d'un ensemble de personnes ayant des spécialités différentes. Ce type de coopération permet l'intégration de compétences différentes qui permettent d'achever des tâches dans lesquelles est nécessaire l'intervention des différents spécialistes pour les réaliser.

La coopération débative est présente dans les situations dont il est nécessaire d'avoir différents points de vue pour l'achèvement d'une tâche.

L'identification de la coopération apparaît dans les travaux de De Michelis [DeMi1995] de Ellis [EW1994, Ell1998] et de Klein [Kle1998]. Dans tous ces travaux, il existe une définition semblable du terme coopération qui présente les caractéristiques communes suivantes : un ensemble d'acteurs, un objectif commun, un processus de travail dans lequel chaque acteur est conscient de la présence des autres et est tenu au courant de leurs travaux. A partir de ces caractéristiques on peut définir la coopération à l'égard des ACD, comme l'aspect qui permet aux participants la mise à jour en collaboration d'un ensemble de décisions ou d'actions partagées.

2.3.1.3 Coordination

Les modèles de Malone [MC1990, MC1994], de Salber [Sal+1995] de Klein [Kle1998] et de Ellis [Ell1998] s'accordent sur l'identification de ce terme. Dans l'ensemble de ces travaux, on distingue l'apparition des interdépendances entre les acteurs, les applications, les données et les activités. On définit la coordination, dans les ACD, comme ce qui assure que les actions de collaboration prises par les participants, basées sur un ensemble de décisions partagées (règles), soient coordonnées pour atteindre un objectif prédéfini de manière efficace.

2.3.1.4 La coopération, la coordination et les activités

On remarque que d'après les définitions de coopération et de coordination décrites ci-dessus, on retrouve dans les deux termes qu'un groupe de participants effectue ou est engagé dans une activité commune. Il faut donc faire la distinction dans cette relation afin de distinguer la caractérisation des actions réalisées au niveau coopération à celles réalisées au niveau coordination. Pour cela, on suit la définition de De Michelis [DeMi1995] :

La coopération arrive quand deux ou plusieurs personnes travaillent ensemble pour réaliser un but commun simple, aucune distinction ne peut être faite entre ce qui est fait de manière individuelle par n'importe lequel des participants ou ce qui est fait de manière collective.

La coordination arrive quand deux ou plusieurs personnes travaillent ensemble pour réaliser, chacune d'entre elles, différentes activités qui se fondent en une activité générale. Chaque participant a donc sa propre activité, mais toutes les activités n'en forment plus qu'une.

2.3.2 Gestion de problèmes de coordination dans les collecticiels

Dans les systèmes collecticiels, des conflits apparaissent inévitablement car plusieurs personnes travaillent sur une même application. Par exemple, deux personnes travaillant sur un tableau blanc partagé peuvent vouloir déplacer un même objet dans deux directions différentes. Dans ce cas il y a deux façons de régler le problème : laisser les utilisateurs le régler, ou mettre en œuvre dans l'application un mécanisme pour le régler. On parle dans le premier cas de protocole social et dans le second de protocole matériel.

Dans certaines applications synchrones, telles que la vidéoconférence, la responsabilité du contrôle de concurrence est souvent confiée aux protocoles sociaux où les utilisateurs ayant une interaction observent un protocole informel pour éviter de s'interrompre les uns les autres. Les protocoles sociaux sont parfois réalisés au niveau utilisateur par la notion de « rôles ». Par exemple, le président d'une conférence a le pouvoir de contrôler les activités concurrentes et de prendre le contrôle de la parole [LFK1988].

Ellis et al [EGR1991] relève la liste suivante de méthodes possibles pour la mise en œuvre de protocoles matériels: le verrouillage, les transactions, la circulation de jeton, le contrôleur centralisé, la détection de dépendances, l'exécution réversible et la transformation des opérations.

2.3.2.1 Verrouillage.

Ce protocole consiste à verrouiller les données avant leur modification. Le problème du verrouillage se pose dans toutes les applications collectives et particulièrement dans les éditeurs partagés. Si plusieurs utilisateurs peuvent accéder simultanément à un même document, il faut une méthode qui assure la cohérence, c'est à dire que chaque utilisateur ait bien le même document. Généralement, les éditeurs coopératifs divisent le texte en unités logiques (des fragments) auxquels sont appliqués des verrouillages [EGR1991] : pour chaque fragment, un nombre illimité de personnes a le droit de le lire mais il n'y en a qu'une qui a le droit d'écrire. Ces verrouillages posent cependant des problèmes, ils prennent beaucoup de temps, il est difficile de savoir à quel niveau les faire; faut-il verrouiller un caractère, un mot, une ligne, un paragraphe?. Si on a des verrous trop fins (caractère par exemple), le temps des verrouillages et déverrouillage devient prohibitif. Avec un grain trop fort, des utilisateurs risquent d'être inutilement bloqués parce que l'un d'entre eux accède à une portion du document. Choisir le bon niveau de verrouillage nécessite donc de connaître l'application.

Enfin un utilisateur qui ne rend pas le verrou (par exemple parce qu'il oublie de le faire) peut bloquer inutilement l'accès à une partie du document. Pour éviter ceci, le verrou peut être libéré après un certain temps, ou on peut donner aux autres utilisateurs une commande pour détruire le verrou [EGR1991], [GS1987].

2.3.2.2 Transactions.

C'est un modèle issu des bases de données. Une transaction est une unité de programme qui accède et met à jour (potentiellement) des variables. Au cours d'une transaction, toutes les variables concernées sont lues exactement une fois et sont écrites au plus une fois. On impose que si les données (qui représentent l'état de l'application) sont cohérentes avant la transaction, elles le sont encore après (mais peuvent ne pas l'être pendant). Le principal défaut de ce mécanisme est d'augmenter le temps de réponse de façon potentiellement inacceptable pour l'utilisateur, spécialement dans le cas du collectif synchrone.

2.3.2.3 Floor control.

Le droit de parole est utilisé plutôt par des interactions que pour des transactions. Avec le mécanisme des *floors* un seul utilisateur a la main. Si un autre utilisateur la veut, il doit la prendre ou la demander et attendre qu'on la lui donne. La

main peut être retirée automatiquement à quelqu'un qui la détient depuis trop longtemps.

Le principal inconvénient de ce système est évidemment qu'à un instant donné, un seul utilisateur a la main. Cependant introduire plusieurs « mains » permet d'obtenir une certaine souplesse (une par application partagée ou une par fenêtre). On peut également autoriser plusieurs personnes à avoir la main.

2.3.2.4 Contrôleur centralisé.

Cette solution correspond à l'architecture centralisée où les données du processus de contrôle central sont répliquées dans tous les sites. Toutes les opérations sont effectuées séquentiellement par un processus unique. Le contrôleur central reçoit des requêtes d'opération des sites et envoie les réponses à tous les sites. Ce modèle a deux avantages : il est simple à mettre en œuvre et il est particulièrement commode dans l'approche client/serveur. Les inconvénients sont : augmentation du temps de réponse, goulet d'étranglement et manque de fiabilité car une panne du contrôleur central entraîne l'arrêt de tout le système.

2.3.2.5 Détection de dépendances.

La détection de dépendances utilise des estampilles pour détecter des opérations en conflit que sont ensuite résolues de façon manuelle. L'absence de synchronisation est le principal avantage de cette solution car les opérations sans conflit sont effectuées immédiatement dès leur arrivée et le temps de réponse est acceptable. L'inconvénient vient de la participation de l'utilisateur pour assurer la cohérence de données car ceci entraîne la vulnérabilité du système aux erreurs de l'utilisateur.

2.3.2.6 Exécution réversible.

Il s'agit d'effectuer immédiatement les opérations. Si une opération entre en conflit avec une des opérations précédentes, les dernières opérations sont défaites jusqu'au conflit celui-ci est résolu, puis on refait toutes les opérations qui ont été défaites. La mise en œuvre de cette technique est complexe. Il faut être capable de défaire toute opération. Une pile est nécessaire et il peut être difficile, voire impossible, de borner sa taille.

2.3.2.7 Transformation des opérations.

Cette technique revient à détecter des dépendances et à les résoudre automatiquement plutôt que manuellement. Lorsque deux opérations sont

conflictuelles, on s'efforce de les composer pour obtenir le résultat voulu. Cette technique est évidemment très difficile à mettre en œuvre.

2.4 La coordination de sessions dans les applications coopératives distribuées

L'interaction de participants utilisant des collecticiels est effectuée lors de sessions. La session est le moyen capable de réunir participants et applications afin de parvenir à un but précis.

2.4.1 Définition de session coopérative

La caractéristique primaire des applications coopératives distribuées est, par définition, l'implication de l'interaction de multiples utilisateurs. La manière dont les utilisateurs d'une application s'associent ensemble est déterminée par la définition et la gestion des sessions coopératives.

D'après la définition de Drira et al [DMN+2001], une session coopérative est un ensemble de participants qui sont connectés à partir de plusieurs sites et qui travaillent ensemble sur un espace partagé de communication. Cet espace peut être un ensemble de données partagées ou bien un canal commun de communication. Le travail effectué par les participants est réalisé par l'utilisation d'un ensemble d'applications du type collecticiel.

Ainsi, on peut donner la définition suivante, une session coopérative est un ensemble de participants qui agissent simultanément et normalement depuis des points d'accès distribués sur des objets partagés en suivant des règles de coopération (implicites ou explicites) et en utilisant un ensemble d'applications qui leur permettent de progresser de façon coordonnée.

2.4.2 Formes de sessions coopératives

Edwards [Edw1994] a réalisé une classification des sessions fondée sur le degré de formalité des sessions. Il a défini les sessions explicites et les sessions implicites. Le premier type désigne des sessions qui sont planifiées, qui ont un nom et une durée plus ou moins longue. Par contre, le deuxième type se rapporte à des sessions spontanées, sans nom et de durée relativement courte.

2.4.2.1 Le modèle explicite

La plupart des applications de gestion de sessions suivent ce modèle. Ce modèle est caractérisé par deux approches:

- L'approche basée sur l'initiateur : dans ce cas, par un certain ordre des dialogues, l'utilisateur initiateur invite d'autres utilisateurs à une session coopérative. Le nombre d'invitations publiées peut être potentiellement important, selon l'application et le contexte de l'activité. Les utilisateurs invités

peuvent accepter ou rejeter l'invitation. Les exemples de cet approche incluent MMConf [CMB+1990] et RTCAL [GS1987]. Les gestionnaires basés sur l'initiateur ont deux buts: d'abord informer les utilisateurs de l'existence de la collaboration et deuxièmement fournir les moyens de rendez-vous entre les utilisateurs.

- L'approche basée sur la connexion : dans ce cas, l'utilisateur initiateur a déjà créé une nouvelle session. Alors les utilisateurs doivent la trouver en naviguant sur une liste de sessions actives (ou ils doivent savoir a priori que la session aura lieu). Une fois qu'ils connaissent l'existence d'une session, ils peuvent essayer de s'y joindre. Ce type d'approche fournit en général seulement les mécanismes du rendez-vous aux utilisateurs. Les participants emploient d'autres moyens pour annoncer qu'une session coopérative est en marche.

La gestion explicite de session semble être utile dans les situations où il y a un degré élevé de formalité ou un nom normal pour l'activité. Une situation « réelle » analogue pourrait être, la « séance du conseil de l'école doctorale d'informatique et télécommunications, le lundi 7 avril 2003 à 10h30, salle du conseil ». L'activité est largement connue des participants, dans un endroit bien connu et incarne un degré élevé de formalité.

2.4.2.2 Le modèle implicite

Il y a un certain nombre de situations dans lesquelles une forme plus légère de gestion de session qui exige moins de frais généraux initiaux peut être utile. Par exemple dans beaucoup de situations, les invitations ne sont pas nécessaires. Les invitations explicites sur ordinateur ne sont pas nécessaires si la collaboration a lieu dans un petit groupe de personnes qui y participent et qui sont censées travailler à une certaine activité collective.

La principale différence entre le modèle explicite et l'implicite réside dans la manière d'effectuer le rendez-vous. En effet, dans le modèle explicite, le rendez-vous est réalisé par un mécanisme voisin du but principal de la session (navigation sur une liste de sessions avant de pouvoir se joindre à un éditeur partagé). Dans le cas du modèle implicite, le format du rendez-vous cherche une manière plus directe de réunir les utilisateurs potentiels. Le modèle implicite conjugue la réunion des utilisateurs et le but même de la coopération. Ainsi, pour éditer en collaboration un dossier, les utilisateurs éditeraient simplement le même dossier. Le système détecterait la potentialité d'une coopération puisque les utilisateurs multiples travaillent sur le même objet, sans avoir besoin d'appeler des sessions ou listes de sessions pour accomplir le rendez-vous. Contrairement aux formes explicites de gestion de session (l'approche basée sur l'initiateur et l'approche basée sur la connexion), cette forme de gestion implicite de session présente trois possibilités : orienté artefacts, orienté activités et basé sur la métaphore de lieu.

2.4.2.2.1 Classification de sessions coopératives implicites

Les sessions implicites peuvent être classifiées selon la structure utilisée pour réunir les participants et les applications. D'après [Lju1998], [RG1997] et [Edw1994] on en trouve trois catégories : les sessions basées sur des artefacts, celles basées sur des activités et finalement celles qui utilisent la métaphore de lieu.

Les sessions basées sur les artefacts désignent les modèles de gestion de session où les participants veulent travailler ensemble quand ils utilisent les mêmes artefacts, e.g. le même document.

Le modèle de sessions classifiées selon les activités apparaît dans les applications *workflow*. Ce modèle réunit les participants qui partagent la même activité.

La métaphore de lieu définit les gestionnaires de sessions basées sur des salles ou des espaces virtuels. Dans cette classification les utilisateurs trouvent d'autres participants et/ou applications dans le même espace virtuel.

2.4.3 Types de sessions coopératives

Les sessions coopératives peuvent contenir une ou plusieurs applications, chacune capable d'exécuter une ou plusieurs instances. Il s'avère donc nécessaire de typer les différents genres de sessions. Il existe deux axes pour classifier une session : le nombre d'applications différentes qui la composent et le nombre d'instances qui peuvent être lancées pour chaque application.

Pour le premier axe, on nomme sessions mono-type celles qui ne comportent qu'un type d'application. On appelle sessions multi-types celles qui sont composées par plusieurs applications de types différentes.

Pour le deuxième axe, on définit comme sessions mono-instance celles où chaque application contenue dans la session ne comporte qu'une instance. Par contre, on définit, sessions multiples, celles où chaque application de la session exécute plus d'une instance.

2.4.4 La gestion de sessions coopératives

La gestion de sessions comprend un ensemble de mécanismes capables de contrôler un groupe de sessions. Un gestionnaire de sessions permet aux utilisateurs le démarrage, l'arrêt, la navigation, l'affichage de l'état et la sélection d'une session [PHRM1990]. Il leur permet aussi de se joindre à une session ou de quitter la session.

2.4.5 Problèmes de coordination dans la gestion des sessions coopératives

Les problèmes de coordination lors de l'exécution de sessions coopératives, sont caractérisés selon le type de session envisagée.

Dans le cas des gestionnaires de sessions mono-instance et multiples mais mono-type, on retrouve les mêmes problèmes de coordination que ceux décrits dans 2.3.2. En effet, ce genre des gestionnaires correspond à la coordination des plusieurs

utilisateurs sur la même application, même si l'application comporte plusieurs instances (session multiple) car il n'existe pas de coordination entre les différentes instances. La coordination est effectuée à l'intérieur de chaque instance.

Par contre, pour les gestionnaires de sessions capables de supporter des sessions multi-types, on remarque l'apparition d'une nouvelle famille de problèmes de coordination entre les applications.

D'abord pour les sessions mono-instance multi-types, il apparaît un problème de coordination dû à l'hétérogénéité des applications. Il faut coordonner les actions entre des applications de type différentes.

Ensuite, on considère les sessions multiples multi-types, dans ce cas il peut exister deux scénarios, soit les instances d'un même type d'applications n'interagissent pas entre elles mais plutôt avec des instances de type différentes, soit les instances du même type interagissent entre elles et aussi avec des instances des autres types. Pour le premier cas de figure, on retombe sur le problème d'hétérogénéité des applications décrit ci-dessus. Par contre pour le deuxième cas, en plus du problème d'hétérogénéité, il faudra considérer que chaque instance se comporte comme une application différente (en identification et en type) et alors elle doit être coordonnée avec d'autres instances du même type ou bien de type différent.

2.4.6 Systèmes de gestion de session

2.4.6.1 CALTECH Infospheres System

Le système Infospheres¹ est un système développé en Java par l'Institut de Technologie de Californie (California Institute of Technology). Le système est basé sur la communication peer-to-peer (aucune commande de droit de parole) des processus disséminés à travers un réseau. Dans le système, les auteurs présentent deux unités compositionnelles, les dapplets [CR1997] (qui sont des objets communicants multi-thread) et les sessions (qui sont des groupes de dapplets composés). La notion de dapplet a été introduite afin de distinguer le processus d'une application coopérative distribuée et celui d'une application distribuée classique. Les dapplets se composent ensemble pour former des sessions distribuées. Une session est un réseau temporaire de dapplets. Chaque session a pour objectif la réalisation d'une tâche par exemple la définition de l'emploi du temps pour la réunion d'un groupe de personnes. Une session est dynamique. Les membres peuvent se joindre et se déconnecter pendant l'exécution. Une session se compose de plusieurs types différents de dapplets (sessions multi-types). Un premier processus (un dapplet initiateur) est associé à chaque session. Il est responsable pour créer une session en envoyant à tous les participants les adresses des ports pour connecter les dapplets. Il est responsable pour lier les dapplets afin de construire la session. Le dapplet initiateur commande la connexion et déconnexion des sessions par le démarrage et l'arrêt de dapplets. Après la réception de l'invitation pour participer à une session, un dapplet peut accepter et se connecter à la session ou bien il peut refuser l'invitation.

¹ <http://www.infospheres.caltech.edu/>

2.4.6.2 Doe2000

Le projet Collaboration Management² initié en 1997 par le département d'énergie des Etats-Unis est centré sur la définition et le développement d'un environnement sophistiqué de gestion de collaboration, qui sera intégré à des environnements de collaboration propriétaires et des systèmes workflow. Ce projet couvre le développement d'un gestionnaire de collaboration qui intègre et contrôle plusieurs applications dans un environnement coopératif en temps réel. Le directeur fournit la base pour développer des interfaces alternatives, telles que les utilisateurs pourraient joindre une session de collaboration en choisissant soit une liste de sessions actives, soit en se reliant à une application partagée.

Le prototype du système fournit un éventail d'applications de communication qui est capable de recevoir d'autres outils et ressources à distance. Pour démarrer ou pour se joindre à une session coopérative en utilisant le système, les utilisateurs cliquent simplement sur les boutons appropriés à la page web et l'ensemble d'applications est lancé sur leurs machines. Le directeur central de session et le serveur central sont cachés de la vue des utilisateurs. Le serveur central permet l'échange des différentes adresses IP et les numéros de ports entre les ordinateurs et configure les divers composants. Le système fournit les applications suivantes:

- Vidéoconférence.
- Tableau blanc partagé.
- Navigation coopérative (communication peer-to-peer).
- Chat.
- Partage d'écran.
- Cahier électronique partagé.
- Fichiers partagés.

2.4.6.3 HABANERO

NCSA Habanero³ [NCSA2001] est le résultat de l'effort de recherches du projet ISAAC financé par le programme de Collaboration Intelligent et de Visualisation (IC&V) du Bureau de Technologie de l'Information (ITO) de l'Agence de Projets de la Recherche Avancée (DARPA).

Le système Habanero fournit le support nécessaire pour créer des environnements de collaboration de travail et des communautés virtuelles [CGJ+1998]. L'environnement inclut un serveur qui héberge les sessions et un client qui connecte le client avec les sessions en utilisant des applications appelées Hablets. Le client de Habanero fournit l'interface pour agir sur une session avec les capacités de définition, énumération, création, connexion et déconnexion. Le client fournit les informations de la session, l'identification de l'utilisateur, un mécanisme de notification, des

² <http://www-unix.mcs.anl.gov/DOE2000/>

³ <http://www.isrl.uiuc.edu/isaac/Habanero/>

possibilités d'enregistrement et de lecture, des options de sécurité, la liste d'utilisateurs et d'applications actives, un carnet d'adresses et des possibilités pour créer facilement la préconfiguration d'une session. Le client a deux interfaces utilisateur: la première est employée pour prédéfinir des sessions off-line la deuxième est employé pour agir avec les sessions actives en ligne en temps réel.

Habanero fournit un environnement de collaboration pour soutenir le processus de gestion de session. Cet environnement permet à chaque participant de créer, de se joindre à, de quitter et de parcourir la liste de sessions.

Une session Habanero est définie comme une réunion où un utilisateur peut aller et se joindre à d'autres participants. Un utilisateur peut lancer une réunion. Dans une réunion plusieurs applications de types différents peuvent être ouvertes et elles fonctionnent en mode de collaboration. L'environnement de Habanero supporte des sessions multiples.

Habanero est conçu selon l'architecture client/serveur. Le serveur enregistre l'état de la session et diffuse les événements vers les clients. D'abord Habanero effectue la réplication des applications à tous les clients ensuite, le système diffuse les changements effectués sur les états vers tous les clients. Après la connexion d'un nouveau client, celui-ci est informé de la liste des applications qui sont exécutées dans la session. Habanero garantit que tous les clients ont la même vue de l'état de la session et que les événements sont reçus dans le même ordre par tous les clients. Ceci garantit le même fonctionnement dans les applications chez chaque client.

Le gestionnaire de sessions Habanero fournit les caractéristiques suivantes :

- Un participant peut collaborer à plusieurs sessions différentes concurremment.
- Des applications peuvent être facilement ajoutées à l'environnement Habanero.
- L'information détaillée sur une session comme son nom, sa planification, l'ordre du jour, la liste des applications de collaboration actuelles et la liste de participants acceptés. Il soutient aussi l'anonymat des participants de la session.

Il est possible d'effectuer la préconfiguration de sessions afin de permettre aux utilisateurs de décrire une session une fois et enregistrer cette information pour la réutilisation postérieure.

Un mécanisme de notification est utilisé pour inviter les participants à se joindre à une session en ligne. Le système supporte des mécanismes de notification synchrone (une fenêtre surgissent sur l'exposition de la personne invitée) et asynchrone (un message électronique).

L'initiateur d'une session a un rôle spécifique dans la définition des caractéristiques d'une session en déterminant le groupe de collaborateurs qui ont le droit de se joindre à la session. Il définit aussi le processus de notification et le mécanisme d'identification. Les participants peuvent jouer des rôles comme participant actif, observateur passif ou d'autres rôles décrits par des applications spécifiques.

2.4.6.4 InVerse

Le système InVerse développé par IBM et codé en Java fournit une infrastructure commune pour déployer une variété d'applications de collaboration basé sur Internet [SNN+1997]. InVerse est une plate-forme qui soutient des applications

interactives en temps réel (telles que l'audio, la vidéo, le texte et le mouvement) sur l'Internet. Il fournit une plate-forme commune pour disséminer et contrôler multiples flux de données en temps réel (streams). Le système InVerse maximise le scalabilité en utilisant une architecture hybride de communication qui s'adapte automatiquement selon la largeur de bande passante disponible du réseau, la latence observée du réseau, les mesures de sécurité installée sur le réseau et les services disponibles comme le multicast.

Un utilisateur peut définir un ensemble de récepteurs pour un segment particulier de données venant d'une application. Cet ensemble est encapsulé dans un objet de groupe. Le groupe peut représenter une des options suivantes:

- Individu: un utilisateur simple.
- Proximité: tous les utilisateurs à qui l'expéditeur est « visible ».
- Groupe Privé: une liste de distribution créée localement et seulement visible au créateur.
- Groupe Public: un groupe d'utilisateurs contrôlé par un serveur.
- Broadcast : tous les utilisateurs enregistrés.

Le terme groupe est semblable à la notion d'une session. Néanmoins, le groupe InVerse offre des possibilités pour diviser des utilisateurs en sous-groupes. En plus de la définition du type de groupe, qui est nécessaire pour transmettre des données, l'expéditeur doit indiquer le niveau de fiabilité (partiel ou total) pour transmettre les données. La couche de communication d'InVerse est responsable pour déterminer la manière de transporter les données à la destination pour assurer le type de transport exigé.

InVerse contient une interface utilisateur graphique qui sert d'outil de gestion de sessions. A travers son utilisation, un membre de groupe effectue les actions suivantes:

- Localisation d'autres utilisateurs enregistrés.
- Obtention des informations sur les sessions publiques en envoyant une question au serveur.
- Possibilité de joindre les groupes publics.
- Création de groupes privés.

Le système ne possède pas de contrôle de droit de parole, mais la création d'un groupe privé assure que l'information est envoyée dans un sens unique. En conclusion, InVerse soutient seulement des sessions mono-type, mais avec une ou plusieurs instances, c.-à-d. des sessions multiples sont permises. Pour créer une session, un groupe est enregistré dans le serveur et celui-ci définit la manière (fiable ou pas fiable) de transmettre les données. Un utilisateur peut se joindre ou quitter une session en envoyant une requête au serveur. La fonction d'ouverture et de fermeture des applications à distance n'est pas envisagé par InVerse. Il n'y a aucun envoi automatique des informations de mise à jour sur l'état de session aux utilisateurs. Cette information peut être envoyée à un utilisateur après la réception d'une requête par le serveur. InVerse n'enregistre ni l'état ni la définition d'une session.

2.4.6.5 Java Collaborative Environment

Le Java Collaborative Environment (JCE)⁴ [AKK+1996] de l'Université Old Dominion et de National Institute of Standards and Technology (NIST) définit le terme session comme un ensemble d'applications qui échangent de l'information, c.-à-d. qui travaillent en mode collaboratif. Néanmoins, JCE ne supporte pas des sessions multiples. JCE ne permet que la création de sessions qui contiennent un seul type d'applications [AKK+1998], c.-à-d. JCE exécute des sessions mono-instance mono-type.

JCE emploie des applets en tant que modules de collaboration. Pour fournir le partage des événements et des données, le système de contrôle de fenêtres de Java (Java's Abstract Window Toolkit AWT) et le kit de développement de Java (Java Developers Kit JDK) sont remplacés par un kit de collaboration propriétaire [AKK+1997].

La gestion de sessions en JCE est réalisée par une application nommée le directeur de collaboration [KKF+1997, AKK+1999]. L'ensemble des actions possibles se compose de:

- La possibilité de joindre/quitter une session, ceci se réalise par le lancement/arrêt d'une application.
- Le système est capable de montrer la liste de sessions ouvertes ainsi que la liste de leurs participants.
- Le système permet à un participant la demande du droit de parole afin de pouvoir modifier l'espace des données de collaboration. Un utilisateur peut avoir plusieurs droits en même temps (un pour chaque session d'un type particulier).
- L'envoi des informations de mise à jour sur l'état des sessions aux utilisateurs est affiché dans la fenêtre du directeur de collaboration.
- Le directeur de collaboration de JCE n'est pas adaptable à moins que le code soit partiellement réécrit. Ne sont possibles ni l'enregistrement de l'état d'une session ou sa définition ni l'ouverture/fermeture des applications à distance.

2.4.6.6 TANGO

TANGO⁵ est une plate-forme d'intégration qui permet l'exécution des environnements de collaboration basés sur la technologie Internet. Le système fournit les moyens d'intégration rapide des applications Web et non Web dans un environnement de collaboration multi-utilisateur [BCF+1997]. TANGO a été développé au Northeast Parallel Architectures Center à l'université de Syracuse. Le système a été implanté en Java et utilise les paradigmes d'intégration Internet (applets, HTTP serveurs, Internet navigateurs).

Une session est un groupe d'instances d'applications fonctionnant simultanément en mode de collaboration [Jur1997]. Toutes les applications appartenant à la même session s'échangent de l'information et partagent le même

⁴ <http://snad.ncsl.nist.gov/madvtg/Java/Java.html>

⁵ <http://www.collabworx.com/legacy/tango/>

comportement. Le comportement en mode de collaboration d'une application dépend de ses caractéristiques particulières.

Chaque application appartient à une session, même si elle n'est pas actuellement employée pour la collaboration. Dans un tel cas la session se compose seulement de cette application.

Dans toutes les sessions il y a un utilisateur distingué qui est considéré comme un maître. Le maître de la session a des privilèges spéciaux pour commander l'accès au comportement et/ou de contrôle d'application par rapport aux autres utilisateurs de cette session. Les privilèges dépendent du type d'application. Les autres participants d'une session s'appellent les esclaves de la session parce qu'ils peuvent seulement observer les actions produites par le maître.

Une exception au mode collaboration maître-esclave est le fonctionnement en mode d'associé. Cela signifie que tous les participants ont les mêmes droits d'employer une telle application. Dans ce cas-ci le mécanisme maître-esclave est simplement transparent pour l'application.

L'état de sessions peut être changé. Pour produire et commander ces changements il y a un mécanisme mis en application dans TANGO, qui est appelé le mécanisme de gestion de session. Ce mécanisme implique:

Le changement d'état d'une session en produisant une action, appelée action de Session Management;

La présentation des informations précises et conformes sur toutes les sessions établies sur l'interface graphique utilisateur sur toutes les machines du système.

Les actions de gestion de session sont :

1. Créer une session : Une session est créée quand un participant lance une application locale. Au commencement, la session contient seulement cette instance de l'application. Le participant devient automatiquement le maître de cette session. D'autres participants peuvent alors se joindre à cette session.
2. Se joindre une session existante : Se joindre à une session existante signifie lancer une nouvelle application et la relier à une session.
3. Lancer une application à distance : Le maître d'une session peut relier un ou plusieurs utilisateurs de TANGO à la session en lançant une application du type approprié sur le serveur central à distance.
4. Quitter une session : Quand un participant quitte une session, il y a trois cas possibles qui ont besoin de considération:
 - Le participant n'est pas le maître de la session, dans ce cas-ci l'application est fermée et supprimée de la session.,
 - Le participant est le maître courant de la session mais il n'a pas créé la session, dans ce cas-ci l'application est arrêtée mais la session ne finit pas. Le mode maître est automatiquement transféré au créateur de la session.
 - Le participant est le maître courant et le créateur de la session, dans ce cas-ci la session est terminée. Tous les participants sont déconnectés de la session, les applications arrêtées et la session est supprimée.
5. Fermer une application à distance : Le maître d'une session peut démonter un des utilisateurs de TANGO de la session en clôturant une application sur le serveur central à distance.

Les deux actions suivantes peuvent être considérées comme des actions de droit de parole, parmi les actions de gestion de session. Ceci signifie qu'elles permettent de changer le propriétaire du mode maître.

6. Demander le mode maître : Un participant d'une session peut demander le mode maître durant une session. Cette demande doit être approuvée par le maître. Si elle est approuvée alors le maître précédent devient automatiquement un esclave en cette session. Si la demande n'est pas approuvée alors l'état de la session ne change pas.
7. Attribuer le mode maître : Le maître d'une session peut céder ce mode à un autre utilisateur. Il n'y a aucun procédé d'acceptation dans ce cas-ci. Après l'accomplissement de cette opération le maître précédent devient automatiquement un esclave en session. L'état de la session est toujours changé.

2.4.7 Standards et recommandations pour la gestion de sessions

2.4.7.1 Session Description Protocol

Le MBONE [Eri1994] est la partie de l'Internet qui soutient l'IP multicast et permet ainsi la communication multiple efficace. Il est employé intensivement pour la communication de conférences multimédias. Dans ce type de conférences, la propriété de coordination d'appartenance à une conférence n'est pas nécessaire. En fait, pour recevoir une conférence, un utilisateur MBONE doit seulement connaître l'adresse multicast du groupe de la conférence et le ports UDP pour les flux de données de la conférence.

Les annuaires de sessions assistent l'annonce de sessions de conférence et communiquent l'information de configuration des sessions aux participants éventuels. Le protocole de description de session (SDP) est conçu pour donner une telle information aux destinataires[HJ1998]. Le SDP est purement un format pour la description de session et n'incorpore pas un protocole de transport. Le SDP est prévu pour employer différents protocoles de transport qui incluent le protocole d'annonce de session (SAP), le protocole d'initialisation de session (SIP), le protocole de gestion des flux temps réel (RTSP), le protocole du courrier électronique MIME étendu et le protocole de transport d'hypertexte.

Le but du SDP est de fournir une description de session aux destinataires, afin de donner les informations sur les flux multimédias qui composent la session. Le SDP est principalement prévu pour l'usage à l'intérieur d'un réseau, bien qu'il soit suffisamment général pour décrire les conférences dans d'autres environnements de réseau.

Une session de multimédia, dans ces buts, est définie comme ensemble de flux de médias qui existent pour une durée déterminée de temps. La fenêtre de temps de la session peut être discontinue.

Le SDP inclut:

- Le nom et but de la session,
- le temps d'activité de la session,

- les médias qui composent la session,
- l'information pour pouvoir recevoir ces médias (adresses, ports, etc.),
- des informations sur le débit à prévoir pour la conférence et
- les coordonnées électroniques de la personne chargée de la session.

En général, le SDP doit donner l'information suffisante pour pouvoir se joindre à une session (à l'exception des possibles clefs de chiffrement) et à annoncer les ressources à employer.

2.4.7.2 Session Initiation Protocol

Le protocole de déclenchement de session (SIP) [HSS+1999, RSC+2002] est un protocole de contrôle dans la couche application qui peut établir, modifier et terminer des sessions multimédias ou des appels téléphoniques. Ces sessions multimédias incluent les conférences multimédias, le télé-enseignement, la téléphonie sur Internet, etc. SIP peut inviter des personnes et des services automatisés, par exemple un service de stockage de médias. Le SIP peut inviter des correspondants aux sessions unicast et multicast. L'initiateur ne doit pas nécessairement être un membre de la session à laquelle il invite d'autres participants.

Le SIP peut être employé pour lancer des sessions aussi bien que pour inviter des participants à des sessions qui ont été annoncées et établies par d'autres moyens. Des sessions peuvent être annoncées en utilisant des protocoles multicast tels que SAP, le courrier électronique, des forums de discussion, des pages web ou des annuaires électroniques (LDAP). Le SIP soutient cinq facettes pour établir et pour terminer des communications multimédias :

- Localisation de l'utilisateur: détermination du système à employer pour communiquer.
- Capacités de l'utilisateur: détermination des médias et des paramètres de médias à employer.
- Disponibilité de l'utilisateur: détermination de la disponibilité du correspondant pour participer aux communications.
- Paramétrisation de l'appel: définition des paramètres d'appel chez le récepteur et chez l'émetteur.
- Manipulation de l'appel: y compris le transfert et l'arrêt des appels.

Le SIP est conçu en tant qu'élément de contrôle et des données globales de multimédia d'IETF dans une architecture composée par différents protocoles tels que RSVP (RFC 2205 [BZB+1997]) pour réserver des ressources de réseau, le protocole de transport en temps réel (RTP) (RFC 1889 [SCF+1996]) pour le transport des données temps réel et de fournir la rétroaction de la qualité de service, le protocole de gestion des flux temps réel (RTSP) (RFC 2326 [SLR1998]) pour le contrôle de la livraison des médias du type flux, le protocole d'annonce de session (SAP) [Han1996] pour la publication des sessions multimédia par l'intermédiaire de multicast et le protocole de description de session (SDP) (RFC 2327 [HJ1998]) pour décrire des sessions multimédias. Cependant, les fonctionnalités et les opérations de SIP ne dépendent d'aucun de ces protocoles.

SIP n'offre pas des services de contrôle de conférence tels que le droit de parole ou le vote et ne prescrit pas comment une conférence doit être coordonnée.

2.4.7.3 Session Announcement Protocol

Le protocole de notification de sessions (SAP) [Han1996] a été créé afin d'assister à la publication de conférences multicast multimédias et d'autres sessions multicast et pour la diffusion des informations relatives aux sessions vers les participants potentiels. Le processus de notification est le suivant, SAP envoie périodiquement un paquet de notification à une adresse multicast pré-définie contenant la description de la session. L'annonce est de type multicast et a la même portée que la session annoncée, ceci assure que les destinataires de l'annonce satisfont les contraintes décrites dans l'annonce (débit, QoS). Ces annonces sont reçues par des SAP listeners, ainsi les participants potentiels peuvent employer la description de session pour mettre en marche les applications requises pour participer à la session.

La Figure 2-2 [HCB1996] montre l'architecture des protocoles pour la

Contrôle de conférence	Audio	Vidéo	Partage d'outils	Annuaire de sessions			
				SDP			
RSVP	RTP et RTCP			SAP	SIP	HTTP	SMTP
Transport non fiable (UDP)					Transport fiable (TCP)		
Couche réseau (IP)							

réalisation des conférences multimédias sur Internet.

Figure 2-2 Pile des protocoles pour des conférences multimédia sur Internet [HCB1996]

2.4.7.4 ITU 120

La série de recommandations ITU T.120 [ITU1995] a pour objectif de normaliser l'architecture qui incorpore également la fonctionnalité de gestion des groupes et des sessions. Les bases de l'infrastructure T.120 sont les protocoles de transport spécifiés dans T.123, qui à l'heure actuelle supporte le transfert de données en utilisant des réseaux numériques à intégration de services (RNIS), des réseaux numériques à circuits commutés (CSDN), des réseaux numériques à circuits publics (PSDN) et des réseaux téléphoniques commutés (PSTN). L'extension pour inclure des réseaux à bande large est à l'étude. La recommandation des protocoles de transport T.123 est employée pour le service de communications multipoint T.122/T.125, qui définit un service indépendant du réseau avec des modes de transfert de données flexibles (émission broadcast et requête/réponse), adressage multipoint (un à tous, un au sous-groupe et un à un) et cheminement multipoint (recherche des chemins les plus courts à chaque récepteur). Puis la recommandation T.124 spécifie une commande de conférence générique qui emploie le service de communications multipoint T.122. Les services abstraits de contrôle de conférence incluent les services pour créer, lister, demander, se connecter, inviter, ajouter, fermer, ouvrir, se déconnecter, terminer,

éjecter un utilisateur et les services de transfert des sessions. Ces services fournissent un environnement performant pour mettre en application des vidéoconférences qui emploient alors les services T.124 et T.122. La Figure 2-3 montre la pile des protocoles qui appartient à la série T.120.

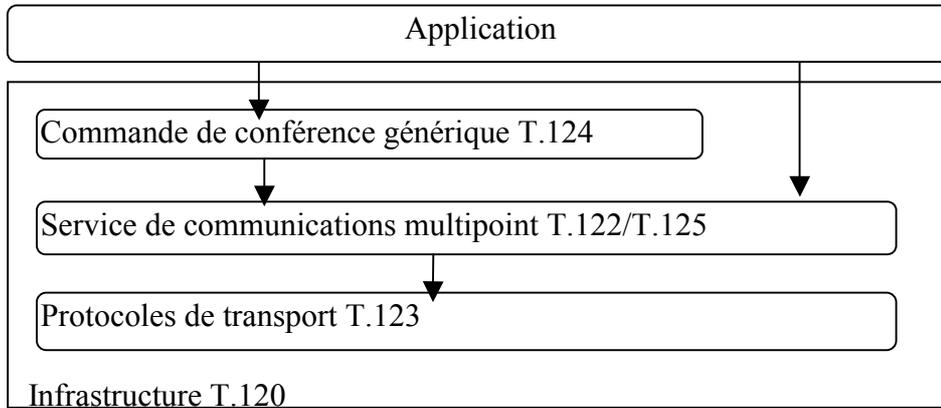


Figure 2-3 Architecture de l'infrastructure des recommandations de la série T.120 [ITU1995]

2.4.8 Modèles pour la gestion de sessions

2.4.8.1 CONCHA : Service de contrôle de conférences basé sur Java et sur le service d'événements CORBA

Orvalho et al [OAB1999, OAB1999a, OAFB1999] présentent un service de contrôle de conférences basé sur le service d'événements de CORBA. Le service a été implanté en Java supporte des communications multicast fiables pour le transfert de données et d'information de contrôle. CONCHA (CONference system based on java and corba event service CHannels) est composé de deux services : le service de contrôle de conférence et le service de communication d'applications mutipoint.

Le système de conférence se fonde sur une hiérarchie des nœuds, clients et serveurs, organisés dans une topologie d'arbre, supportée par des liens de JSDT. Les serveurs sont les fournisseurs des services de conférence, avec un nœud serveur supérieur et quelques serveurs proxy. Les applications de collaboration sont dans les nœuds clients et sont reliées à un ou plusieurs serveurs, ou bien logées dans un serveur. La racine de l'arbre est un serveur central qui contrôle les ressources de conférence - les sessions, les canaux et les jetons - et constitue une base de données des conférences. Les requêtes pour demander des ressources sont faites directement au nœud supérieur hiérarchique, qui à son tour les envoie vers le haut dans la chaîne jusqu'à l'arrivée au serveur supérieur. Pour éviter des temps de réponse élevés, les proxys ont la capacité d'allouer des ressources, car elles gardent une réplique de la base de données de ressources du serveur supérieur. Pour garantir la mise à jour, le serveur supérieur et les proxys partagent une voie de transmission multicast fiable.

Le service de contrôle de conférence implante un ensemble de fonctionnalités basiques pour la gestion de conférences basé sur la recommandation ITU 124.

2.4.8.2 Un système de gestion de groupes et de sessions pour des applications multimédias distribuées : GMS

Wilde et. al. présentent un modèle de définition et de gestion des groupes et des sessions pour des applications multimédias distribuées [WFK+1996]. Le modèle, appelé GMS, est composé des agents de l'utilisateur et des agents du système. Les agents de l'utilisateur sont les composants qui seront intégrés dans la plate-forme de communication de groupe. Les agents du système fonctionnent comme des annuaires distribués qui fournissent une base de données distribuée aux agents de l'utilisateur.

Le système GMS distingue trois classes d'éléments : les utilisateurs, les groupes et les sessions. Les utilisateurs sont les individus qui utilisent les applications, les groupes sont des ensembles d'utilisateurs, les sessions sont la représentation de collaborations de travail entre les utilisateurs distribués, chaque session est constituée d'un ou plusieurs flux de données. Le but du GMS est de créer un composant intégrable aux infrastructures de communication de groupe afin de les rendre capables de gérer et d'intégrer de sessions. La Figure 2-4 montre l'architecture d'un composant GMS dans une infrastructure de communication de groupe. Le composant GMS, appelé GUA, communique avec le composant de la sécurité et celui du gestion de ressources. On peut constater que le composant GUA fait partie de l'infrastructure de communication de groupe, i.e. il n'est pas possible aux programmeurs d'applications d'accéder directement au GUA. Cette approche a été choisie parce qu'un certain nombre d'opérations (particulièrement celle de se joindre à une session car elle implique la connexion de plusieurs flux de données) ne peuvent être complètement traitées à l'intérieur du GUA car elles ont besoin également de l'infrastructure de communication de groupe (par exemple pour effectuer un contrôle d'admission afin de savoir si les ressources locales et du réseau satisferont les contraintes de la session).

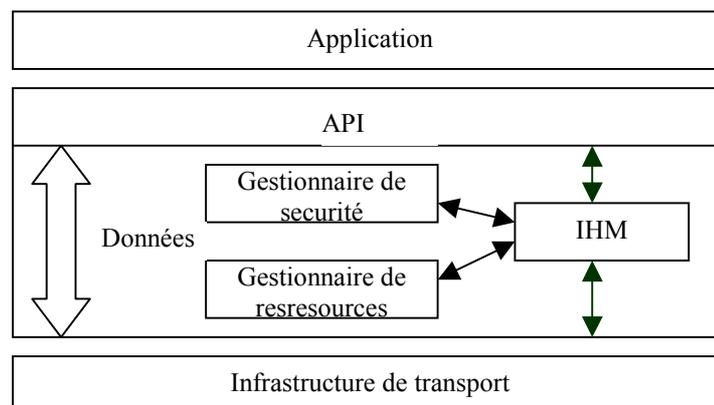


Figure 2-4 Gestion de groupe et de session dans un environnement de communication de groupe [WFK+1996]

En outre, le modèle de données GMS est conçu de manière à permettre la modélisation des utilisateurs, des groupes et des sessions d'une manière abstraite qui convient à différentes infrastructures de communication de groupe. La Figure 2-5 montre l'architecture du système GMS qui est basé sur un service d'annuaire distribué. Le noyau du système est composé d'un ensemble d'agents de contrôle du système (GSA), qui sont interconnectés en structure hiérarchique par le protocole GSP. L'organisation des GSA reflète la structure organisationnelle du monde réel. Le protocole GSP est un protocole multicast basé sur deux hypothèses, d'abord le domaine est relativement petit (pas plus de 10 GSA par domaine) et ensuite la hiérarchie est relativement plate (pas plus de 5 niveaux). L'échange de données utilisateur et des données de contrôle du système (protocole de données d'accès, GAP) est exécuté indépendamment. Ainsi il est possible d'employer des infrastructures de transport différentes pour l'échange de données et la connectivité du GAP.

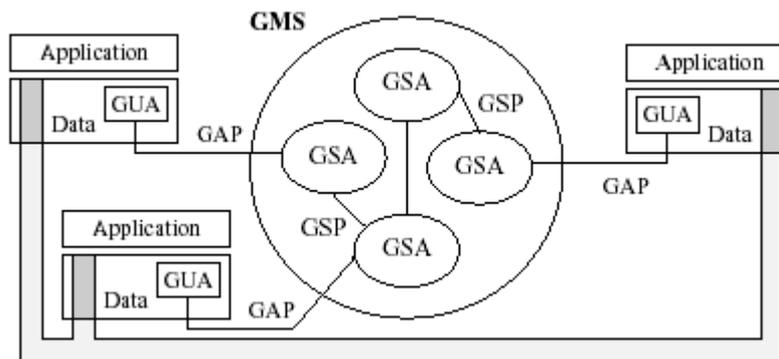


Figure 2-5 Architecture du système GMS qui comporte la gestion de groupes et de sessions pour des applications multimédias distribuées [WFK+1996]

2.4.8.3 Au-delà du mediaspace : un modèle pour la collaboration médiatisée

Roussel [Rou1997] relève l'inefficacité pour collaborer dans les systèmes mediaspace traditionnels [BHI1993] en raison de l'insuffisance des possibilités de coordination et d'action offertes aux utilisateurs. Il propose alors une nouvelle approche qui prend en compte ces besoins pour définir un nouveau modèle de mediaspace pour la collaboration médiatisée.

Le modèle proposé repose sur une approche multi-agents. Le terme d'agent étant fréquemment utilisé et afin d'éviter des confusions, est caractérisé par Roussel en quatre propriétés fondamentales :

- **Persistance** : un agent n'a pas de début ni de fin. Il est (ou semble être) tout le temps actif.
- **Capacité à communiquer** : un agent est capable de communiquer avec d'autres entités (autres agents, autres composants logiciels ou humains).
- **Autonomie** : un agent est responsable du choix des moyens utilisés pour satisfaire les requêtes qui lui sont adressées.

- Réactivité : un agent est capable de percevoir son environnement et de réagir aux changements.

Cette définition ne préjuge pas de la façon d'implanter un agent. En particulier, un agent peut être réalisé par plusieurs processus communicants. A chaque utilisateur est associé un agent appelé agent utilisateur. Un agent de liaison est chargé de gérer les sessions en cours et l'ensemble des ressources audiovisuelles (caméras, micros, écrans, haut-parleurs, magnétoscopes, projecteurs, etc.). C'est lui qui établit les connexions physiques entre ces ressources (voir Figure 2-6).

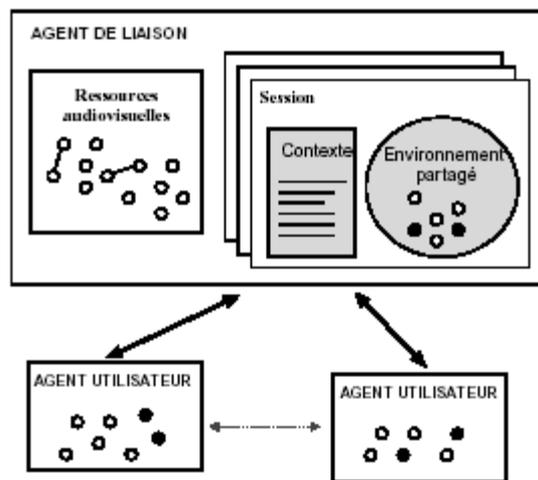


Figure 2-6 Agents utilisateurs et agent de liaison [Rou1997]

Chaque ressource est caractérisée par plusieurs attributs indiquant sa nature (audio ou vidéo), son type (entrée ou sortie) et ses particularités (micro directionnel, objectif grand-angle, caméra pilotable, etc.). Chaque agent utilisateur dispose d'un ensemble de ressources, qu'il a acquis auprès de l'agent de liaison. Cet ensemble correspond typiquement aux équipements situés dans le bureau de l'utilisateur. Un contrôle d'accès permet d'assurer qu'aucun autre agent utilisateur ne pourra acquérir ces ressources.

L'agent de liaison permet aux agents utilisateurs de créer des sessions, d'y entrer et d'en sortir. A chaque session sont associés un environnement partagé et un contexte. L'environnement partagé initial contient les ressources audiovisuelles mises en commun par les différents agents utilisateurs. Mais d'autres objets peuvent y être ajoutés par la suite, comme l'éditeur du scénario. Le contexte décrit les effets de l'entrée et de la sortie de session ainsi que les actions qui peuvent être effectuées par les agents utilisateurs. Ces actions peuvent ajouter ou retirer des objets à l'environnement partagé et établir des connexions entre les ressources. Elles déterminent également la façon dont la session sera fermée. Le degré d'intégration entre la session de collaboration médiatisée et celle d'une application partagée dépend des possibilités de communication entre celle-ci et les agents utilisateur.

Les services offerts par ce modèle dépendent donc du contexte dans lequel se déroule la session, des ressources apportées par les participants et du comportement

des participants (des actions exécutées par leur agent utilisateur). La combinaison de ces trois éléments crée la dynamique de coordination et d'action nécessaire à la collaboration médiatisée, le contexte fixant les limites de cette dynamique.

2.4.8.4 Gestion de sessions pour des applications de collaboration

Edwards [Edw1994] présente un modèle de gestion de sessions basé sur le partage d'information de l'utilisateur et des activités. L'information des activités est l'information qui contient les détails des tâches courantes, des tâches actives à travers le réseau: les utilisateurs connectés, les applications ou les tâches où ils sont actuellement engagés et les objets associés à ces tâches (c'est-à-dire, les données sur lesquelles les applications fonctionnent). Edwards définit une activité comme un triplet:

$$A_n = (U_n, T_n, O_n)$$

Où A_n représente la nième activité. Cette activité est composée de U_n , T_n et O_n qui représente le nième utilisateur, la nième tâche (application) et le nième objet (données) respectivement. Pour les buts de la gestion de session, U_n et T_n peuvent être considérés comme de simples jetons qui identifient uniquement l'utilisateur et l'application sur laquelle il travaille. Les relations entre les utilisateurs et les applications et leurs jetons U_n et T_n , sont de type un à un.

Les objets sont légèrement plus compliqués, car les applications peuvent manipuler différents domaines de données. Par exemple, un éditeur de texte manipule des documents, alors qu'un calendrier peut manipuler une base de données de rendez-vous. Chaque ensemble de données possède fondamentalement une sémantique différente vis-à-vis de l'utilisation et de la représentation.

Ainsi le jeton O_n se compose d'un identificateur de domaine et d'un nom valide à l'intérieur du domaine. Le domaine définit le type de données utilisées par l'application. Les exemples des domaines sont des répertoires, la sélection de bases de données et ainsi de suite. Chaque domaine définit un ensemble de noms qui ont une correspondance linéaire avec les objets qu'ils représentent. Par exemple, un fichier particulier est représenté par un nom unique dans le domaine du dossier.

Les applications qui souhaitent participer à la gestion implicite de session doivent éditer l'information d'activité de sorte qu'elle puisse être rendue évidente au service de gestion de session. En exécutant la gestion implicite de session, le service de gestion de session détectera automatiquement les situations de collaboration potentielles et prendra les actions « appropriées ». Il n'y a aucun besoin de la part des utilisateurs de publier explicitement des invitations ou de créer des sessions. Le service de gestion de session détecte les situations de collaboration potentielles en recherchant des chevauchements ou des confluences dans l'information publiée par les applications dans le réseau. Quand deux triples d'activité existent et contiennent la même marque d'objet (c'est-à-dire, quand le domaine et les noms de deux objets s'assortissent exactement), alors le service de gestion de session peut agir pour permettre aux utilisateurs d'entrer dans une situation de collaboration. Par exemple, si deux utilisateurs éditent le même dossier, le service de gestion de session peut informer les

utilisateurs de ce fait et les laisser entrer facilement dans l'état de collaboration. Les mécanismes pour joindre une collaboration sont assez proches de la dynamique humaine de collaboration. Quand deux collègues souhaitent travailler ensemble sur un document, ils expriment la décision en disant, « on se réunit après le déjeuner et on clôt le budget de cette période ». Aucune invitation formelle n'est publiée et aucun nom n'est donné à l'activité. Au lieu de cela, les collègues simplement commencent à travailler sur le budget à peu près à la même heure. L'action de travailler au même budget porte implicitement avec elle la notion de la collaboration. Puisque dans la collaboration explicite il n'existe pas de surcharge de travail pour la gestion de la session, en fait le système lui-même peut assumer la charge de détecter et de manipuler la collaboration potentielle. Il faut noter que cette forme de gestion implicite de session, par sa spécificité de transparence, exige que les applications ou l'environnement de support de collaboration ait des mécanismes puissants pour permettre aux utilisateurs de modifier dynamiquement le comportement du service de gestion de session. Evidemment, les utilisateurs ne veulent pas être jetés automatiquement dans un éditeur partagé lorsqu'ils viennent d'ouvrir un dossier qu'une autre personne a ouvert. Les mécanismes pour éditer et rechercher l'information d'une activité, pour produire des noms uniques d'objets dans un domaine et pour prendre une mesure appropriée lors de la détection d'une collaboration potentielle sont définis par des réalisations particulières de ce modèle.

2.5 Synthèse de nos contributions

Les travaux réalisés dans le cadre de cette thèse appartiennent au domaine de la conception et du développement des services de gestion de sessions coopératives distribués. Plus exactement, ils sont centrés sur la résolution des dysfonctionnements qui ont pour origine des problèmes d'inconsistance liés à l'absence de coordination des actions lors de l'exécution des sessions comportant différentes applications de types distincts et différents participants connectés à partir des sites distribués, c'est à dire des sessions de coopération multi-applications multi-utilisateurs, par la suite on appellera ce type de sessions : « sessions multi-applications ». Nous défendons que les causes des problèmes d'incohérence dans la gestion de sessions coopératives, notamment dans les applications d'ingénierie coopérative distribuée, sont liées au non respect des dépendances entre les actions effectuées par les différents acteurs (participants et applications) membres de la session.

En examinant l'état de l'art, nous avons constaté que les approches pour la gestion des sessions de coopération négligent la gestion des interdépendances entre les participants et les applications.

La coordination des actions dans les systèmes actuels de gestion de session se regroupent de deux types. D'un côté les systèmes « centrés sur les applications », qui se caractérisent par l'implantation des mécanismes de gestion de session dans les applications elles mêmes, c'est à dire que les applications ont été conçues pour fonctionner de manière coopérative avec plusieurs utilisateurs. Ces applications possèdent des mécanismes pour faire face aux problèmes de coordination décrits en 2.3.2. Parmi les systèmes étudiés, le système JCE pourrait appartenir à ce catégorie,

car, même s'il dispose d'un directeur de collaboration, une session est simplement la connexion des plusieurs utilisateurs en employant une même instance d'une application collaborative. D'un autre côté, il existe les systèmes «centrés sur l'environnement», dans ces systèmes, la gestion des sessions est une fonctionnalité inhérente à l'environnement dans lequel les applications sont pilotées. Un composant est spécialement implanté pour effectuer la gestion des sessions. Dans les systèmes étudiés, ce composant est capable d'effectuer plusieurs opérations de gestion telles que l'intégration de plusieurs applications de types distincts (CALTECH, Doe2000, InVerse, JCE, Tango), la gestion du droit de parole entre les utilisateurs (Habanero, InVerse, JCE, Tango,) ou bien la personnalisation du gestionnaire de sessions vis-à-vis de l'interface graphique, du regroupement des applications (Tango), mais ne tiennent pas compte des dépendances entre les actions.

Les standards et recommandations élaborés pour la gestion de session concentrent leurs efforts sur la définition des protocoles d'interopérabilité. Ils proposent des spécifications pour la description des sessions (SDP), l'invitation des participants aux sessions (SAIP) et l'initialisation des sessions (SIP). Ils proposent aussi des architectures pour supporter des conférences multicast multimédias (ITU120).

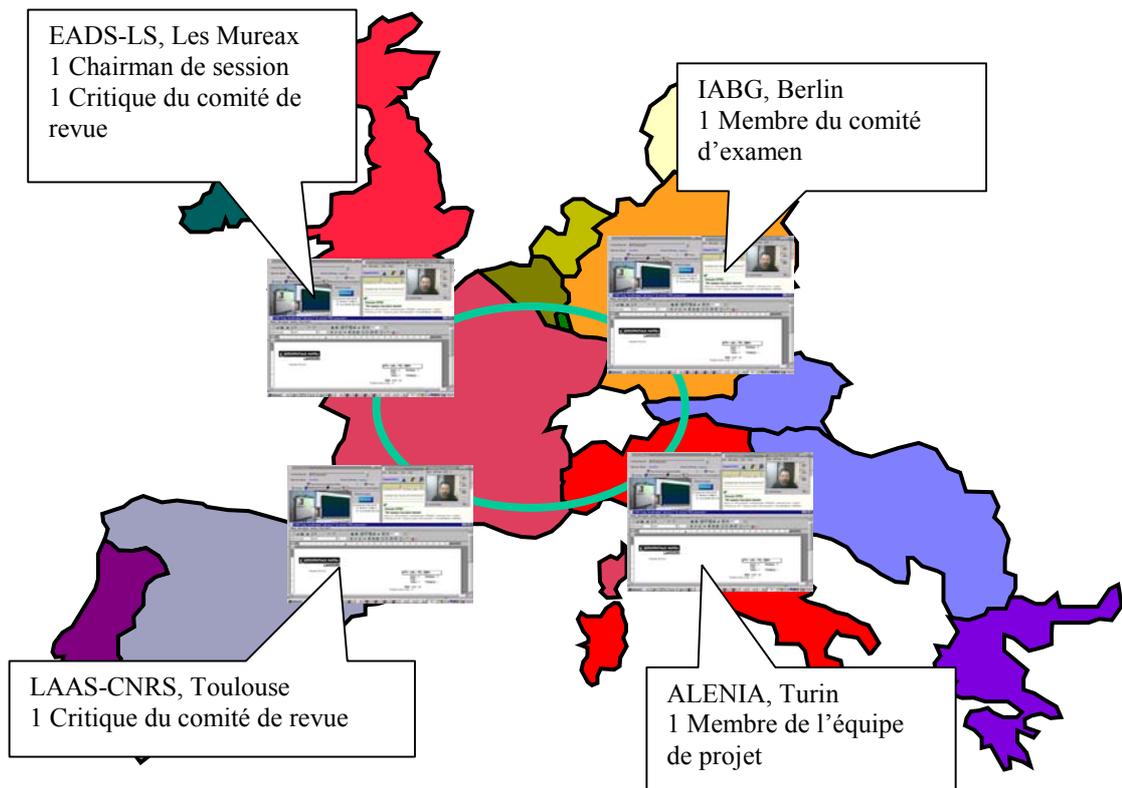
Les modèles de gestion sont les travaux le plus proches de notre problématique. Ils proposent des gestionnaires de sessions basés sur un modèle spécifique. Normalement, ils ciblent un ou plusieurs aspects de la gestion de sessions. Les modèles COCHA [OAB1999], GMS [WFKP1996] et le modèle meadiaspace [Rou1997] sont centrés sur l'intégration de plusieurs entités (agents ou composants) distribuées. Le modèle d'Edwards [Edw1994] est capable de représenter les liens entre les utilisateurs, leur tâches et les objets manipulées. Mais, il a pour objectif principal la détection de simultanéité dans l'accès et l'utilisation des objets partagés afin de créer des sessions implicites.

Notre travail a un double objectif, d'abord celui de proposer un modèle formel capable de spécifier les relations de dépendance lors l'exécution d'une session de coopération multi-applications et ensuite de proposer un service capable de soutenir la conception et la réalisation d'un gestionnaire pour ce type de sessions.

2.5.1 Application à un scénario de coordination en ingénierie coopérative distribuée

Nous analyserons un scénario de coopération issu de l'ingénierie coopérative distribuée pour montrer les différentes difficultés de coordination des actions rencontrées lors de l'exécution d'une session coopérative multi-applications. Ce scénario, appelé PDR (Preliminary Design Review) a été défini dans le cadre du projet DSE (Distributed Systems Engineering) [FPM+2000]. Le scénario PDR est une des revues de programme qui se produisent pendant le cycle de vie du projet ATV (Automated Transfer Vehicle). Ce scénario sera amplement décrit dans le Chapitre 5. Pendant la phase de revue du programme, les différents spécialistes, jouant différents rôles, travaillent ensemble afin d'analyser des documents de conception générés par les ingénieurs responsables de la conception de l'ATV. Les buts principaux du processus PDR sont de mettre en évidence des difficultés et les risques potentiels afin de les

réduire et de mettre les documents et les produits dans une base de données de référence. Trois types différents de groupes de personnes composent les groupes de revue PDR: 1) le comité d'examen -ils jouent le rôle des représentants du client et décident si les recommandations issues du PDR sont acceptées ou rejetées; 2) l'interface de l'équipe du projet -ses membres agissent en tant que voie de transmission entre les concepteurs et les critiques; 3) le comité de la revue -composé par différents groupes d'experts externes (critiques) au projet qui recommandent des actions et élaborent des documents d'anomalies RID (Review Item Discrepancy). L'exécution du processus PDR peut être composée d'une ou plusieurs sessions coopératives. Chaque session a un objectif précis vis-à-vis de l'ensemble des documents, par exemple la session de la révision des documents liés à l'aérodynamique de l'ATV. La Figure 2-7 montre l'architecture d'une session coopérative du processus d'exécution PDR. Cette session est composée par cinq participants qui ont les rôles suivantes : un chairman et un critique situés à Paris, un critique à Toulouse, un membre du comité d'examen à Turin et un membre de l'équipe de projet à Berlin. Ils coopèrent à travers trois applications : l'outil de partage d'applications pour distribuer les documents de conception ainsi que pour l'édition du document d'anomalies, l'application de vidéoconférence pour pouvoir discuter en temps réel et l'application de droit de parole pour pouvoir coordonner l'insertion et la modification du RID par les participants ainsi que pour éviter la cacophonie pendant la



discussion des participants.

Figure 2-7 Scénario de la revue préliminaire de conception du projet ATV

2.5.1.1 Les conflits liés aux dépendances inter-applications

Ce type de conflit concerne les dépendances entre applications de type différentes. Dans chacun des sites de la Figure 2-7, trois applications sont présentes : l'application de partage, l'application de vidéoconférence et l'application de droit de parole. Mais un ordre d'exécution est nécessaire pour leur exécution. En effet, l'application de droit de parole doit être lancée en dernier car elle interagit avec les deux autres. On suppose qu'on envoie un message aux trois sites pour démarrer en même temps les trois applications. On va supposer qu'à Toulouse, à cause de la configuration et des caractéristiques de l'ordinateur ou à cause du grand nombre d'applications extérieures à l'environnement DSE qui tournent en parallèle avec la session, l'outil de partage d'applications nécessite un temps de démarrage beaucoup plus long que sur les autres sites. Alors, tandis que les participants situés à Paris, à Turin et à Berlin constatent que les trois applications ont déjà démarré, le critique de Toulouse n'en voit que deux. Si dans cet intervalle, un message d'attribution de droit de parole sur l'application de partage d'application intervient, ceci provoquera un conflit à Toulouse, car l'application de droit de parole essaiera d'établir communication avec l'outil de partage d'applications, qui n'a pas encore démarrée.

2.5.1.2 Les conflits liés aux dépendances intra-applications

Ce type de conflits apparaît lorsqu'il existe des dépendances entre les composants qui intègrent une application d'un type particulier. Dans le cas du scénario PDR, l'outil de partage d'applications possède trois types de composants. Chaque type de composant est défini en fonction du rôle du participant qui l'utilise. Les trois types de composants sont : un ensemble d'un ou plusieurs composants serveurs, un composant proxy et un ensemble d'un ou plusieurs composants clients. Les composants serveurs sont utilisés par les participants de l'équipe de projet, le composant proxy est exécuté par le chairman de session et les composants clients sont exécutés par tous les critiques. Il existe des relations de dépendances entre ces trois types de composants et ils ne peuvent pas être démarrés en même temps. Les composants du type serveur doivent être les premiers démarrés car ils représentent les fournisseurs de base des applications à partager. Ensuite le composant proxy doit être mis en marche, il effectue les connexions vers les serveurs déjà démarrés. Finalement, c'est le tour des composants clients de démarrer, ceux-ci effectuent des connexions vers le proxy.

2.5.1.3 Les conflits liés aux dépendances entre participants

Les actions réalisées par les participants peuvent être aussi interdépendantes. En analysant le scénario PDR, on remarque que selon le rôle associé aux participants, ils doivent arriver à la session dans un ordre spécifique. Le chairman doit se connecter avant n'importe quel autre participant, à cause de deux contraintes. La première est

d'ordre technique, le chairman exécute sur son site des serveurs qui seront ensuite utilisés par les autres participants. Dans le cas spécifique du PDR le PC du chairman héberge le serveur de vidéoconférence multipoint. Une deuxième contrainte lui est imposée du point de vue organisationnel, car c'est le chairman qui préside la session. Le déroulement d'une session PDR n'est valide que si au moins un membre du comité d'examen est connecté, cette contrainte répond aux politiques de légalité de la révision.

2.5.1.4 Les conflits liés aux dépendances entre participants et applications

On peut aussi noter qu'il existe des liens entre les actions réalisées par les participants et les applications. Par exemple si la session PDR n'accepte pas de participants retardataires, alors l'action de connexion des participants est liée à l'action d'ouverture de la session. Un second exemple, il consiste à l'interdiction de connexion de participants non invités à la session. En effet l'action de connexion des participants dépend de l'invitation à participer venant du gestionnaire de session.

2.6 Conclusion

Dans ce chapitre nous avons développé la problématique liée à la gestion des sessions de coopération. Cette problématique dépend de deux facteurs, d'une part le type d'applications exécutées pendant la session d'autre part la définition de contraintes de coordination liées à l'enchaînements des actions. Dans les travaux de cette thèse, nous avons ciblé des sessions de coopération qui répondent aux problèmes issus de l'ingénierie coopérative. Pour répondre au premier facteur, nous avons identifié ce type de sessions comme multi-applications et multi-utilisateurs, c'est à dire qui supportent plusieurs types d'applications exécutées en parallèle par un ensemble d'utilisateurs distribués. Nous avons trouvé que ce type de sessions de coopération se trouvent dans l'éventail des applications coopératives distribuées. Nous avons aussi présenté les problèmes de coordination qui impliquent la conception et la réalisation de ce type d'applications. Pour répondre au deuxième facteur, nous avons défini et classifié les différents types de gestionnaire de sessions, de même nous avons présenté les différents systèmes et modèles actuels de gestion des sessions de coopération, ainsi que les standards et recommandations qui abordent le même sujet. Enfin, nous avons présenté la problématique de coordination des actions dans les sessions multi-applications, en utilisant un scénario de coopération issu de l'ingénierie coopérative distribuée.

L'étude des systèmes de gestion de session de coopération présenté dans ce chapitre provient de la recherche bibliographique de l'état de l'art que nous avons mené dans le cadre du projet DSE [DRN+2000]. Cet étude a été publiée par la suite comme ouvrage [DMN+2001].

A partir de ces études, le chapitre suivant va présenter le modèle formel développé pour la représentation des dépendances entre les actions pendant l'exécution de sessions coopérative multi-applications.

Chapitre 3. Formalisation de la gestion des sessions multi-applications

3.1 Introduction

Nous avons présenté dans le chapitre précédent les sessions de coopération comme des entités qui contiennent des participants, des applications et des données. La coopération permet aux participants de réaliser des objectifs qui ne sont pas à la portée d'une seule personne, en donnant la possibilité à chacune de profiter des connaissances des autres. Cela signifie que les activités des participants ne s'exécutent pas de manière isolée, mais au contraire interagissent au cours de leur exécution. Nous entendons par interactions, les différentes opérations que les acteurs, c'est à dire les participants et les applications, d'une session de coopération exécutent dans le but de réaliser leur objectif commun.

Les interactions doivent être pertinentes et constructives dans le sens où elles doivent se compléter pour faire converger le travail de chacun vers l'objectif de la coopération. Dans ce but, chaque acteur dans la coopération doit jouer un rôle bien déterminé et conforme à ses compétences ou ses fonctionnalités. Il est possible par l'attribution des rôles de répartir les activités et les responsabilités sur l'ensemble des acteurs.

Par ailleurs, si les interactions sont incohérentes et destructives, elles peuvent conduire à un comportement chaotique et nuisible de la session. Dans ce cas, les interactions au lieu d'aider les acteurs à atteindre leur objectif, elles provoquent une divergence entre le résultat de la coopération et l'objectif prédéfini. Par exemple, si un des participants agit sans prendre connaissance des actions des autres acteurs, il peut

de manière involontaire annuler leurs actions. Les interactions peuvent être donc destructives si elles ne sont pas coordonnées. En effet, le résultat des interactions dépend fortement de l'ordre dans lequel elle sont réalisées.

Pour qu'une session de coopération atteigne son objectif, il faut que les interactions respectent un ensemble de règles de coordination bien établies. Ces règles doivent être à la fois suffisamment restrictives pour éviter les interactions incohérentes et destructives et suffisamment permissives pour favoriser les interactions pertinentes et constructives.

Nous avons développé un modèle de coordination de sessions. Ce modèle est basé sur la spécification des interactions entre les acteurs pendant une session. Ce modèle permet la définition des règles de coordination lors de sessions de coopération. Le modèle est fondé sur deux aspects, d'une part par la représentation des sessions de coopération par des ordres partiels étiquetés. Et d'autre part, l'utilisation de formules de premier ordre afin d'exprimer les règles de coordination sur les sessions de coopération. Nous avons défini trois modalités, la *précédence*, l'*inhibition* et le *déclenchement*, qui sont suffisants pour décrire les dépendances dans les sessions de coopération. Ces trois modalités constituent la base de notre modèle.

L'organisation de ce chapitre est la suivante : la première section présente la description fonctionnelle des services impliqués dans la gestion des sessions de coopération, la deuxième section présente le modèle de coordination des sessions de coopération que nous proposons. Cette description est divisée en trois parties. D'abord, nous développerons la définition de session de coopération à l'aide des ordres partiels étiquetés. Puis, nous montrerons la définition des règles de coordination à l'aide de formules logiques. Enfin, nous présentons les trois modalités de base qui sont utilisées pour construire les règles de coordination. La deuxième section présente une application du modèle qui définit les règles de coordination de base pour les sessions de coopération. La troisième section présente une application du modèle de coordination dans le cas de l'ingénierie distribuée du domaine aérospatiale. Il s'agit de la modélisation de la revue préliminaire de conception PDR. Le scénario PDR appartient aux scénarios de validation du Project DSE. Le scénario PDR nous a servi comme guide pour tester l'application du modèle de coordination et par la suite aussi de tester sa conception et sa mise en œuvre. Enfin la quatrième section présente la définition des règles de coordination du scénario PDR.

3.2 Description fonctionnelle de la gestion des sessions

La gestion des sessions de coopération est composée de deux aspects, d'une part, la préparation hors coopération des sessions et d'autre part, la gestion en coopération de la session. La préparation des sessions prend à charge la définition et mise à jour des personnes, des équipes de collaboration et des sessions. La gestion en coopération de la session se charge de la coordination des actions pendant le déclenchement de l'activité de coopération. Nous avons défini deux services qui répondent aux besoins de ces deux aspects dans le cadre du projet DSE. Pour effectuer la préparation et configuration des sessions, nous avons élaboré le service «Responsibility and User Management Service (RMS)» et pour la gestion des sessions, le service «Session Management Service (SMS)».

Nous allons utiliser dans la suite le modèle UML¹[OMG2003] comme langage de spécification de ces services. Il a été choisi comme langage commun entre les différents partenaires impliqués dans le projet DSE. Le modèle UML résultant a permis la création d'une vue complète aux différents membres du projet, la réalisation d'un prototype rapide et il a facilité l'intégration des différents modules et services.

UML est à la fois une norme, un langage de modélisation objet, un support de communication et un cadre méthodologique. Il permet d'exprimer et d'élaborer des modèles objet indépendamment du langage de programmation. Il facilite également la représentation et la compréhension des solutions objet.

UML permet d'exprimer des modèles objet en faisant abstraction de leur implémentation. Cela signifie qu'ils sont valables quel que soit le langage de programmation. UML est un langage qui s'appuie sur un métamodèle qui définit ses éléments (les concepts utilisables) et leur sémantique. Ce dernier est lui-même décrit en UML, mais il est plus générique. Le métamodèle fait d'UML un langage formel possédant les caractéristiques suivantes :

- un langage sans ambiguïtés,
- un langage universel pouvant servir de support pour tout langage orienté objet,
- un moyen de définir la structure d'un programme,
- une représentation visuelle permettant la communication entre les acteurs d'un même projet, et
- une notation graphique simple, compréhensible même par des non informaticiens.

3.2.1 Responsibility and User Management Service RMS

La session est la notion de base de notre sujet. Il doit être possible de la définir en amont de son exécution et de conserver ces informations afin de pouvoir les réutiliser. Il s'agit d'un processus asynchrone. Le service RMS fournit les fonctions nécessaires pour définir le profil des participants, les groupes et les sessions. Le service RMS a pour mission aussi de fournir au service SMS et aux applications utilisées, les informations relatives à la session qui va être exécutée.

Le profil de participant inclut l'information générale de participant et les attributs appropriés permettant de le sélectionner en fonction de la nature de l'activité de coopération et selon ses qualifications pour participer dans une session de coopération. Les responsabilités dans un groupe peuvent être dynamiquement attribuées (par le chairman de la session ou automatiquement) aux participants selon leurs qualifications et leur disponibilité pour rejoindre une session. Une session est définie essentiellement par le groupe de participants qui y participent et le groupe des applications que ces derniers utilisent.

3.2.1.1 Diagramme Use Case UML de RMS

La Figure 3-1 décrit le diagramme UML de cas d'utilisation pour le service RMS. Les diagrammes de cas d'utilisation, également nommés use case, sont la solution UML pour représenter le modèle conceptuel. Ces diagrammes permettent de structurer les besoins des utilisateurs et les objectifs correspondants d'un système. Ils

¹ Unified Modeling Language

centrent l'expression des exigences du système sur ses utilisateurs : ils partent du principe que les objectifs du système sont tous motivés. Ils se limitent aux préoccupations « réelles » des utilisateurs; ils ne présentent pas de solutions d'implémentation et ne forment pas un inventaire fonctionnel du système. Ils identifient les utilisateurs du système (acteurs) et leur interaction avec le système. Ils permettent de classer les acteurs et de structurer les objectifs du système. Ils servent de base à la traçabilité des exigences d'un système dans un processus de développement intégrant UML.

Dans la construction des diagrammes de cas d'utilisation de nos services RMS et SMS, nous avons identifié trois types d'utilisateurs, il s'agit :

- Du président de la session ou chairman : il doit être défini au niveau de la session comme tout autre participant, mais il a des responsabilités particulières. C'est lui qui contrôle le déroulement de la session : il invite les participants, il initialise, ouvre, clôt et termine la session, il gère le droit de parole des invités et leur donne l'accès aux outils utilisés.
- Du participant de la session : c'est un utilisateur qui participe à des sessions de coopération et qui utilise les applications affectées à la session en cours. Chaque participant est défini par des caractéristiques propres telles que son identité, ses compétences, ses rôles et ses thèmes. Il a accès aux informations le concernant et il peut s'il le souhaite les modifier.
- Du responsable de la définition des sessions : c'est lui qui définit et planifie la session. Il lui attribue une liste de participants et un chairman. Ces derniers sont sélectionnés en fonction des rôles qu'ils peuvent avoir, de leurs compétences et des domaines d'activités dans lesquels ils travaillent. Dans la plus part des cas, la définition de la session incombe au chairman lui même.

La Figure 3-1 résume les actions que l'utilisateur en charge de la définition des sessions peut demander au service RMS. Le service RMS se résume à trois grandes activités, composées à leur tour de plusieurs sous-activités. La première consiste en la définition d'une liste de participants, ces participants sont les utilisateurs qui interviendront lors de sessions. La deuxième consiste en la définition d'une liste de groupes, où chaque groupe représente un ensemble de participants. Enfin, la troisième consiste en la définition d'une liste de sessions, où chaque session est composée d'un groupe de participants et d'autres attributs. La composition de chacune de ces trois activités est présentée dans la sous-section suivante par un diagramme d'activités.

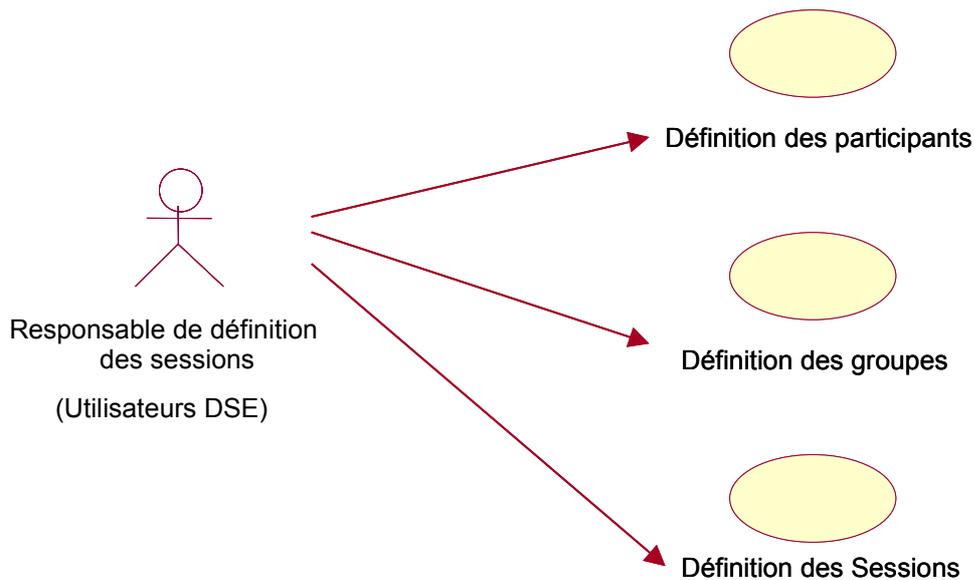


Figure 3-1 Diagramme de cas d'utilisation du RMS

3.2.1.2 Diagramme d'activités

Les diagrammes d'activités permettent de représenter graphiquement le comportement d'une méthode ou le déroulement d'un cas d'utilisation. Une activité représente une exécution d'un mécanisme, un déroulement d'étapes séquentielles. Le passage d'une activité vers une autre est matérialisé par une transition. Les transitions sont déclenchées par la fin d'une activité et provoquent le début immédiat d'une autre (elles sont automatiques). Les barres de synchronisation désignent le rendez vous de plusieurs activités avant de franchir la transition aux activités suivantes.

La Figure 3-2 présente le diagramme d'activités du service RMS. Ce diagramme montre les trois activités primaires du service (à gauche) et l'ensemble des sous-activités à effectuer pour les obtenir.

Chaque participant est caractérisé par quelques informations sur son identité (nom, prénom, société...), par un profil (compétences, rôles, thèmes) et des données nécessaires pour l'interaction des différentes applications du système (adresse e-mail, adresse IP, login et mot de passe).

La définition des groupes consiste en la saisi de l'information générale du groupe et la sélection de un ou plusieurs participants, suivies par la sélection parmi ces derniers d'un responsable de groupe (chairman). L'affectation des participants à un groupe est réalisée en fonction des rôles qu'ils peuvent avoir (chairman, secrétaire, critique...), de leurs compétences (membre du groupe outils et logiciels pour la communication, membre du comité de revue, membre du comité de programme...) et des thèmes (protocoles de communication, sûreté de fonctionnement, applications multimédias...) dans lesquels ils exercent leur activité. Un même groupe peut intervenir dans plusieurs sessions.

Pour définir une session, il faut :

- Saisir des caractéristiques qui lui sont propres à savoir son libellé, des informations de planification...

- Lui affecter un groupe de participants qui seront automatiquement invités à y participer et ayant les compétences requises pour son bon déroulement.
- Lui affecter un ensemble d'applications qui seront employées pour réaliser le travail coopératif entre les participants.

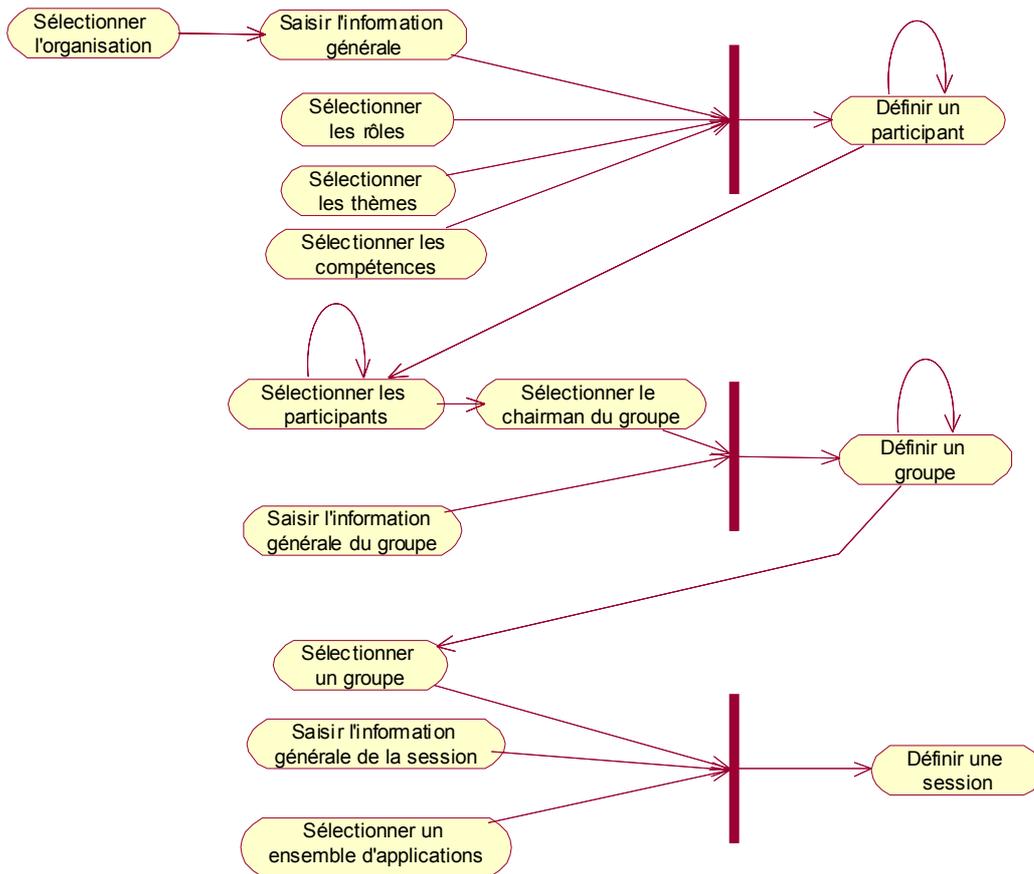


Figure 3-2 Diagramme des activités du service RMS

3.2.2 Session Management Service SMS

Le service SMS se compose de l'ensemble de mécanismes par lesquels les utilisateurs peuvent initialiser, trouver, ouvrir, clôturer, joindre, quitter et terminer une session de coopération. Le service commande les changements de l'état d'une session et affiche l'information au sujet de ces changements. Le service de gestion des sessions (SMS) a les responsabilités suivantes:

- le chairman de la session est responsable d'inviter, d'accepter ou de refuser un participant.
- SMS invite automatiquement les participants.

- le chairman de la session décide le moment où la session est ouverte, c'est à dire, le moment où les participants se mettent à travailler de manière coopérative en utilisant les applications affectées à la session.
- l'état de la session est mis à jour si un participant se joint ou quitte la session courante.
- l'état de la session est mis à jour si le chairman de la session initialise, ouvre, clôture, ou termine la session courante.
- le chairman de la session décide le moment de fermeture. Ceci marque l'arrêt du travail coopératif.
- SMS permet la communication entre participants par une fonctionnalité d'échange des messages de communication synchrone
- SMS permet la communication asynchrone entre les participants par l'envoi des messages asynchrones. Ces messages peuvent être adressés aux participants invités (les participants invités constituent la liste des participants définis dans une session au niveau RMS) ou bien aux participants connectés.
- le chairman de la sessions est capable de concéder le rôle de « chairman » à un autre participant à condition de que celui-ci possède l'attribut de chairman dans sa liste de rôles.

3.2.2.1 Diagramme Use Case de SMS

Le diagramme illustré par la Figure 3-3 définit les mécanismes fournis par le service de gestion de session SMS. Ces mécanismes répondent aux fonctions de gestion des sessions de base :

- Initialiser une session.
- Convoquer les participants d'une session.
- Mécanisme de recherche et découverte des sessions.
- Ouvrir et fermer une session.
- Connecter et déconnecter les participants à une session.
- Assigner un responsable d'une session.
- Afficher l'état courant d'une session.

Le cas d'utilisation « Initialiser la session » correspond à la création des objets nécessaires capables d'effectuer la coordination des actions pendant l'exécution de la session. Cette opération est la première activité dans le cycle de vie d'une session de coopération. La réalisation de cette activité est sous la responsabilité du chairman de la session.

Le cas d'utilisation « Annoncer la session » prend a charge l'invitation des participants potentiels à une session. Il comprend aussi la réponse, l'acceptation ou la négative, de chaque participant convoqué.

Le cas d'utilisation « Ouvrir et se joindre à la session » est composé par deux activités. D'une part, les participants se joignent à une session, c'est à dire qu'ils se connectent depuis leurs postes de travail. D'autre part, le chairman est en charge d'ouvrir la session, c'est à dire de marquer le moment ou le travail coopératif commence.

Le cas « Quitter la session » est composé par la déconnexion des participants.

Le cas « Clôre la session » correspond à la clôture de la session effectué par le chairman. Cette activité marque la fin du travail coopératif.

Enfin, le cas « Terminer la session » comprend la destruction des objets de la session qui ont été créés pendant le cas « Initialiser la session ». Cette activité est effectuée par le chairman de la session.

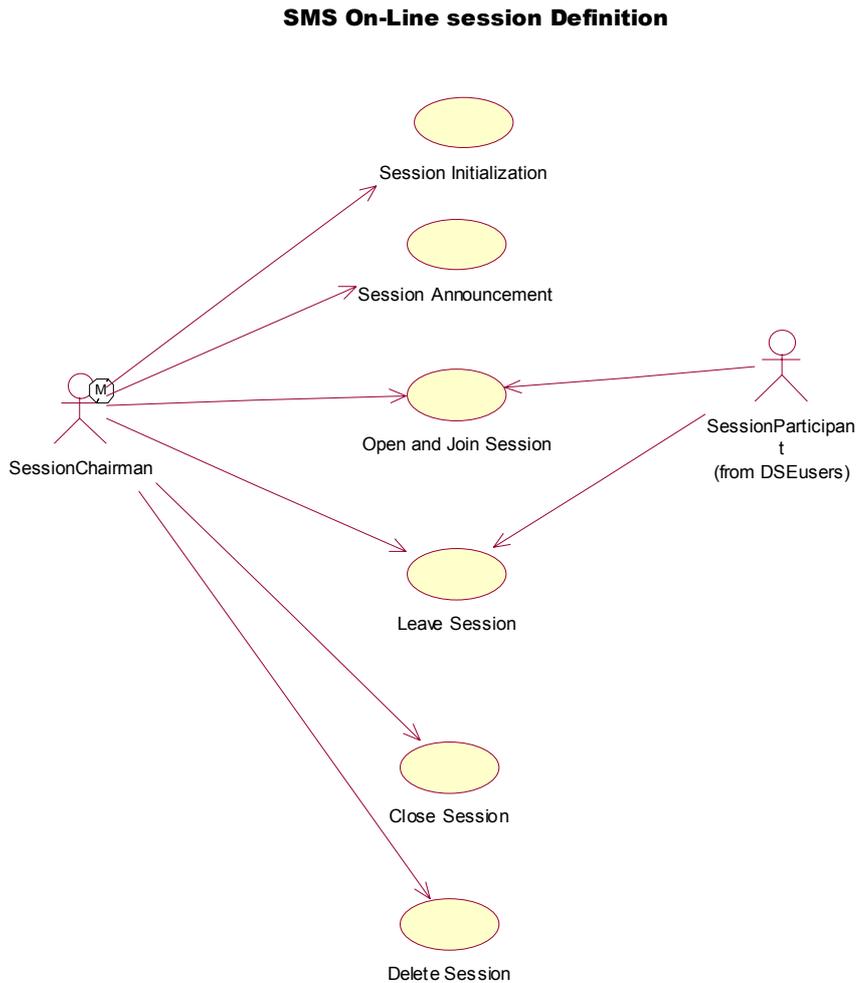


Figure 3-3 Diagramme Use Case du service SMS

3.2.2.2 Diagrammes d'états-transitions SMS

Les diagrammes d'états-transitions permettent de décrire les changements d'états d'un objet ou d'un composant, en réponse aux interactions avec d'autres objets/composants ou avec des acteurs. Un état se caractérise par sa durée et sa stabilité, il représente une conjonction instantanée des valeurs des attributs d'un objet. Une transition représente le passage instantané d'un état vers un autre. Elle est

déclenchée par un événement. En d'autres termes : c'est l'arrivée d'un événement qui conditionne la transition. Les transitions peuvent aussi être automatiques, lorsqu'on ne spécifie pas l'événement qui la déclenche. En plus de spécifier un événement précis, il est aussi possible de conditionner une transition, à l'aide de « gardes » : il s'agit d'expressions booléennes, exprimées en langage naturel (et encadrées de crochets).

Nous avons choisi ce type de diagramme à la place des diagramme d'activités pour montrer la dynamique des participants et des sessions de coopération.

La Figure 3-4 illustre les état et les transitions dans le cycle de vie d'un participant. Le diagramme est divisé en deux parties, à gauche se trouvent les état correspondants à la définition du participant par le service RMS (*inexistent*, *defined*, *idle* et *authorized*) et à droite se trouvent les états du participant pendant son intervention dans l'exécution d'une session gérée par le service SMS (*invited*, *expected*, *connected* et *disconnected*).

Du côté RMS, un participant commence à l'état *inexistent*, car il n'est pas encore créé. Une fois que le responsable de la définition des sessions saisit l'information nécessaire pour remplir un profil, le participant devient défini (*defined*). Un participant défini reste en attente (*idle*) jusqu'à qu'il soit associé à un groupe et celui-ci employé dans une session initialisée, alors le participant est autorisé (*authorized*) à prendre part dans la session.

Du côté SMS, le chairman de la session invite les participants, alors les participants autorisés passent à l'état d'invités (*invited*). Les participants qui acceptent deviennent attendus (*expected*) et ceux qui refusent, retournent à l'état d'attente (*idle*). Les participants attendus peuvent se joindre à la session, alors ils passent à l'état de connexion (*connected ordinary*). Pendant la connexion, un participant peut devenir chairman de la session, dans ce cas le participant passe à l'état de connexion en tant que chairman (*connected chairman*). En tant que participant, ou en tant que chairman, le participant peut quitter la session, alors il devient déconnecté (*disconnected*). Un participant déconnecté peut terminer son cycle de vie du point de vue SMS, ou bien retourner à l'état *idle* pour attendre une nouvelle participation.

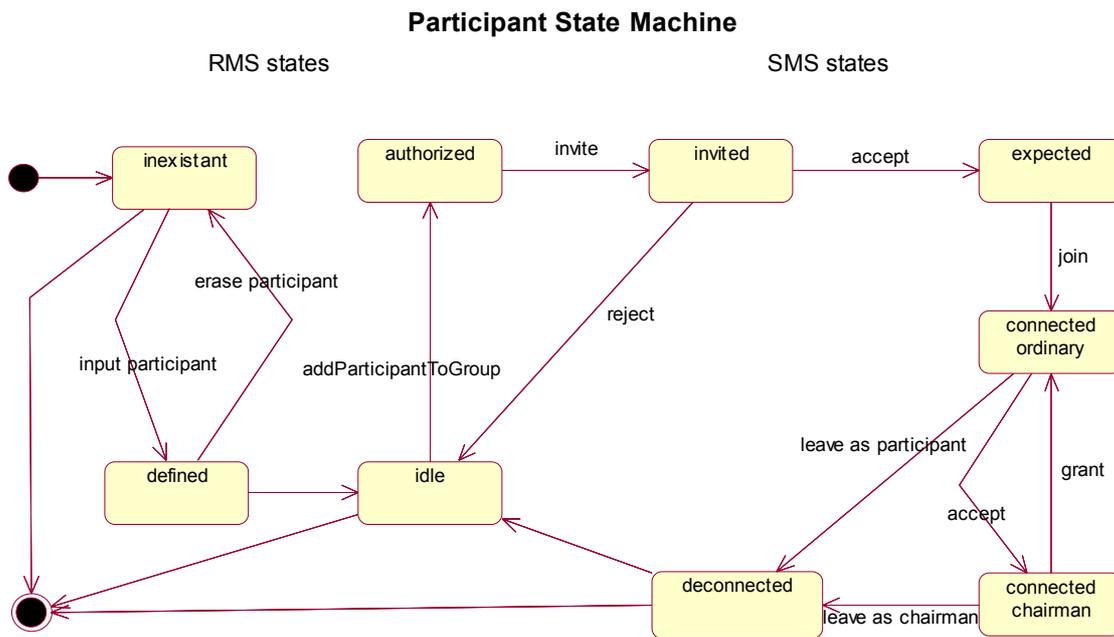


Figure 3-4 Diagramme d'états-transitions d'un participant

La Figure 3-5 illustre le diagramme d'états-transitions d'une session. La partie gauche du diagramme contient l'état du point de vue RMS (*inexistant* et *defined*), le reste des états d'une session appartient aux actions du service SMS. Une session commence à l'état *inexistant*, une fois que le responsable de la définition des sessions saisit l'information correspondante, elle passe à l'état défini (*defined*). Une session définie commence son cycle de vie dans le service SMS une fois que le chairman de la session l'initialise (*initialized*). Cette action apporte l'objet session de l'état défini à l'état initialisé. Une fois créé, l'objet session est employé pour annoncer la session aux participants impliqués (*announced*). Dans notre service cette annonce est faite via un message électronique qui est envoyé à chaque participant potentiel. Le courrier contient le nom de session, le mot de passe, la date et l'heure de début, la durée, le thème et l'URL pour répondre à l'invitation.

En recevant cette annonce chaque participant (par l'intermédiaire d'une page Web) peut décider d'accepter ou de refuser (voir les transitions *accept* et *reject* du diagramme précédent) l'appel pour rejoindre la session. Ensuite, les participants ayant accepté l'invitation se joignent à la session (*join*). Quand tous les participants attendus sont connectés ou bien quand un nombre « représentatif » des participants est connecté, le chairman de la session peut décider d'ouvrir (*opened*) ou d'annuler (*deleted*) la session. Ceci dépend de la politique suivie par le chairman. Quand la session est ouverte (*opened*), un message est envoyé à tous les participants impliqués. L'activité de collaboration est alors commencée. En ce moment, tous les participants impliqués lancent leurs outils de coopération tels que des applications de vidéoconférence, de partage d'applications, des programmes de simulation. Pendant cet état les participants travaillent ensemble afin de réaliser un but commun. Au cours

de cette phase les participants peuvent quitter (*leave*) la session après l'approbation du chairman. Une liste de participants réels aide le chairman à décider si un participant peut quitter la session.

À la fin de l'activité, le président déclare la session clôturée (*closed*) et informe tous les participants en envoyant un message spécifique. Ceci signifie la fin de la phase de coordination. Après cette annonce, les participants peuvent quitter la session sans demander l'approbation du chairman. Le SMS est alors consacré aux activités de compte rendu de la session. À la fin de cette activité, la session est déclarée comme terminée (*deleted*).

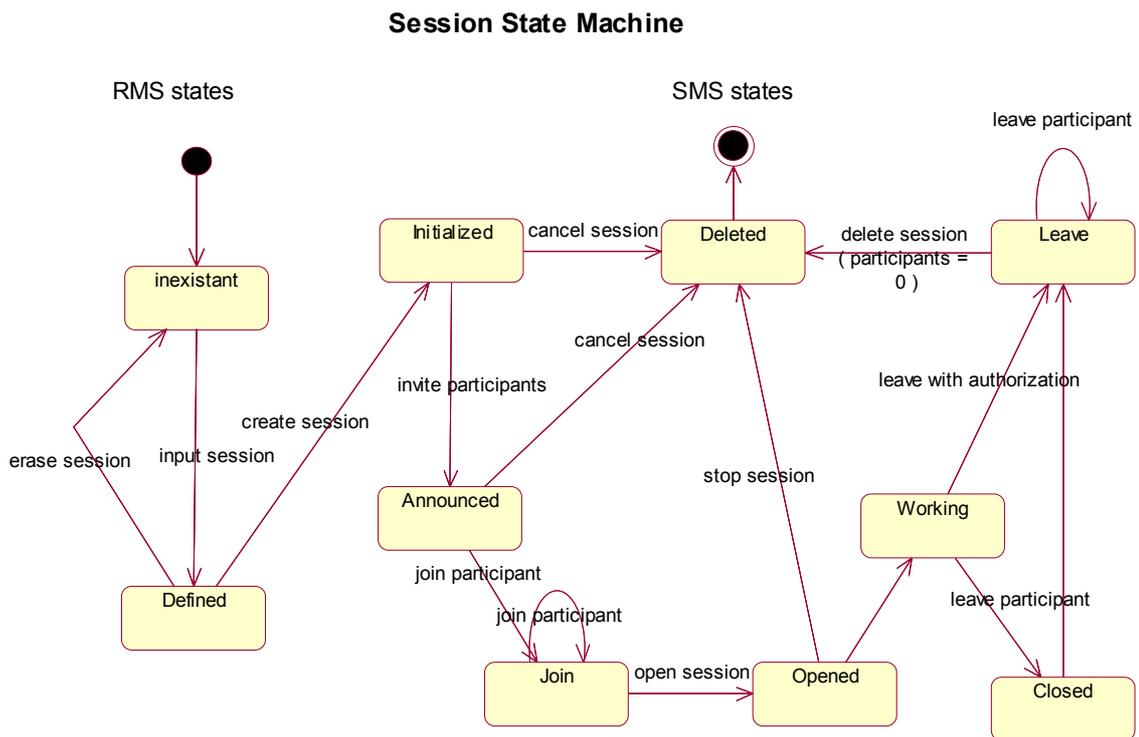


Figure 3-5 Diagramme d'états-transitions d'une session

3.3 Modèle de coordination de sessions de coopération

Dans cette section nous décrivons le modèle de coordination de sessions proposé. Ce modèle se caractérise par la coordination des interactions entre acteurs lors des sessions de coopération. Les interactions entre acteurs se traduisent par des événements, un événement est l'exécution d'une action par un acteur. Le modèle est basé sur des ordres partiels étiquetés (OPE) et sur des formules logiques (FL).

La modélisation du comportement des systèmes concurrents par des ordres partiels étiquetés permet d'exprimer les relations de causalité entre les événements [Pra1987]. Grâce à l'utilisation des OPE, nous avons défini l'ordre des actions réalisées par les participants et les applications pendant une session. L'avantage des OPE vis-à-vis des comportements du type séquentiel ou du type entrelacement réside dans le fait que quelques propriétés ne peuvent être vérifiées que dans l'ordre partiel.

Par exemple, la relation de dépendance immédiate [PMS1997] ne peut pas, en général, être vérifiée dans les entrelacements.

Grâce à l'utilisation des formules logiques de premier ordre, nous pouvons exprimer les contraintes de coordination entre événements, et ainsi définir les règles de coordination. Ces règles de coordination sont le résultat de la combinaison des formules avec des opérateurs logiques.

3.3.1 Modélisation de sessions de coopération

En suivant la définition de session de coopération donné en 2.4.1, les éléments principaux à abstraire sont les acteurs et les actions de la session. Les acteurs représentent les entités qui réalisent les interactions. Les interactions correspondent à l'exécution des actions, par exemple l'action d'ouvrir une session de coopération effectuée par le chairman de la session.

Définition 3.1 (Acteurs et Actions) *Nous définissons les acteurs d'une session de coopération par l'union de l'ensemble des applications et des utilisateurs. Formellement :*

$N = T \cup P$ est un ensemble fini dénotant les acteurs de coopération dans une session. Ces acteurs peuvent être des applications logiciels (éléments de T) ou des participants humains (éléments de P).

Dans la suite, A est un ensemble fini qui dénote les actions de coordination de sessions effectuées par les acteurs.

En utilisant la définition 3.1 d'acteurs et d'actions, nous formalisons une session de coopération par :

Définition 3.2 (Session Coopérative) *Une session coopérative sur N et A est un ordre partiel étiqueté $s = (E_s, \leq_s, l_s)$ où :*

E_s est un ensemble fini d'événements,
 \leq_s est une relation d'ordre, c'est à dire une relation binaire réflexive, antisymétrique et transitive,
 $l_s : E_s \rightarrow N \times A$ est une fonction d'étiquetage sur les événements,
 Un événement $e \in E_s$ est appelé une occurrence du couple (acteur, action) $l_s(e)$.

Nous définissons aussi :

- La fonction $act(e)$ spécifie l'acteur n de l'événement e :
 $act : E_s \rightarrow N$
 $act(e) = n \Leftrightarrow \exists a \in A \ l(e) = (n, a)$
- La fonction $mgt(e)$ détermine l'action de gestion de session a de l'événement e :
 $mgt : E_s \rightarrow A$
 $mgt(e) = a \Leftrightarrow \exists n \in N \ l(e) = (n, a)$
- Nous remarquons que pour tout couple d'événements $e, e' \in E_s$:

si $act(e) = act(e')$ alors $e \leq e' \vee e' \leq e$

3.3.2 Modélisation des règles de coordination

Les règles de coordination d'actions entre les acteurs d'une session de coopération sont définies par des formules logiques, celles-ci sont formalisées ci-après :

Définition 3.3 (Syntaxe et sémantique) On dénote par $FO(\leq, N \times A)$ les formules logiques du premier ordre construites sur la relation \leq et l'alphabet $N \times A$, et qui sont définies par la grammaire :

$$\varphi ::= P_{(n,a)}(x) \mid x \leq y \mid \varphi \wedge \psi \mid \neg \varphi \mid \exists x \varphi \mid \forall x \varphi$$

On écrit $\varphi(x_1, \dots, x_n)$ afin de spécifier l'occurrence de variables libres dans une formule $\varphi \in FO$. Etant donné une formule $\varphi(x_1, \dots, x_n)$, une session coopérative $s = (E_s, \leq_s, l_s)$ et $e_1, \dots, e_n \in E_s$ on dénote $(s, e_1, \dots, e_n) \models \varphi(x_1, \dots, x_n)$, pour signifier que φ est vérifiée par s (s satisfait φ) lorsqu'on affecte à x_i la valeur e_i pour $i = 1, \dots, n$:

$$\begin{aligned} (s, e) &\models P_{(n,a)}(x) \text{ si et seulement si } l_s(e) = (n, a) \\ (s, e_1, e_2) &\models x_1 \leq x_2 \text{ si et seulement si } e_1 \leq_s e_2 \\ (s, e_1, \dots, e_n, e_{n+1}, \dots, e_{n+p}) &\models \varphi(x_1, \dots, x_n) \wedge \psi(y_1, \dots, y_p) \text{ si et seulement si } (s, e_1, \dots, e_n) \\ &\models \varphi(x_1, \dots, x_n) \text{ et } (s, e_{n+1}, \dots, e_{n+p}) \models \psi(y_1, \dots, y_p) \\ (s, e_1, \dots, e_n) &\models \neg \varphi(x_1, \dots, x_n) \text{ si et seulement si il n'est pas vrai que } (s, e_1, \dots, e_n) \\ &\models \varphi(x_1, \dots, x_n) \\ (s, e_1, \dots, e_n) &\models \exists x \varphi(x, x_1, \dots, x_n) \text{ si et seulement si il existe } e \in E_s \text{ tel que } \\ (s, e, e_1, \dots, e_n) &\models \varphi(x, x_1, \dots, x_n) \\ (s, e_1, \dots, e_n) &\models \forall x \varphi(x, x_1, \dots, x_n) \text{ si et seulement si, pour tout } e \in E_s, (s, e, e_1, \dots, e_n) \models \\ &\varphi(x, x_1, \dots, x_n) \end{aligned}$$

En particulier, si φ est un énoncé (φ ne contient pas de variables libres), on dit que s vérifie φ et on écrit $s \models \varphi$.

On définit aussi les formules dérivées suivantes :

$$Act_n(x) \equiv \bigvee_{a \in A} P_{(n,a)}(x)$$

$$Mgt_a(x) \equiv \bigvee_{n \in N} P_{(n,a)}(x)$$

$$(s, e) \models Act_n(x) \text{ si et seulement si } act(e) = n$$

$$(s, e) \models Mgt_a(x) \text{ si et seulement si } mgt(e) = a$$

Définition 3.4 (Relation d'inégalité) Un ordre \leq donne naissance à une relation d'inégalité stricte $<$ par :

$$x < y \Leftrightarrow (x \leq y \wedge x \neq y)$$

Définition 3.5 (Relation de couverture) Etant donné \leq la relation de couverture \rightarrow est définie par :

$x \rightarrow y$ si et seulement si $x < y \wedge \forall z (x \leq z \leq y \Rightarrow z = x \vee z = y)$. Ceci signifie qu'il n'y a pas de z tel que $x < z < y$.

3.4 Modélisation des dépendances entre événements

3.4.1 L'énoncé de précédence

L'énoncé de précédence est défini afin d'exprimer la relation causale entre deux événements étiquetés par des actions de gestion des sessions effectuées par les acteurs lors d'une session coopérative.

Définition 3.6(Précédence) Pour toute paire d'événements (n,a) , (n',a') appartenant tous les deux à $N \times A$, l'énoncé précédence, noté $Pred((n,a),(n',a'))$, est défini par :

$$Pred((n,a),(n',a')) \equiv \forall x, P_{(n',a')}(x) \Rightarrow (\exists y, y < x \wedge P_{(n,a)}(y))$$

On définit aussi les énoncés annexes :

$$Pred_n(a,a') \stackrel{def}{=} Pred((n,a),(n',a'))$$

$$Pred_a(n,n') \stackrel{def}{=} Pred((n,a),(n',a))$$

Cette relation est interprétée ainsi : tout événement (x) qui indique l'exécution de l'action a' par l'acteur n' doit être précédé par un événement (y) qui indique l'exécution de l'action a par l'acteur n . Un exemple typique de l'ordre de précédence des actions dans une session coopérative est la définition d'un ordre de connexion pour les participants. Par exemple selon la relation représentée par la formule φ_1 décrite dans l'Équation 3.1 ci-après, le *secrétaire* de la session ne peut pas être relié (action *join*) avant le *président* de la session, et tous les *participants* de la session doivent attendre la connexion du *secrétaire*. Ainsi l'ordre de connexion est d'abord le *président*, puis le *secrétaire* et finalement les *participants*.

$$\varphi_1 = Pred_{join}(président, secrétaire) \wedge_{p \in Participants} Pred_{join}(secrétaire, p)$$

Équation 3.1 Exemple d'ordre de connexion d'une session

3.4.2 L'énoncé d'inhibition

L'énoncé d'inhibition permet définir l'interdiction de l'exécution d'un événement étiqueté par une action de gestion de session après l'exécution d'une action effectuée préalablement.

Définition 3.7(Inhibition) Pour toute paire d'événements (n,a) , (n',a') appartenant tous les deux à $N \times A$, l'énoncé inhibition, noté $Inhib((n,a),(n',a'))$, est défini par :

$$Inhib((n,a),(n',a')) : \forall x, P_{(n',a')}(x) \Rightarrow (\forall y, y < x \Rightarrow \neg P_{(n,a)}(y))$$

On définit aussi les énoncés annexes :

$$Inhib_n(a,a') \stackrel{def}{=} Inhib((n,a),(n',a'))$$

$$Inhib_a(n, n') \stackrel{def}{=} Inhib((n, a), (n', a))$$

La relation d'inhibition suit l'interprétation suivante: tout événement (x) qui indique l'exécution de l'action a' par l'acteur n' ne peut être précédé par un autre événement (y) qui indique l'exécution de l'action a par l'acteur n . Un exemple d'une telle dépendance est l'interdiction de connexion des participants retardataires après que la session ait été ouverte, cet exemple est représenté par φ_2 dans l'Équation 3.2.

$$\varphi_2 = \bigwedge_{p \in P} Inhib((session_management_tool, open), (p, join))$$

Équation 3.2 Exemple d'interdiction de connexion des retardataires après l'ouverture d'une session

3.4.3 L'énoncé de déclenchement

L'énoncé de déclenchement est défini pour permettre l'expression d'une relation causale immédiate entre deux événements étiquetés par des actions de gestion des sessions effectuées par les acteurs lors d'une session coopérative. Cette relation sert à définir une séquence de deux événements consécutifs entre lesquels il n'y a pas d'événements intermédiaires comme dans le cas de l'énoncé de précédence).

Définition 3.8 (Déclenchement) *Pour toute paire d'événements (n, a) , (n', a') appartenant tous les deux à $N \times A$, l'énoncé déclenchement, noté $ImPred((n, a), (n', a'))$, est défini par :*

$$ImPred((n, a), (n', a')) : \forall x, P_{(n', a')}(x) \Rightarrow (\exists y, y \rightarrow x \wedge P_{(n, a)}(y))$$

On définit aussi les énoncés annexes :

$$ImPred_n(a, a') \stackrel{def}{=} ImPred((n, a), (n, a'))$$

$$ImPred_a(n, n') \stackrel{def}{=} ImPred((n, a), (n', a))$$

Cette relation est interprétée comme suit: tout événement (x) qui indique l'exécution de l'action a' par l'acteur n' est précédé immédiatement par un autre événement (y) qui indique l'exécution de l'action a par l'acteur n . Un exemple de cette relation est l'ordre de lancement pour des applications du type producteur/consommateur. La formule φ_3 illustrée par l'Équation 3.3 montre la dépendance de l'action de démarrage $start$ pour le comportement de collaboration producteur/consommateur entre l'application de production $producer_tool$ et son application de consommation associé $consumer_tool$.

$$\varphi_3 = ImPred_{start}(producer_tool, consumer_tool)$$

Équation 3.3 Exemple du démarrage des applications producteur/consommateur

3.5 Application du modèle : spécification d'une session de coopération avec des règles de coordination de base

Dans cette sections, nous présentons la définition des règles de coordination de base pour des sessions de coopération. Cet ensemble montre les choix faits pour définir les politiques de coordination entre les actions effectuées par les participants et les applications lors d'une session.

3.5.1 Définition des acteurs

L'Équation 3.4 montre la définition de l'ensemble d'acteurs d'une session de coopération. L'ensemble d'acteurs est le résultat de la réunion d'un ensemble de participants et d'un ensemble d'applications. Parmi les participants, il existe un qui joue un rôle spécial. Il s'agit du chairman de la session. Ce participant présidera la session et se chargera de sa gestion. Ceci est illustré par l'Équation 3.5. L'Équation 3.6, montre les différents types d'applications qu'on peut considérer comme « basiques » afin d'établir une session de coopération. D'abord, il faut un service capable de gérer la session : configuration de la session, invitation des participants, gestion en ligne, etc. Ensuite, il faut une application de vidéoconférence qui permettra à n participants de pouvoir discuter en direct. On suppose que le travail coopératif sera réalisé à l'aide des applications du type collectif, il est donc nécessaire d'avoir un service capable de gérer l'ensemble de ce type d'applications. Finalement, les participants doivent avoir connaissance des actions faites par les autres participants, il faut donc un service de notification d'événements.

$$N_{basic} = P_{basic} \cup T_{basic}$$

Équation 3.4 Acteurs de base

$$P_{basic} = \{chairman\} \cup \{p_i \mid 1 \leq i \leq n\}$$

Équation 3.5 Définition des participants

$$T_{basic} = \{smt, gct, tms, evs\} \text{ où :}$$

smt : (*Session Management Tool*) Service de Gestion de Session
gct : (*Group Conferencing Tool*), Application de vidéoconférence à n
tms : (*Tool Management Service*), Service de Gestion d'Applications
Collecticiels
evs : (*Event Notification Service*), Service de Notification

Équation 3.6 Définition des applications

3.5.2 Définition des actions

L'Équation 3.7 montre l'ensemble des actions ayant lieu lors d'une session de coopération. Cet ensemble est le résultat de la réunion des actions des participants, des applications et du chairman.

$$A_{basic} = A_{participant} \cup A_{outil} \cup A_{chairman}$$

Équation 3.7 Définition des actions

3.5.2.1 Les actions des participants

Chaque participant est capable d'effectuer un certain nombre d'actions, qui sont décrites par l'Équation 3.8. Un participant effectue l'action *join*, lorsqu'il se connecte à une session. De manière similaire, il effectue *leave* au moment de se déconnecter d'une session. Un participant peut accepter (*accept*) le rôle du chairman. Enfin, un participant réalise l'action *message*, lorsqu'il envoie un message à un ou plusieurs autres participants. Le chairman, en tant que participant, peut réaliser les mêmes actions que les autres. Et dû à son rôle spécial, il peut en plus effectuer l'action *grant*, illustrée par l'Équation 3.9, qui consiste à passer le rôle chairman à un autre participant.

$$A_{participant} = \{join, leave, accept, message\}$$

Équation 3.8 Définition des actions des participants

$$A_{chairman} = \{grant\} \cup A_{participant}$$

Équation 3.9 Définition des actions du chairman

3.5.2.2 Les actions des applications

L'ensemble d'actions des applications est composé par la réunion des ensembles d'actions des différents types d'applications. Il est défini par l'Équation 3.10.

$$A_{outil} = A_{smt} \cup A_{gct} \cup A_{tms} \cup A_{evs}$$

Équation 3.10 Actions réalisées par les applications

L'Équation 3.11 montre l'ensemble d'actions du service de gestion de session. Le gestionnaire est capable d'initialiser (*create*) et de terminer (*delete*) des instances de sessions de coopération. Il a en charge aussi de l'ouverture (*open*) et la clôture (*close*) de sessions. Le gestionnaire réalise la convocation (*invite*) des participants à une session.

$$A_{smt} = \{create, delete, open, close, invite\}$$

Équation 3.11 Actions du gestionnaire de session

L'application de vidéoconférence met en service (*enable*) ou hors service (*disable*) l'utilisation d'une conférence associée à la session, voir Équation 3.12.

$$A_{gct} = \{enable, disable\}$$

Équation 3.12 Actions du gestionnaire des applications collecticiel

Le service de gestion des applications collecticiels effectue le démarrage (*start*) et l'arrêt (*stop*) des applications, voir Équation 3.13.

$$A_{tms} = \{start, stop\}$$

Équation 3.13 Actions du gestionnaire du contexte

L'Équation 3.14 illustre les actions du service de notification. Ce service effectue l'envoi (*push*) et la récupération (*pull*) d'événements.

$$A_{evs} = \{push, pull\}$$

Équation 3.14 Actions du gestionnaire de notification

3.5.3 Règles de d'ordonnancement des événements contrôlant le changement d'état

3.5.3.1 Gestion de la cohérence du changement de l'état de la session

La règle de coordination représentée dans l'Équation 3.15 montre les transitions d'une session de coopération. Une session commence son cycle de vie après l'exécution de l'événement *initialize* qui établit l'état de la session à *initialized*. Après sa création, une session peut passer à l'état *announced* ou à l'état *deleted*, selon l'événement exécuté. L'état *announced* signifie que les participants ont été invités à la session. Tandis que l'état *deleted* est l'état de terminaison de la session. Par ailleurs, une session annoncée soit est ouverte, dans ce cas elle passe à l'état *opened*, soit elle est annulée, dans ce cas la session passe à l'état *deleted*. L'état *opened* marque le début du travail collaboratif, celui-ci doit être clôturé par l'état *closed*. Le travail collaboratif peut être arrêté en passant de l'état *opened* à l'état *deleted*. Une fois la session fermée (*closed*), elle est terminée en passant à l'état *deleted*.

$$\delta_1 = Pred_{smt}(create, invite) \wedge Pred_{smt}(invite, open) \wedge Pred_{smt}(open, close) \\ \wedge \left(\begin{array}{l} Pred_{smt}(create, delete) \vee Pred_{smt}(invite, delete) \\ \vee Pred_{smt}(open, delete) \vee Pred_{smt}(close, delete) \end{array} \right)$$

Équation 3.15 Description des règles de coordination d'une session

3.5.3.2 Gestion de la cohérence du changement de l'état d'un participant

L'Équation 3.16 définit la règle de coordination qui spécifie qu'un participant ne peut se déconnecter que si et seulement s'il a été connecté auparavant.

$$\delta_2 = \bigwedge_{p \in P_{basic}} Pred_p(join, leave)$$

Équation 3.16 Règle de participation effective

3.5.3.3 Gestion de la cohérence du changement de l'état des applications

L'Équation 3.17 définit une propriété de sûreté, laquelle consiste à assurer qu'une application ne peut être arrêtée que si elle a été démarrée préalablement. Comme conséquence, le gestionnaire de session doit avoir en mémoire la liste des applications démarrées.

$$\delta_3 = \bigwedge_{t \in T_{basic}} Pred_t(start, stop)$$

Équation 3.17 Règle d'arrêt des applications

La règle de l'Équation 3.18 définit que les applications sont arrêtées automatiquement après la terminaison de la session.

$$\delta_4 = \bigwedge_{t \in T_{basic}} \text{ImPred}((smt, delete), (t, stop))$$

Équation 3.18 Règle d'arrêt automatique des applications

3.5.4 Règles d'appartenance au groupe (membership rules)

Cette catégorie spécifie les actions que peuvent ou ne peuvent pas et que doivent entreprendre les participants en fonction de l'état de la session.

3.5.4.1 Règles de formation du groupe

3.5.4.1.1 Admission uniquement sur invitation

Nous avons considéré que la connexion des participant à une session n'est possible que si et seulement s'ils ont été invités à cette session. Cette règle est illustrée par l'Équation 3.19.

$$\delta_5 = \bigwedge_{p \in P_{basic}} \text{Pred}((smt, invite), (p, join))$$

Équation 3.19 Règle d'admission sur invitation des participants

3.5.4.1.2 Admission uniquement avant terminaison

La règle illustrée par l'Équation 3.20 indique que les participants ne peuvent pas se connecter à une session qui a été terminée. Cette règle assure l'interdiction de connexion à une session qui n'existe plus.

$$\delta_6 = \bigwedge_{p \in P_{basic}} \text{Inhib}((smt, delete), (p, join))$$

Équation 3.20 Règle d'admission des participants avant terminaison

3.5.4.2 Règles de dissolution du groupe

3.5.4.2.1 Contrôle du départ des participants

La règle de l'Équation 3.21 définit que les participants sont automatiquement déconnectés après la terminaison de la session.

$$\delta_7 = \bigwedge_{p \in P_{basic}} \text{ImPred}((smt, delete), (p, leave))$$

Équation 3.21 Règle de déconnexion automatique des participants

3.5.5 Règles de spécification de la portée des communications à l'intérieur et à l'extérieur du groupe

3.5.5.1 Les participants ne peuvent pas communiquer (à part avec le chairman) avant de se joindre au groupe

La règle représentée par l'Équation 3.22 indique que les participants qui quittent la session ne reçoivent plus de messages.

$$\delta_8 = \bigwedge_{p \in P_{basic}} Pred_p(join, message)$$

Équation 3.22 Règle de communication entre les participants

3.5.5.2 Seuls les membres du groupes connectés peuvent envoyer et recevoir des informations

La règle de l'Équation 3.23 montre que les participants ne peuvent plus ni recevoir ni envoyer de messages après leur déconnexion.

$$\delta_9 = \bigwedge_{p \in P_{basic}} Inhib_p(leave, message)$$

Équation 3.23 Règle d'envoi/réception des messages

3.5.6 Règles de consistance des rôles

3.5.6.1 Un seul chairman a la fois

L'équation Équation 3.24 représente la règle qui définit le passage du rôle chairman entre participants. Cette règle indique qu'un participant ne peut accepter le rôle du chairman que si, et seulement si, le chairman de la session lui a accordé ce rôle. Elle indique aussi qu'il n'y a pas d'interférence entre les sessions. La décision interne au groupe d'effectuer la cession du privilège (*grant*) doit provenir du chairman du groupe.

$$\delta_{10} = \bigwedge_{p \in P_{basic}} Pred((chairman, grant), (p, accept))$$

Équation 3.24 Règle de cession du rôle chairman entre participants

3.5.6.2 Seuls les membres du groupe peuvent prendre le contrôle de la session

La règle montrée par l'Équation 3.25 indique que les participants qui peuvent accepter le rôle de chairman de la session doivent appartenir au groupe et être connectés à la session.

$$\delta_{11} = \bigwedge_{p \in P_{basic}} Pred_p(join, accept)$$

Équation 3.25 Règle de cession du rôle chairman

3.6 CoDES : processus coopératif de conception et d'analyse dans le domaine aérospatiale

Dans cette section, nous décrivons un des scénarios de validation étudié dans le cadre du projet DSE (pour une description détaillée du projet voir la section 5.2). Il s'agit du scénario « Collaborative Design and Analysis » (Co-Des).

L'ingénierie des exigences fait partie de l'ingénierie des systèmes. L'ingénierie des exigences détermine comment un système devrait soutenir les activités des utilisateurs et une stratégie de conception de la façon dont le système doit être conçu. Elle se compose de deux activités : la définition du cahier des charges, habituellement faite par le client, et la spécification des exigences habituellement faite par l'entrepreneur.

Le cahier des charges contient la synthèse des informations recueillies lors des étapes préliminaires. La spécification des exigences contient l'information suffisante pour garantir la réussite du processus de conception. Cette activité indique en détail la fonctionnalité que le système devrait fournir.

La difficulté de définir correctement et de donner la priorité aux conditions pour les systèmes complexes implique une longue phase dans la création du cahier des charges. Cette difficulté est aggravée quand les personnes impliquées sont situées dans différents emplacements. Ceci signifie que pour échanger l'information et pour analyser les données du système, les personnes doivent utiliser des outils de communication à distance, e.g. téléphone, fax, email, courrier, etc.

La plupart des activités de conception de produits sont exécutées de manière centralisée. Souvent les personnes travaillent côte à côte. Dans ce cas, les communications sont directes et les problèmes peuvent être résolus rapidement.

Par contre, les projets du domaine spatial comportent la participation de compagnies distribuées autour du monde. Les membres d'un projet peuvent être situés dans différents emplacements. Ceci pose un problème significatif. Dans un environnement distribué, la discussion face à face ou la résolution de problèmes au tour d'une table est très rare. La taille des attachements de e-mail est limitée. Les conversations téléphoniques sont insatisfaisantes pour décrire les changements de la conception et donc les réunions face à face devient une exigence. Mais, le déplacement de personnes est coûteux car une grande partie de temps est investie dans le voyage et même la réunion peut être inefficace (étant donné que beaucoup doit être fait en peu de temps). L'éloignement rend difficile aux individus d'examiner interactivement des documents ou de décrire leurs idées graphiquement. Ainsi l'intérêt croissant des méthodes pour soutenir la collaboration à distance.

3.7 Scénario PDR

Cette section décrit en détail le scénario de la revue préliminaire de conception qui fait part du scénario CoDes.

3.7.1 Objectifs de PDR

Le cycle de vie d'un programme aéronautique européen est divisé en plusieurs étapes [ABC+2001]. Les résultats de chaque étape doivent être obtenus et validés avant le commencement de la suivante. Pour cette raison, chaque phase est généralement clôturée par un rendez-vous afin de vérifier que les objectifs de la phase sont atteints, avant de commencer la phase suivante. Ces rendez-vous constituent les revues de programme et sont composés par l'évaluation des documents, des produits, des processus et de l'organisation. Les buts d'une revue de programme sont:

- obtenir une évaluation d'un point de vue indépendant des accomplissements du programme,
- mettre en évidence des difficultés et des risques potentiels afin de les réduire,
- évaluer l'état d'avancement,
- mettre à disposition les documents et les produits dans une base de référence,
- et finalement fournir du soutien pour prendre la décision du commencement de la phase suivante ou bien la révision de la phase actuelle.

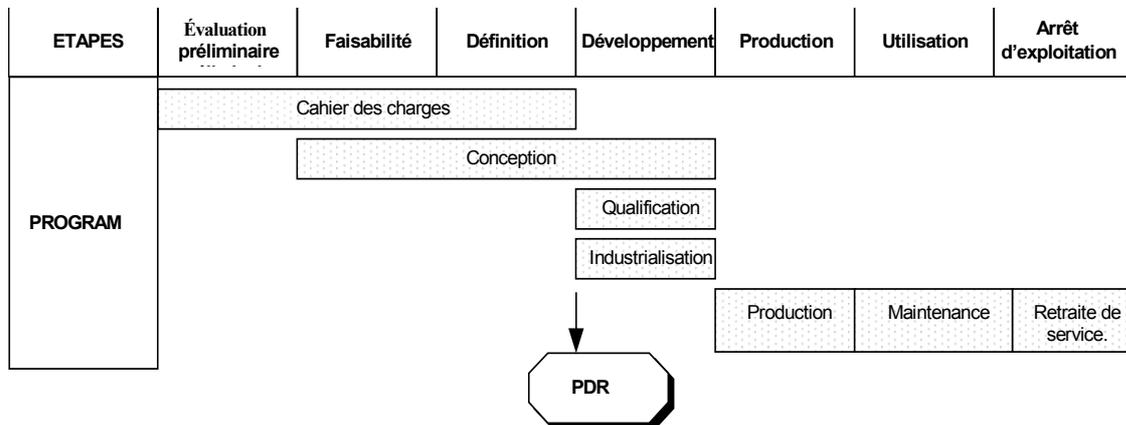


Figure 3-6 Cycle de vie du programme de processus d'ingénierie

La revue de conception préliminaire (PDR) [ABC+2001] est un de ces rendez-vous. Elle se situe à la fin de la phase de définition et avant l'étape de développement, voir Figure 3-6.

La revue de conception préliminaire du véhicule de transfert automatique (ATV) est un exemple représentatif. Cette revue a des objectifs spécifiques pour vérifier:

- Si la conception préliminaire de l'ATV est conforme aux conditions prédéfinies et l'approbation de l'ESA vis-à-vis des caractéristiques industrielles et des plans.
- Si la fabrication, l'assemblage, l'intégration et les plans d'essai sont conformes aux dates de livraison prévues,
- Si le plan de vérification sous-jacent aux équipements est proportionné.

3.7.2 Acteurs

Les acteurs impliqués dans le processus PDR sont divisés en trois groupes: le comité d'examen, le comité de revue et l'équipe de projet.

Le Comité de revue est responsable d'examiner les documents soumis, détecter des désaccords et d'élaborer de documents d'anomalie RID (*Review Item Discrepancy*). Il prend en charge aussi la recommandation des actions au comité d'examen.

Le président du comité de revue est un expert non impliqué dans le programme, et donc son opinion est impartiale. Il est responsable d'organiser et de conduire la revue du programme.

Le secrétaire du comité de revue est responsable de préparer la revue, de rassembler et de distribuer la documentation requise. Le secrétaire peut être aidé par plusieurs personnes pour effectuer des tâches administratives et logistiques (lettres d'invitation et de convocation, réservation de salles, gestion de la reproduction de documents, etc.).

Dans un grand programme international, tel que l'ATV, le comité de revue est généralement composé par plusieurs groupes (5 à 10), chaque groupe traite les documents correspondant à un thème particulier. Un groupe se compose de 8 à 12 critiques. Les critiques sont des experts externes au projet et les principaux acteurs car ils ont en charge l'analyse des documents et la résolution des désaccords éventuels.

Les membres de l'équipe de projet (8 à 12) répondent aux questions et aux commentaires posés par le comité de revue. Selon ces réponses, le comité de revue génère des recommandations au comité d'examen. L'équipe de projet organise l'envoi des RID et la récupération des réponses.

Le comité d'examen décide de mettre en application ou de rejeter les recommandations et les actions proposées par le comité de revue. Il approuve et modifie les documents qui sont à l'étude. Le comité d'examen est composé de représentants des clients avec l'appui des représentants de l'entrepreneur principal. Le conseil fixe la date de limite pour l'application des recommandations et des actions issues du PDR.

Les décisions à prendre dans le programme, e.g. la décision de commencer ou non la phase suivante du programme, sont en dehors de la portée de la revue. La revue a pour objectif principal le rassemblement des éléments qui serviront comme base pour prendre ce type de décisions.

3.7.3 Modélisation de la revue PDR

Nous avons modélisé la revue PDR [ABC+2001] en utilisant le langage IDEF0[Fed1993]. Ce langage permet d'exprimer clairement les activités, les participants, et les résultats d'un processus. IDEF0 appartient à la famille des méthodologies de définition d'intégration pour la modélisation de fonctions (*Integration Definition for Function Modeling* IDEF). La méthode de modélisation de fonctions (*Function Modeling Method*²) IDEF0 est une méthode conçue pour modéliser les décisions, les actions, et les activités d'une organisation ou d'un système. IDEF0 a été dérivé d'un langage graphique bien établi, de l'analyse structurée et de la technique de conception SADT [Sof1981]. Les modèles pertinents d'IDEF0 aident à organiser l'analyse d'un système et à favoriser la bonne communication entre

² Knowledge Based Systems, Inc., "IDEF0 Function Modeling Method," <http://www.idef.com/overviews/idef0.htm>.

l'analyste et le client. IDEF0 est conçu pour saisir et organiser des informations sur les fonctions exécutées par une organisation et leurs corrélations en termes d'entrées, de sorties, de commandes, et de mécanismes. En tant qu'outil de communication, IDEF0 met en valeur la participation de domaine et la prise de décisions expertes de consensus par les dispositifs graphiques simplifiés. En tant qu'outil d'analyse, IDEF0 assiste le concepteur dans l'identification des fonctions qui doivent s'exécuter, de ce qui est nécessaire pour exécuter ces fonctions, de ce que l'actuel système fait correctement, et de ce que l'actuel système fait mal. Ainsi, des modèles d'IDEF0 sont souvent créés lors des premières phases du cycle de développement de système.

La Figure 3-7 montre les entrées, les sorties et les mécanismes correspondant au niveau 0 de PDR.

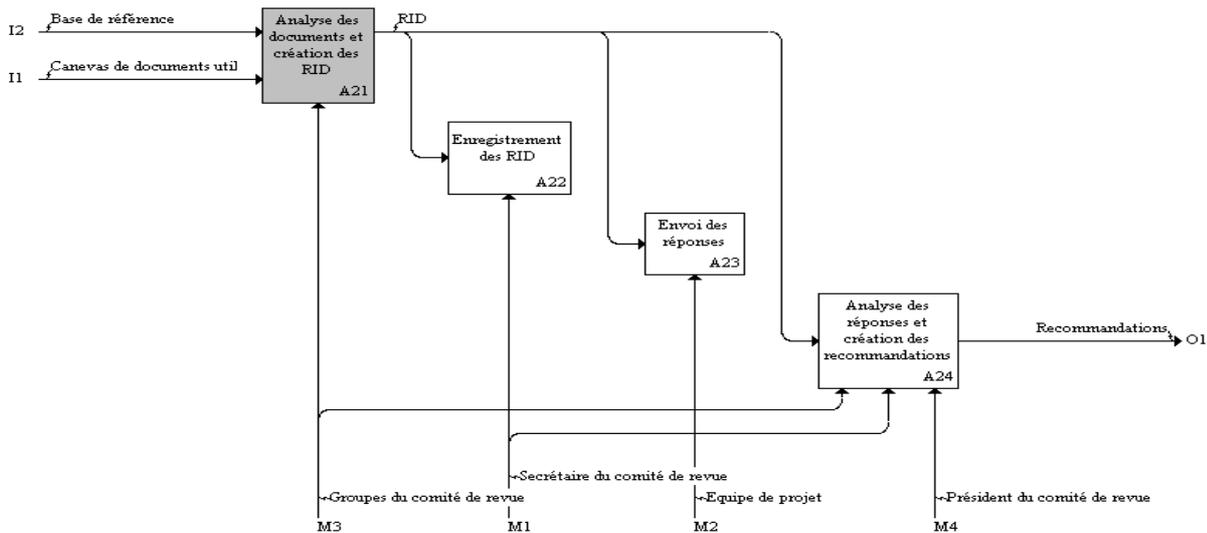
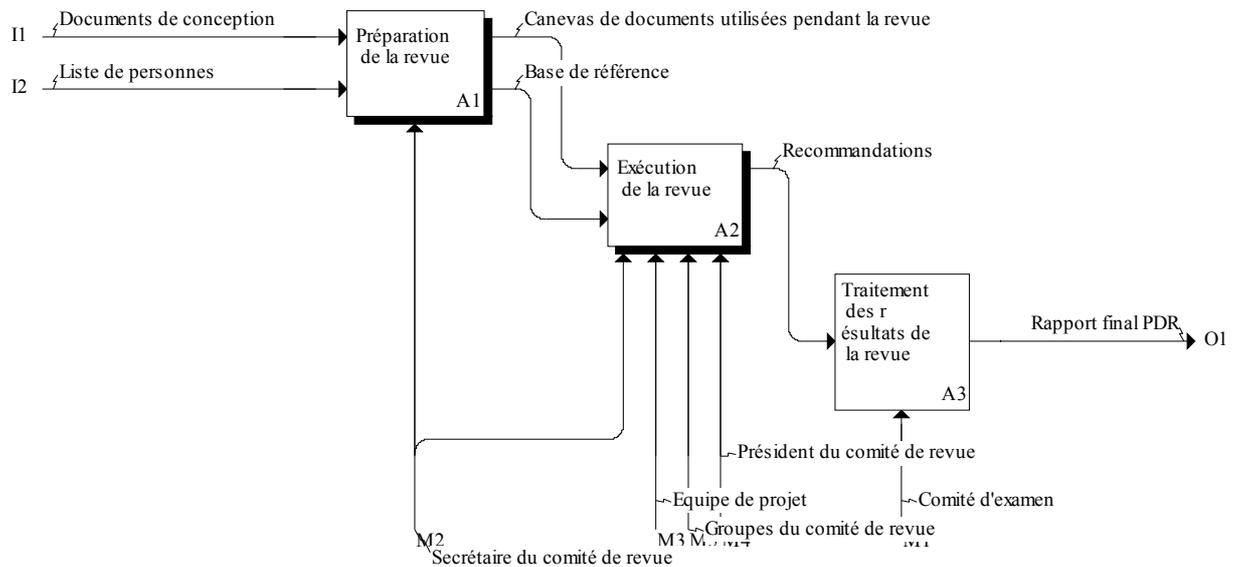


Figure 3-7 Ensemble des mécanismes, des entrées et des sorties de la revue PDR

La revue PDR est composée par trois tâches qui sont préparées et exécutées de manière séquentielle. La Figure 3-8 montre le niveau 1 de la modélisation de la revue en IDEF0. Ces phases sont :

1. Préparation de la revue,
2. Exécution de la revue
3. Traitement des résultats de la revue

Figure 3-8 Activités de la revue PDR-ATV



3.7.3.1 Préparation de la revue

La préparation de la revue correspond à la première tâche de la revue. Cette tâche a pour objectif d'organiser et de configurer la revue. La Figure 3-9 montre les activités de cette phase :

- Constitution des groupes de la revue (comité de revue, équipe de projet),
- Assignation des tâches de la revue à chacun de ces groupes (par exemple en termes de documents à examiner par critique),
- Programmation de la revue : de la réunion pilote à la réunion de fermeture PDR.
- Rassembler et faire disponible la base de référence (ensemble de documents à passer en revue),

Les résultats de cette phase sont:

- Génération du document d'organisation de la revue,
- Production de la base de référence de la revue,
- Création des canevas de documents utilisés pendant la revue (RID, compte rendu de réunions, etc.).

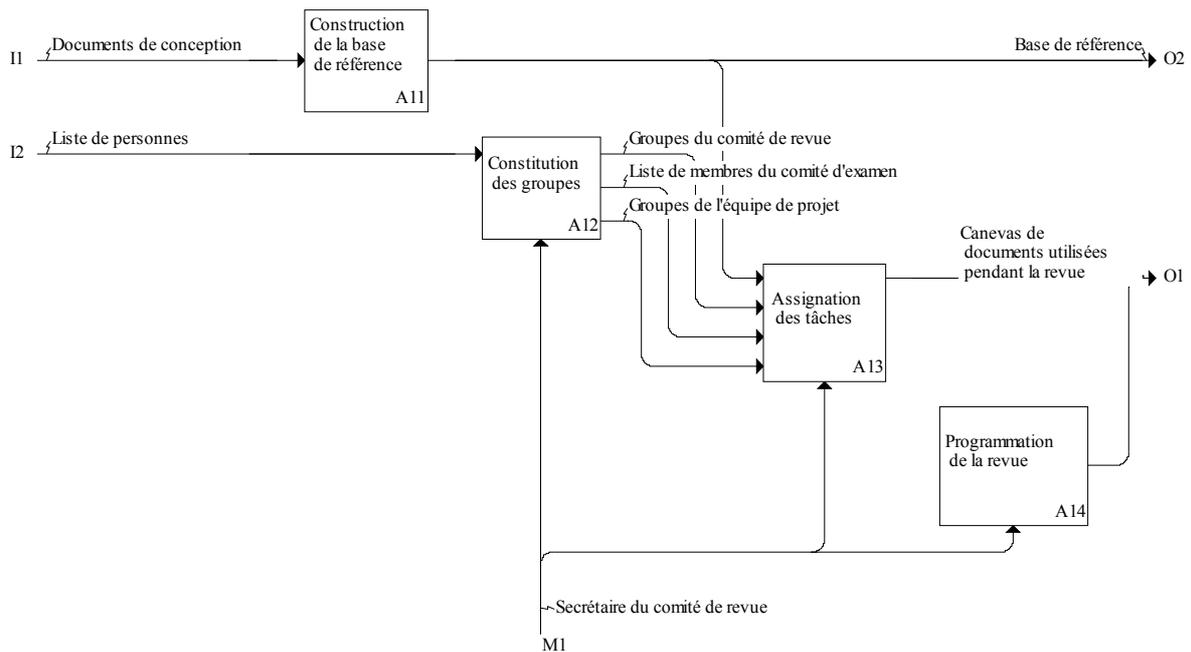


Figure 3-9 Model de la préparation de la revue PDR

3.7.3.2 Exécution de la revue

La deuxième tâche de la revue consiste à son exécution. La Figure 3-10 montre les activités de cette phase :

- Organiser et effectuer la réunion pilote de la revue (pour ATV : 150 à 200 personnes),
- Publier les RID faits par les membres du comité de revue,
- Synthèse et distribution des RID par le secrétaire aux équipes de projet
- Répondre, par les membres de l'équipe de projet aux commentaires et questions posés par le comité de revue,
- publication des recommandations par le comité de revue, avec un dialogue en groupe entre le président du comité de revue, le secrétaire du comité de revue et les groupes du comité de revue,
- présentation des recommandations du comité de revue au comité d'examen.

Pendant cette phase, les réunions effectuées par les groupes du comité de revue sont organisées et peuvent durer de 5 à 7 jours pour chaque groupe. Chaque document est généralement passé en revue par 2 critiques, exceptionnellement par 3 à 5 critiques.

Tandis que les RID correspondent à l'identification d'un désaccord de conception en particulier, les recommandations peuvent produire un grand nombre d'actions de correction. La Figure 3-11 montre les acteurs et les actions effectuées pendant l'exécution de la revue PDR. Dans le cas de l'ATV, 2500 RID et entre 30 à 50 commentaires ont été établis.

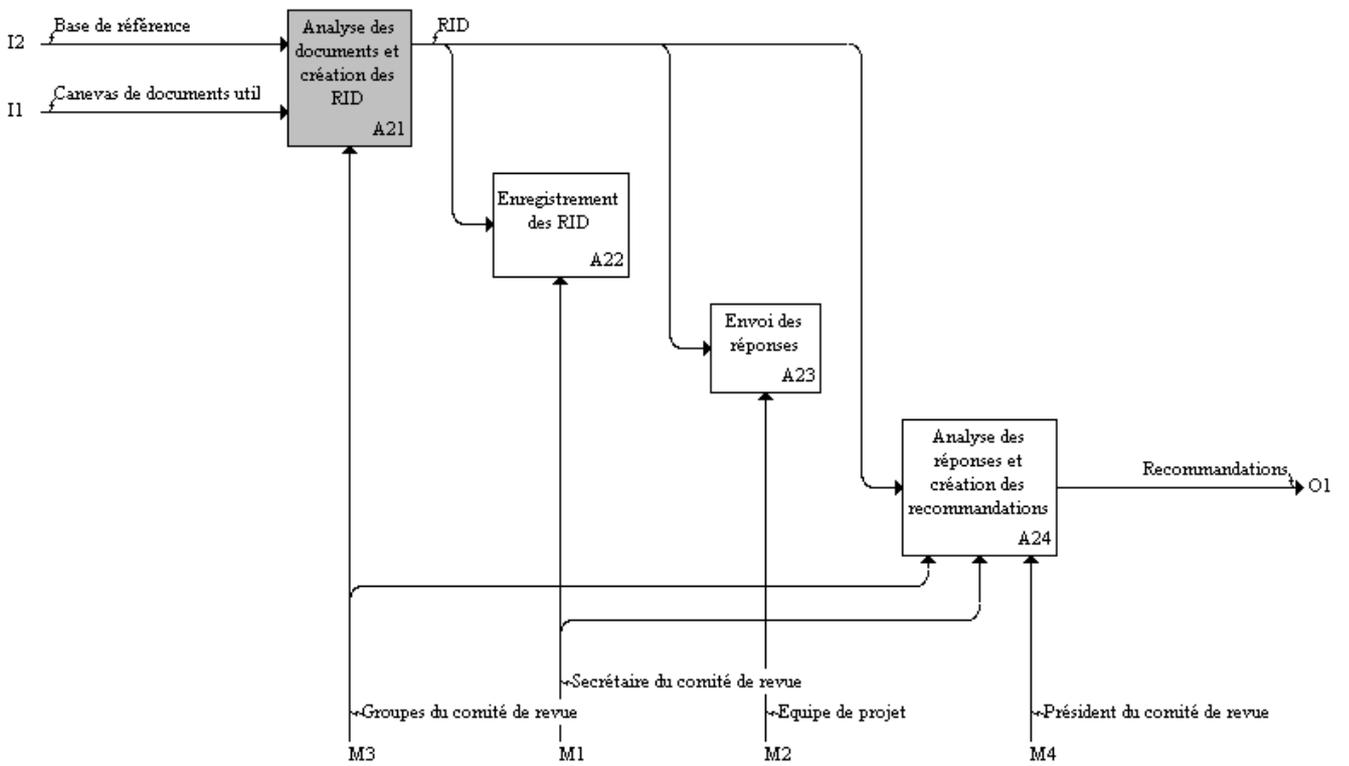


Figure 3-10 Modèle de l'exécution de la revue PDR

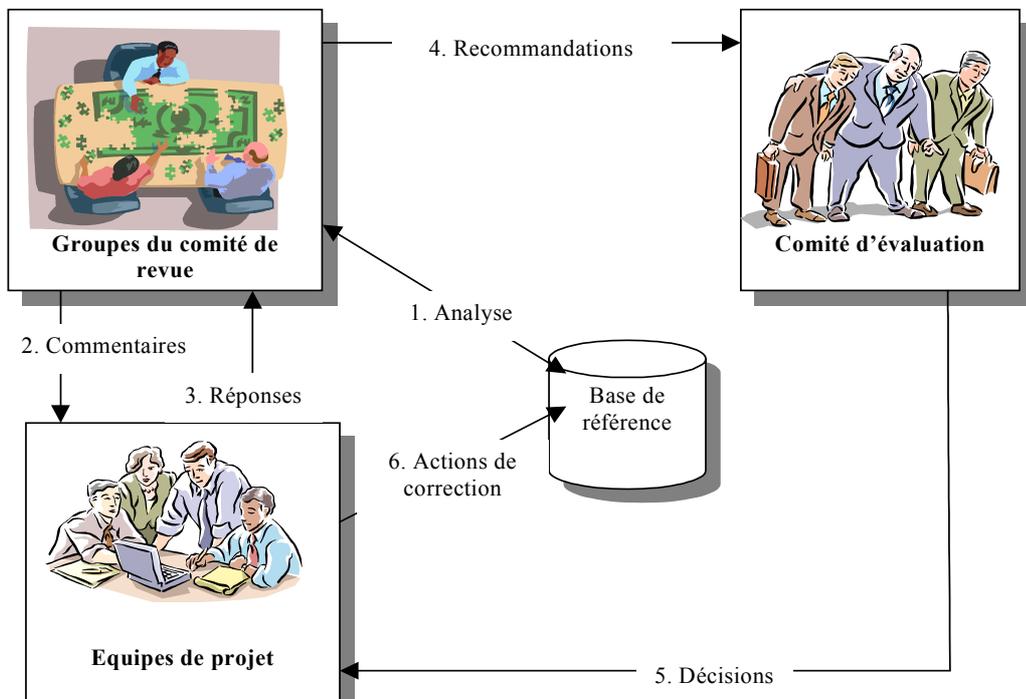


Figure 3-11 Activités et acteurs de l'exécution de la revue PDR

3.7.3.3 Traitement des résultats de la revue

Cette activité constitue la dernière phase de la revue. Elle est constituée par les activités illustrées par la Figure 3-12:

- Publication du rapport final PDR
- Exécution des recommandations et des actions par l'équipe de projet
- Fermeture des RID,
- Archivage des résultats de revue
- Réalisation de la réunion de fermeture PDR.

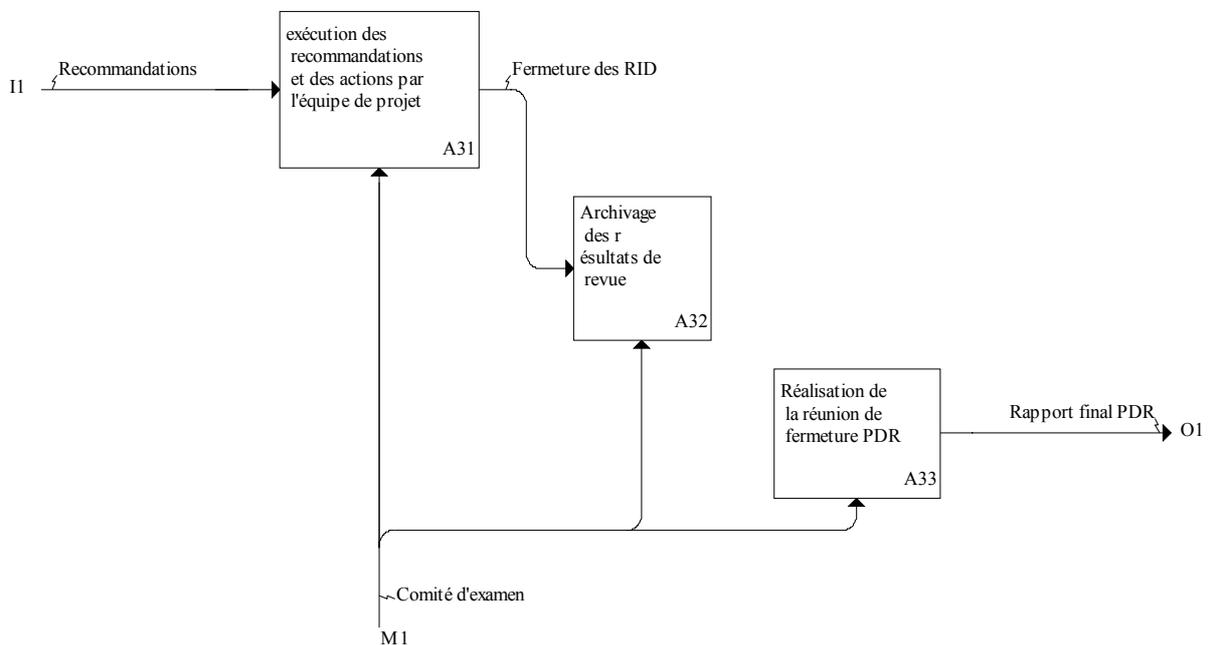


Figure 3-12 Modèle du traitement des résultats de la revue PDR

3.8 Règles de coordination pour un cas de l'ingénierie coopérative distribuée (PDR)

3.8.1 Définition des participants

Nous avons étendu l'ensemble des définitions de base présentées en 3.5 afin de définir les acteurs impliqués dans le scénario PDR. Nous avons défini l'ensemble des critiques qui participent à la revue et qui sont représentés par l'Équation 3.1 Équation 3.26. Ainsi, l'ensemble des participants du scénario PDR est formé par les participants définis en 3.5.1, plus un secrétaire plus l'ensemble des critiques. Cet ensemble est défini à l'Équation 3.27.

$$REVIEWER = \{r_i \mid 1 \leq i \leq n\}$$

Équation 3.26 Définition des critiques

$$P_{CoDes} = P_{basic} \cup \{secretary\} \cup REVIEWER$$

Équation 3.27 Définition des participants du scénario CoDes : PDR

3.8.2 Définition des applications

Le scénario PDR emploie les applications définies en 3.5.1, et en plus il utilise un nouveau type d'application qui prend a charge le partage d'applications. Ce nouveau type est composé par trois types de composants : des serveurs, des clients et un proxy. Ci-après, nous présentons la formalisation de ce nouveau type.

D'abord, nous avons défini un ensemble de composants du type serveur (voir Équation 3.28), un ensemble de composants du type client (voir Équation 3.29). Puis en utilisant ces définitions, nous avons défini l'ensemble du partage d'applications comme la réunion des serveurs, des clients et d'un proxy Cette ensemble constitue la définition de l'outil de partage d'applications et est montré par l'Équation 3.30. Enfin, nous arrivons à la définition des applications du scénario PDR-ATV, illustré par l'Équation 3.31.

$$AST_SERVER = \{ast_server_i \mid 1 \leq i \leq n\}$$

Équation 3.28 Définition des composants serveur pour le partage d'applications

$$AST_CLIENT = \{ast_client_i \mid 1 \leq i \leq n\}$$

Équation 3.29 Définition des composants clients pour le partage d'applications

$$AST_TOOLS = \{ast_proxy\} \cup AST_SERVER \cup AST_CLIENT$$

Équation 3.30 Définition des outils de partage d'applications

$$T_{CoDes} = T_{basic} \cup AST_TOOLS$$

Équation 3.31 Définition des applications du scénario CoDes : PDR

3.8.3 Règles de coordination des participants

Le comportement des participants du scénario PDR suit les règles de coordination définies en 3.5, renforcées par de nouvelles règles supplémentaires. La première extension concerne l'ordre de connexion des participants. En effet, les différentes rôles joués par les participants impose un certain ordre dans leur arrivée à une session. Etant donné que le chairman de la session est le responsable de la session et qu'il effectue les actions de création, de destruction, d'ouverture et de clôture de la session, alors il doit être connecté avant n'importe quel participant. Ceci se traduit par la définition de la règle illustrée par l'Équation 3.32 qui définit que les participants ne peuvent se connecter que si, et seulement si, le chairman l'a déjà fait.

De manière similaire, le secrétaire d'une session doit arriver avant les critiques lors d'une session PDR. Cette contrainte répond aux responsabilités du secrétaire, car il a la charge de rédiger le compte rendu de la session qui contient la liste des participants, les actions effectuées, etc. Cette contrainte est définie par la règle de l'Équation 3.32 qui indique que les critiques ne peuvent se connecter que si, et seulement si, le secrétaire a déjà été connecté.

$$\lambda_1 = \bigwedge_{p \in P_{CoDes}} Pred_{join}(chairman, p)$$

Équation 3.32 Règle de l'ordre de connexion chairman/participant

$$\lambda_2 = \bigwedge_{r \in REVIEWER} Pred_{join}(secretary, r)$$

Équation 3.33 Règle de l'ordre de connexion secrétaire/reviewer

3.8.4 Règles de coordination des applications

Les applications de base présentées dans l'Équation 3.6 sont utilisées dans le déroulement du scénario PDR. Le gestionnaire des sessions présente une nouvelle règle qui définit les conditions d'ouverture d'une session. L'ouverture d'une session du scénario PDR est conditionnée par la présence du chairman, du secrétaire et d'au moins un critique. Ceci est défini dans la règle de l'équation 3.32. Cette règle spécifie que l'ouverture de la session effectuée par l'application de gestion de session doit être précédée par la connexion du participant *chairman*, du participant *secrétaire* et par au moins la connexion d'un participant *critique*.

$$\lambda_3 = Pred((chairman, join), (smt, open)) \wedge Pred((secretary, join), (smt, open)) \wedge \left(\bigvee_{\substack{R \subset REVIEWER \\ |R| \geq 1}} \left(\bigwedge_{r \in R} Pred((r, join), (smt, open)) \right) \right)$$

Équation 3.34 Règle d'ouverture d'une session PDR

L'utilisation du nouvel outil de partage d'applications dans le scénario PDR implique la spécification de son comportement. La figure 3.7 illustre le fonctionnement de l'outil de partage d'applications. La philosophie de fonctionnement de cet outil consiste à connecter les clients au proxy et celui-ci à son tour est relié aux serveurs. Les serveurs fonctionnent comme fournisseurs d'applications. Les clients utilisent les applications partagées par les serveurs. Le proxy effectue la connexion entre un serveur et plusieurs clients. Au moment de changer de serveur d'applications, la connexion entre les clients et le proxy est conservée. En effet, le proxy réalise la commutation de connexion d'un serveur à un autre (lignes pointillées de la Figure 3.7) sans perdre les connexions vers les clients.

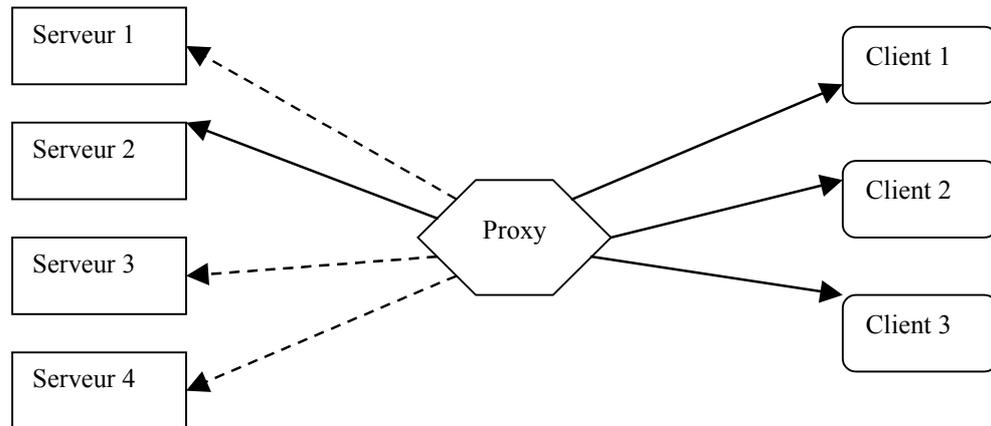


Figure 3-13 Composants de l'outil de partage d'applications

Cette architecture implique la spécification d'un ordre de démarrage et d'arrêt entre les trois types de composants. En suivant la fonctionnalité décrite ci-dessus, nous remarquons que les composants clients ne peuvent être démarrés que si le proxy a été démarré. Nous pouvons remarquer aussi que les serveurs doivent être enregistrés chez le proxy. Ceci implique alors qu'avant de démarrer le proxy il doit y avoir au moins un serveur démarré. Cette contrainte est définie dans l'Équation 3.35. Cette règle spécifie que le proxy ne peut être démarré que si et seulement s'il y a au moins un serveur qui a été démarré.

$$\lambda_4 = \bigvee_{\substack{T \subset AST_SERVER \\ |T| \geq 1}} \left(\bigwedge_{t \in T} Pred_{start}(t, ast_proxy) \right)$$

Équation 3.35 Règle de démarrage des serveurs

A cause de la relation entre les serveurs et le proxy, si les serveurs sont arrêtés alors le proxy doit être aussi arrêté. L'Équation 3.36 spécifie que le proxy doit être arrêté automatiquement si l'ensemble des serveurs sont arrêtés. Cet arrêt du proxy entraîne l'arrêt automatique des clients, car s'il n'y a pas de proxy alors il est inutile que les connexions vers ces clients soient conservées. L'Équation 3.37 décrit que les clients doivent être arrêtés après l'arrêt du proxy.

$$\lambda_5 = \bigwedge_{t \in AST_SERVER} ImPred_{stop}(t, ast_proxy)$$

Équation 3.36 Règle d'arrêt des serveurs

$$\lambda_6 = \bigwedge_{t \in AST_CLIENT} ImPred_{stop}(ast_proxy, t)$$

Équation 3.37 Règle d'arrêt des clients

Les composants clients ne peuvent être lancés que si le proxy a été mis en route. Ceci est représenté par la règle de l'Équation 3.38.

$$\lambda_7 = \bigwedge_{t \in AST_CLIENT} Pred_{start}(ast_proxy, t)$$

Équation 3.38 Règle de démarrage des client

L'ouverture d'une session de coopération marque le début du travail collaboratif. A ce moment les participants commencent à utiliser les applications. La règle de l'Équation 3.39 spécifie que les applications sont démarrées automatiquement après l'ouverture de la session. De manière analogue, à la fin de la session, les applications doivent être arrêtées. Ceci est défini par l'équation 3.39.

$$\lambda_8 = \bigwedge_{t \in T_{CoDes}} ImPred((smt, open), (t, start))$$

Équation 3.39 Règle de démarrage des applications

$$\lambda_9 = \bigwedge_{t \in T_{CoDes}} ImPred((smt, close), (t, stop))$$

Équation 3.40 Règle d'arrêt des applications

Le travail collaboratif d'une session PDR est effectif seulement entre les actions d'ouverture et de fermeture de la session, c'est à dire pendant son état de coopération. En dehors de cet état, les applications ne peuvent pas être démarrées (voir l'Équation 3.41) et les participants ne peuvent pas être connectés (voir l'Équation 3.42).

$$\lambda_{10} = \bigwedge_{t \in T_{CoDes}} Inhib((smt, close), (t, start))$$

Équation 3.41 Règle d'interdiction de démarrage des applications

$$\lambda_{11} = \bigwedge_{p \in P_{CoDes}} Inhib((smt, close), (p, join))$$

Équation 3.42 Règle d'interdiction de connexion des participants

3.9 Mise en œuvre du scénario PDR en utilisant un outil workflow

Dans cette section nous proposons de présenter une spécification Workflow pour compléter les fonctionnalités de notre service de coordination pour la coordination des phases asynchrones des activités de coopération.

Nous avons effectué la mise en œuvre du modèle de la revue PDR-ATV décrit ci-dessus en utilisant TeamFlow³. TeamFlow est un logiciel Workflow basé sur les formulaires ou documents, c'est à dire, les formulaires ou les documents sont remplis au fur et à mesure qu'ils circulent entre les participants. En fonction du rôle de chaque participant, celui-ci remplit une partie du formulaire. Nous avons créé pour PDR-ATV les formulaires, le diagramme de flux des activités et la liste de rôles. La Figure 3-14

³ <http://www.teamflow.com>

illustre l'outil *Flow Planner* de TeamFlow qui prend en charge de la définition du digramme de flux des activités, ainsi que de la définition des rôles et de l'association des formulaires. Le diagramme du flux des activités défini pour représenter le scénario PDR-ATV est montré dans la Figure 3-15.

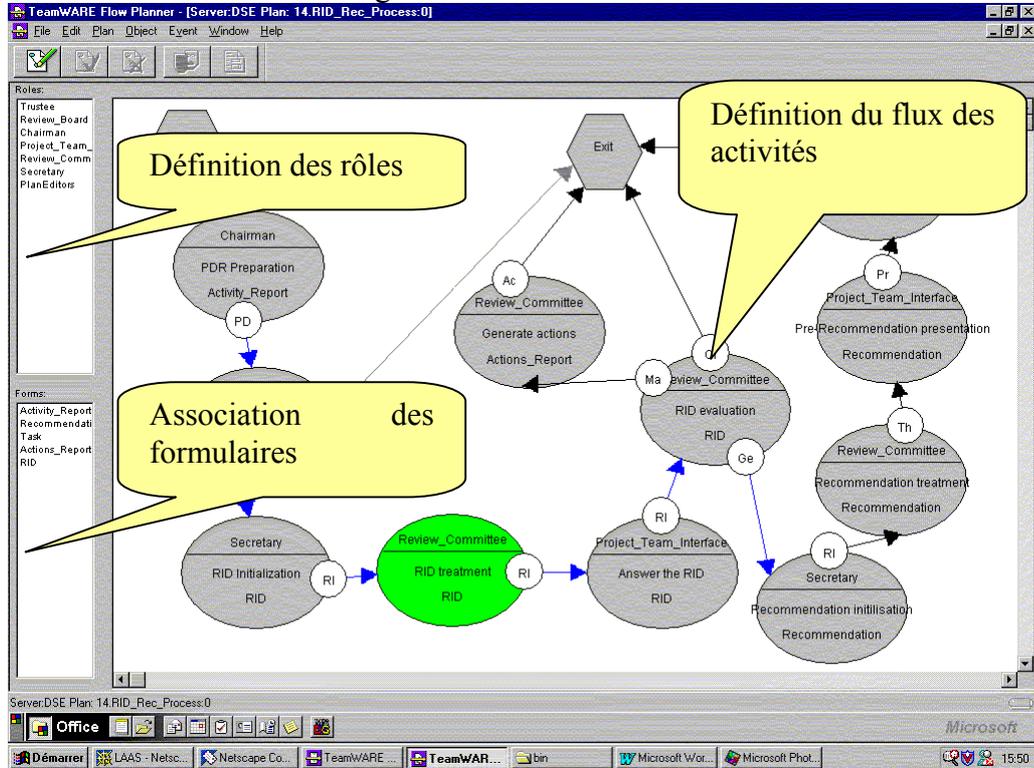


Figure 3-14 Définition des rôles, des formulaires et du diagramme de flux des activités

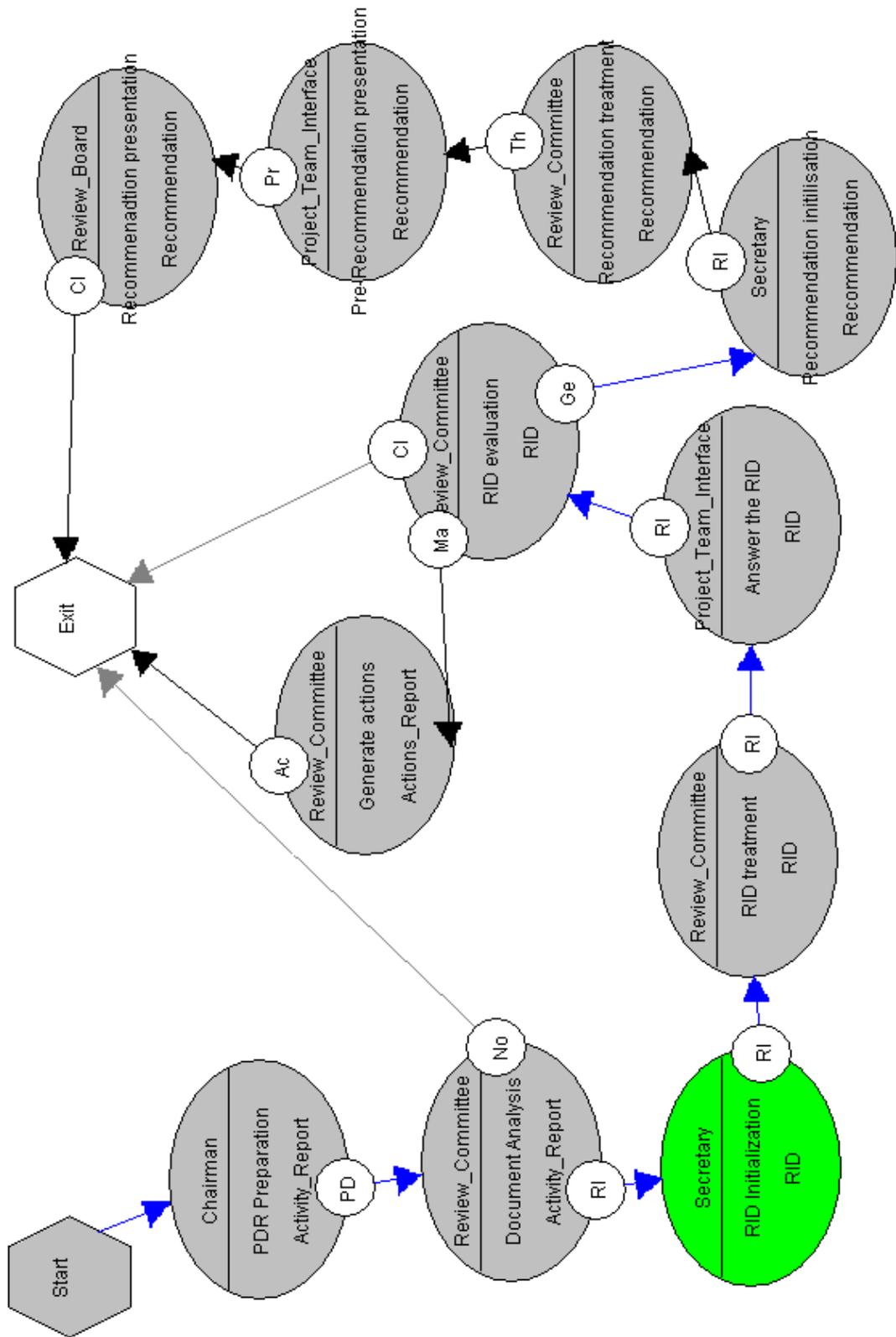


Figure 3-15 Digramme du flux des activités du scénario PDR

Nous avons créé trois formulaires : le formulaire des documents d'anomalie RID, le formulaire des actions issues d'un RID, et le document des recommandations associées à un RID. La Figure 3-16 montre la structure du formulaire RID. Celui-ci est composé par trois parties, en haut les données d'identification du RID, à gauche les éléments pour la description de l'anomalie et à droite les champs pour répondre ou ajouter des commentaires vis-à-vis du RID. Un RID circule entre plusieurs participants selon la définition faite en 3.7.3.2 et 3.7.3.3.

D'abord un RID est créé et initialisé par la demande d'un groupe de comité de revue. Le secrétaire affecte un numéro d'identification unique. On présentera notre mise en œuvre par le traitement d'un RID issu de la revue réelle PDR qui a été effectuée en 2000. Le RID appartient au domaine de la télémétrie et sa création par le secrétaire est montrée par la Figure 3-16.

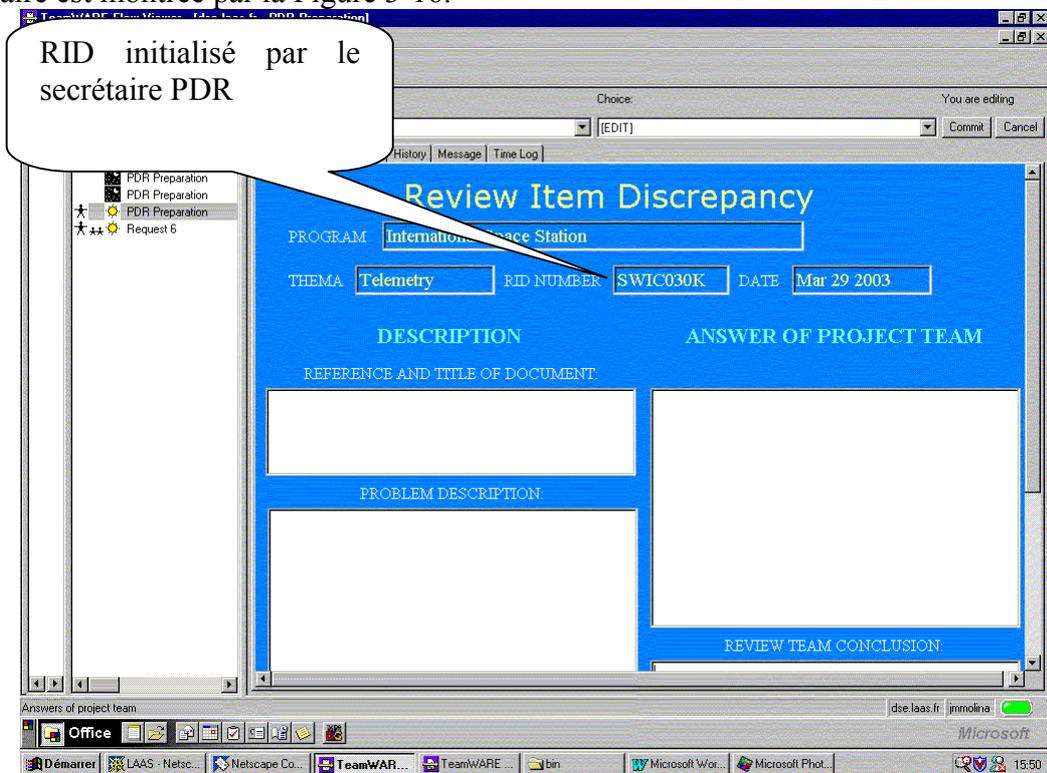


Figure 3-16 Structure du formulaire RID

Une fois initialisé le RID est rempli par le groupe du comité de revue qui a demandé sa création. Le groupe cite le document révisé où l'anomalie a été trouvée et aussi fait une description de l'anomalie. La Figure 3-17 montre les champs qui sont complétés par le groupe du comité de revue.

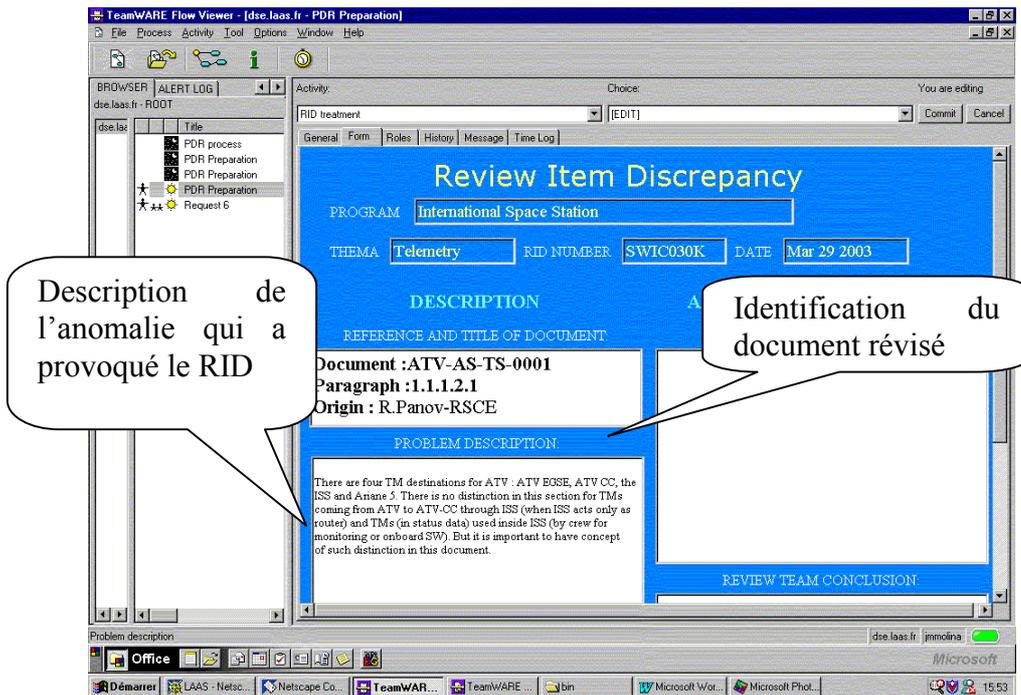


Figure 3-17 Remplissage d'un RID

Le RID, une fois rempli par le groupe du comité de revue, est envoyé à l'équipe de projet. Celui-ci répond ou écrit des commentaires vis-à-vis du RID. Ceci est illustré par la Figure 3-18

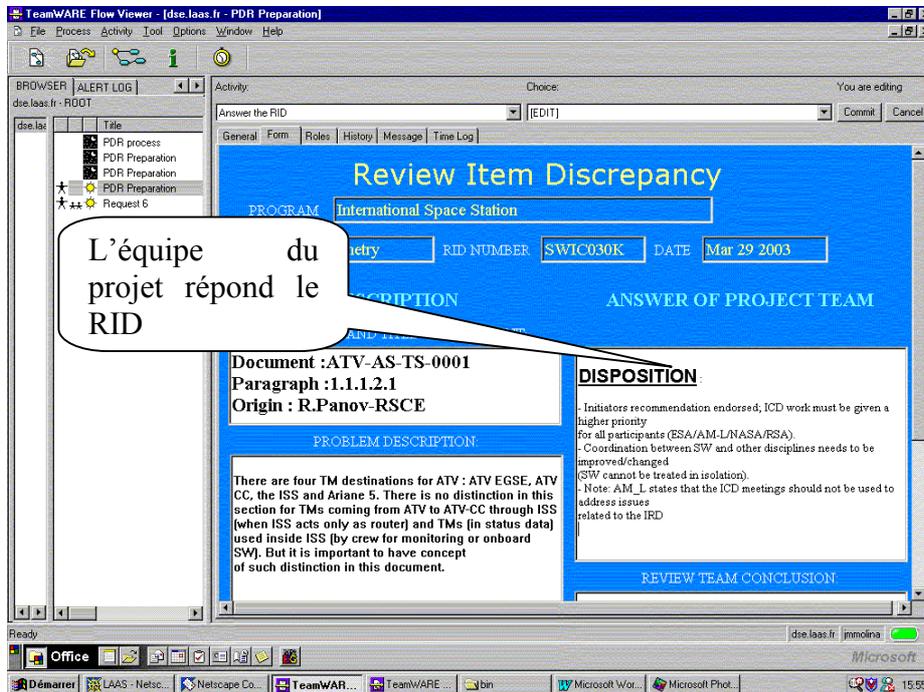


Figure 3-18 Réponse d'un RID

Un RID peut occasionner une série d'actions ou bien une recommandation. La Figure 3-19 montre le formulaire défini pour le cas de génération d'un ensemble

d'actions. De manière similaire, la Figure 3-20 montre le formulaire défini pour le cas des recommandations.

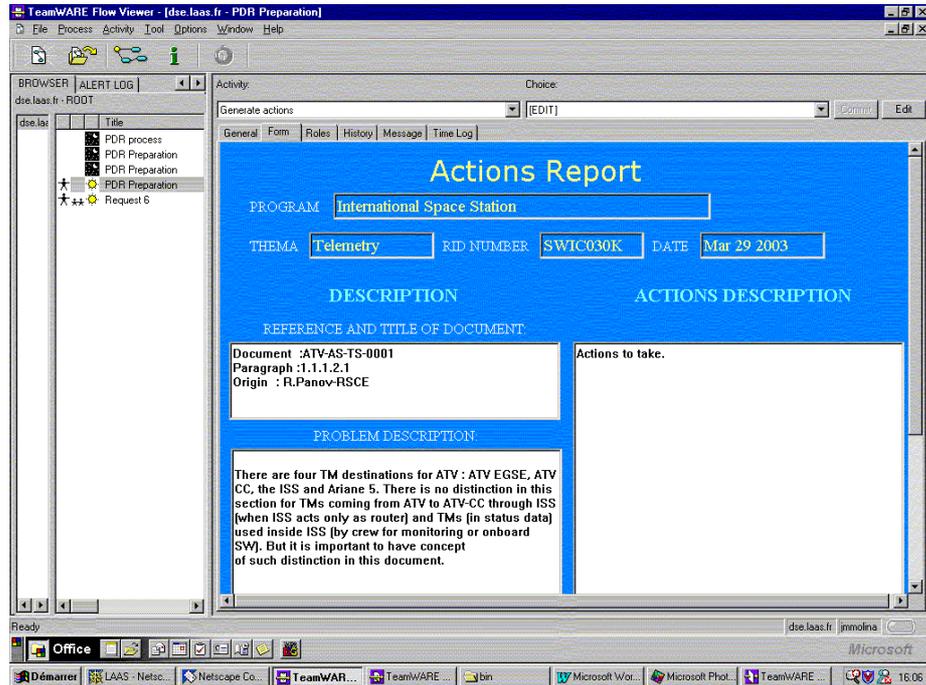


Figure 3-19 Formulaire des actions issues d'un RID

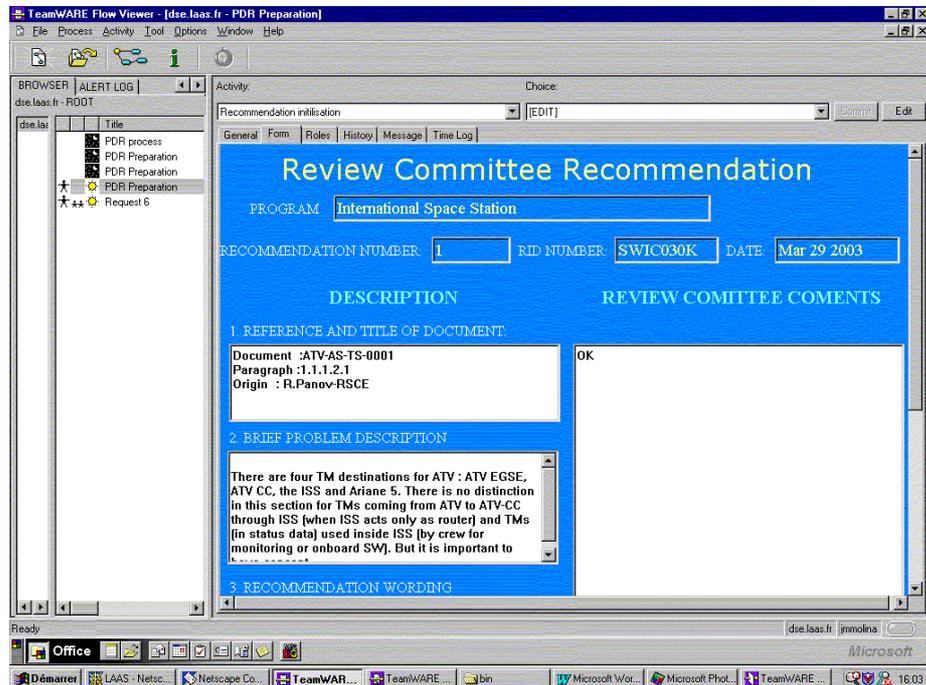


Figure 3-20 Formulaire des recommandations issues d'un RID

3.10 Conclusion

Dans ce chapitre, nous avons proposé un modèle de coordination des interactions pour les session de coopération. Dans le modèle, nous avons défini les

interactions en terme de règles de coordination. Le modèle se base sur la formalisation des sessions de coopération à l'aide des ordres partiels étiquetés et par la définition des règles de coordination par des formules logiques. Le modèle fournit une représentation abstraite et à haut niveau d'une session de coopération, il permet la définition des interactions entre acteurs afin d'éviter des interactions incohérentes et destructives.

Nous avons appliqué notre modèle afin de définir l'ensemble des règles de coordination de base pour des sessions de coopération multi-utilisateurs multi-applications.

Dans ce chapitre, nous avons aussi modélisé un scénario réel d'application. Ce scénario issu de l'ingénierie coopérative distribuée nous a servi comme guide pour prouver l'expressivité de notre modèle. Dans la suite nous l'utiliserons pour tester la conception et la mise en œuvre du modèle.

La formalisation de la coordination des sessions a été conduite en plusieurs étapes des le début du projet DSE. D'abord de façon informelle [MD2002], en phase de spécification des exigences de nos systèmes. Ces spécifications ont été généralisées et définies sous forme de trois relations de dépendances [MD2003].

Ce chapitre correspond à l'aboutissement de l'effort de formalisation basé sur la logique et les ordres partiels [MFD2003].

Chapitre 4. Environnement d'aide à la configuration et à la gestion des sessions multi-applications

4.1 Introduction

Fondé sur les principes de spécification et de modélisation des services pour la préparation et la gestion de sessions de coopération multi-applications présentés dans le chapitre précédent, ce chapitre présente leur conception et leur mise en œuvre. Plutôt que présenter une description exhaustive de la réalisation des toutes les classes et de tous les modules développés, nous avons choisi de présenter les aspects techniques les plus pertinents qui nous ont permis de proposer des services de gestion de sessions ayant les caractéristiques suivantes :

- configuration orientée rôles des sessions
- architecture hybride [Tro1997] (partialement centralisée, partialement distribuée) qui supporte la connexion des participants retardataires
- communication orientée événements assurant la mise à jour de l'état global en permettant l'évolution parallèle coordonnée des composants en mode WYSIWIS¹.

La description intégrale de la conception et de la réalisation de services Responsibility and User Management Service et du service Session Management Service est disponible dans les rapports suivants [Bas2001, Bau2001, BCD+2001, DRN+2001].

Le chapitre est divisé en cinq sous-sections : deux premières sous-sections réservées à la conception des services RMS et SMS suivies de trois sous-sections qui présentent les aspects plus pertinentes de leur mise en œuvre.

¹ What You See Is What I See

4.2 Préparation des sessions coopératives

Comme il a été présenté en 3.2.1, avant de lancer une session coopérative, il faut la configurer. La configuration d'une session correspond à sa préparation hors activité de groupe. La configuration inclut la définition de la liste des participants, la liste des applications à employer, la date et heure du début, la durée, etc.

La configuration d'une session est strictement liée au type de coopération. Les sessions explicites sont configurées par un ensemble de paramètres bien connus à l'avance et en plus sont plutôt fermés c'est à dire qui n'acceptent pas de changements ultérieurs. Par contre, la configuration de sessions implicites se résume à la préparation des applications potentiellement utilisées.

4.2.1 Structuration de groupes ouverts

Sur la base de la classification des sessions de coopération présentée en 2.4.2, on distingue deux catégories de structuration :

- Les sessions avec des groupes ouverts : où les utilisateurs peuvent joindre le groupe de collaboration impliqué dans la session à tout moment. Aucune information sur les utilisateurs n'est nécessaire avant de programmer les sessions. Dans les gestionnaires de sessions existants, la solution la plus simple qui a été adoptée c'est la création des groupes d'utilisateurs identifiés avec leurs adresses IP. Les utilisateurs doivent découvrir les sessions existantes avant de joindre le groupe impliqué. Ils sont toujours admis en collaboration et aucun mécanisme d'invitation n'est prévu pour le lancement de la session. De telles solutions ont été adoptées dans le travail de [DOM1997, BCF+1997, CGJ+1998] où des groupes sont identifiés en utilisant la définition de l'ensemble d'outils qui sont utilisés et [TP1999] qui considère la définition implicite de groupe par l'identification des objets qui sont partagés par les participants.
- Les sessions avec des groupes fermés : cette deuxième catégorie suppose que l'identité des participants potentiels est connue avant le lancement de la session coopérative. Cette solution permet aux mécanismes d'invitation d'être mis en application mais représente l'inconvénient que les groupes de participants construits ne permettent pas l'ajout des participants après leur définition. De telles solutions ont été adoptées dans le travail de [DV1997].

Le modèle de structuration des groupes génériques ouverts proposé a les avantages des deux catégories. Il permet à des sessions publiques d'être contrôlées comme dans la première catégorie permettant l'ouverture du groupe. Mais il permet également au mécanisme d'invitation d'être mis en application comme dans la deuxième catégorie. Dans notre approche, différents types d'attributs sont associés aux participants pour permettre de s'adresser à eux ou de les identifier d'une manière unique (par leur identité) ou d'une manière générique (par leur propriétés).

4.2.2 Diagramme des classes RMS

Un diagramme de classes UML est une collection d'éléments de modélisation statique (classes, paquets, etc.), qui montre la structure d'un modèle. Il fait abstraction des aspects dynamiques et temporels. Pour représenter un contexte précis, un

diagramme de classes peut être instancié en diagramme d'objets. Le diagramme de la Figure 4-1 décrit les classes définies afin de spécifier les caractéristiques du service *Responsibility and User Management Service*. Chacune des classes est décrite dans l'une des sous-sections suivantes.

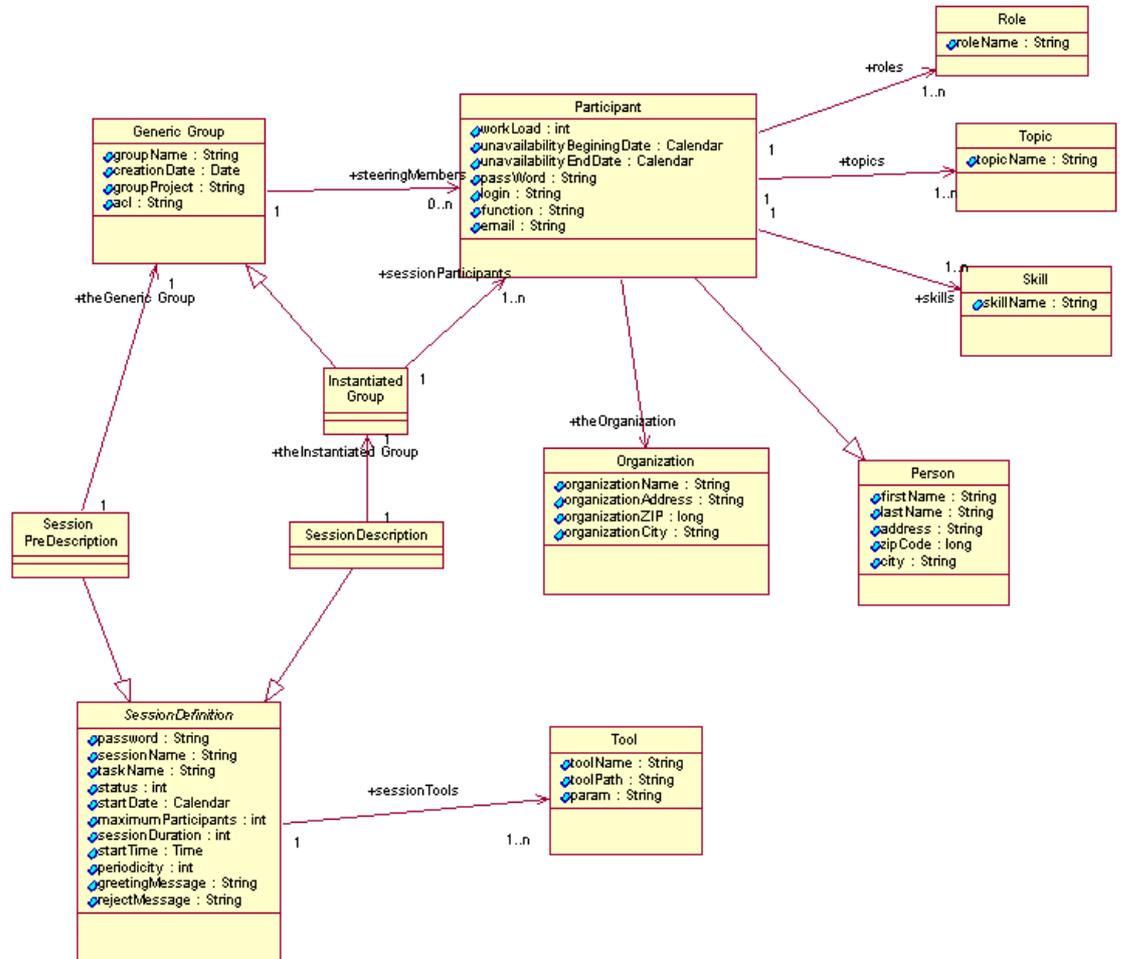


Figure 4-1 Diagramme de classes pour la configuration des sessions RMS

4.2.2.1 Définition des participants

Les principaux composants d'une session sont les participants. Les participants sont les personnes qui coopèrent dans la session. Nous avons défini une classe *person* qui réunit les attributs d'une personne physique, par exemple : nom, prénom, adresse personnelle, âge. On considère que chaque personne travaille dans une *organisation*, celle-ci est décrite par le nom, l'adresse, le ZIP et la ville.

Afin de distinguer les participants qui sont connectés depuis le même site, nous avons défini des classes qui manipulent des propriétés particulières à chaque participant. La classe *skill* définit les compétences du participant. Pour distinguer les divers participants (pouvant avoir les mêmes fonctions) dans le groupe, nous avons introduit la classe *role* qui définit les rôles que le participant peut jouer. Enfin une

troisième classe des propriétés supplémentaires, appelée *topic* est définie pour différencier les thèmes d'intérêt du participant.

4.2.2.2 Définition des groupes de participants

Afin des réunir un ensemble de participants, on a créé deux genres de groupes: les groupes génériques (*Generic Group*) et les groupes de cas (*Instanciated Group*). Les groupes génériques permettent aux participants d'être identifiés par leurs qualifications (compétences, rôles et thèmes). Les groupes de cas permettent aux participants d'être identifiés par leur identité.

La classe des groupes génériques est composée par les données d'identification du groupe telles que le nom du groupe, la date de création, le projet associé et les droits d'accès. Cette classe ne contient pas de liste nominative des participants, mais plutôt une liste des clauses logiques. Chaque clause représente le besoin d'un participant dans le groupe ayant un profil qui correspond à la liste des compétences, des rôles et des thèmes décrits dans la clause. La clause est composée par une expression logique qui met en relation les trois types de qualification exposés ci-dessus et des opérateurs logiques « *et* » et « *ou* » [DD2000]. Ainsi on peut exprimer des besoins du type, « il est nécessaire à un participant de posséder dans ses compétences l'appartenance au comité de révision, ainsi que le rôle du chairman et dans ses thèmes de spécialité des connaissances sur la propulsion ou l'énergie solaire ». Il faut noter que cette classe comporte aussi une liste de participants explicitement nommés (*steering Members*) et qui désigne une liste de participants obligatoires pour la réunion du groupe, par exemple dans la composition d'un groupe générique pour représenter une soutenance, le groupe ne sera valable que s'il est composé au moins par un candidat et par les membres du jury.

La classe des groupes de cas hérite de la classe des groupes génériques les données d'identification, et substitue la liste de clauses par la liste nominative des participants. Pour effectuer cette transition, il faut remplacer chaque clause par le nom d'un ou plusieurs participants. En effet, chaque clause désigne un ensemble potentiel des participants qui satisfait les contraintes décrits par la clause, il faut alors sélectionner un participant de cet ensemble afin de passer vers un groupe de cas.

4.2.3 Configuration des sessions de coopération

En suivant la définition des sessions de coopérations énoncée en 2.4.1 et en utilisant les groupes génériques décrits ci-dessus, une session de coopération multi-applications est constituée d'informations générales, de la définition d'un groupe de participants et de la définition d'un ensemble d'applications. La Figure 4-1 montre, une classe *SessionDefinition* qui groupe les informations générales d'une session qui incluent le nom de la session, l'heure et la date de début, la durée, le nombre maximum de participants, le mot de passe, etc. Ensuite ces caractéristiques communes sont héritées par les classes de préconfiguration et de définition de sessions.

En fonction du type de groupe utilisé, on peut créer soit la préconfiguration d'une session (classe *SessionPreDescription*) dans le cas d'utilisation d'un groupe générique, soit la description d'une session (classe *SessionDescription*) dans le cas d'utilisation d'un groupe de cas. La première classe sert à décrire des configurations

génériques de sessions, qui peuvent par la suite être réutilisées. La deuxième classe décrit des instances concrètes de sessions.

La différence entre ces deux classes réside dans la composition des listes des participants et des outils. Dans la classe de préconfiguration, la liste de participants est désignée par un groupe générique et donc par un ensemble des contraintes qui décrivent les profils de participants. Cette classe est aussi composée d'une liste des applications à employer dans la sessions, mais cette liste ne contient pas les noms des applications mais plutôt une liste des types d'applications. Cette liste générique est similaire, dans sa structure, à la liste des participants de la classe des groupes génériques. En utilisant une telle structure on peut exprimer des besoins du type : pour cette session on utilisera un outil de vidéoconférence, un outil de partage d'applications et un outil de messagerie instantanée. La classe de description contient, quant à elle, une liste nominative des participants et des outils, c'est à dire qu'on trouve dans les deux listes les noms des participants et des applications spécifiquement énumérées. Le passage de la classe *SessionPreDescription* vers *SessionDescription* consiste à énumérer la liste d'applications explicitement.

4.3 Gestion de sessions coopératives multi-applications

4.3.1 Diagramme des classes SMS

La Figure 4-2 illustre l'ensemble de classes créées pour la définition du service SMS. Du fait que le service SMS a un caractère distribué, nous avons prévu la conception des classes en utilisant un approche client-serveur. Les classes serveur (à gauche du diagramme) ont en charge la définition de l'ensemble de mécanismes fournis par SMS. Les classes clients (à droite du diagramme) sont associées aux classes des interfaces graphiques d'accès distant permettant aux utilisateurs d'accéder aux services SMS.

Les mécanismes fournis par SMS résident dans la définition de trois services : la classe *SessionManagementServiceFactory*, dénoté SMSF, la classe *SessionManagementService*, dénotée SMS et la classe *ReponseCollectorService*, dénotée RCS.

La classe RMSF fonctionne comme une fabrique d'objets de type SMS et RCS. A chaque session de coopération correspond la création d'un couple d'objets SMS-RCS. Chaque couple contient l'ensemble des mécanismes capables de gérer complètement une session de coopération. SMSF assure l'initialisation et la terminaison symétrique des objets SMS et RCS associés à chaque session.

L'abstraction des sessions de coopération multi-applications est réalisée par la classe SMS. Chaque objet SMS gère une session, il est donc responsable de la définition des mécanismes pour ouvrir, et clôturer la session. Il est responsable de permettre aux participants de se joindre, de quitter la session et de s'échanger le rôle du chairman entre eux. SMS offre des possibilités de communication synchrones et asynchrones entre les participants connectés. Il est capable de démarrer et d'arrêter les applications utilisées pendant la session. Enfin, SMS joue un rôle principal dans l'architecture du projet DSE, car c'est l'unique responsable de la génération des événements relatifs à la gestion en ligne des sessions, des participants et des

applications. Ces événements sont capitaux pour le fonctionnement des autres services et sous-systèmes de l'environnement DSE.

La classe RCS est le service responsable de la gestion des réponses des participants à l'invitation de la session.

Nous avons aussi défini les classes *SessionState* et *AwarenessService*. L'état courant d'une session est composé de plusieurs éléments qui ont été groupés dans la définition de la classe *SessionState*. La classe *AwarenessService* a été définie comme API pour la communication des événements produits par le service SMS aux autres systèmes. Nous avons adopté une API générale pour la diffusion des événements pour rendre invisible les éventuelles évolutions dans le support sous-jacent à la diffusion de ces événements (CORBA, Java, etc.).

Le reste des classes illustrées dans la Figure 4-2 ont été déjà définies précédemment, elles ont été dessinées afin de rendre plus complète la description de la conception du service SMS.

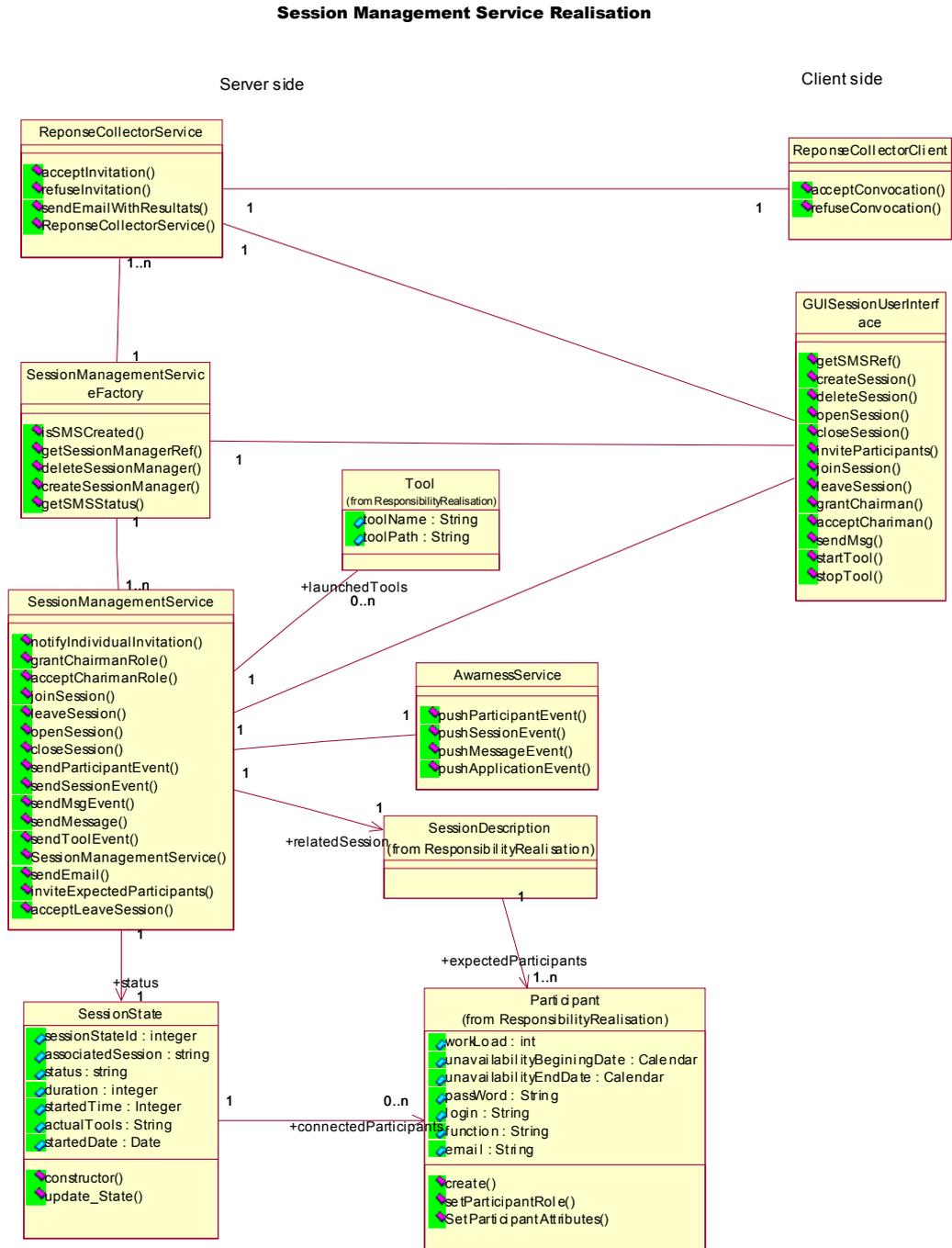


Figure 4-2 Diagramme de classes pour la gestion des sessions de coopération multi-applications SMS

4.3.1.1 Primitives de service de la classe *Session Management Service*

Cette classe représente l'abstraction des sessions de coopération multi-applications. L'ensemble d'objets SMS présents dans l'environnement est géré par un

objet de type SMSF. Le Tableau 4-1 présente la définition de cette classe. Chaque objet SMS contient les attributs suivants :

- Un attribut de type SessionDescription. Cet attribut constitue une description de la session de coopération effectuée par le service RMS. A travers cet attribut, l'objet SMS peut accéder à la description de tous les éléments qui composent la session, e.g., la liste de participants, la liste d'applications, l'identificateur de la session, etc.
- Un attribut SessionState. Cet attribut garde l'état courant de la session. Il est utilisé par SMS pour déterminer la liste des participants connectés, la liste des applications démarrées, la date et l'heure effectives du démarrage de la session, etc.

Les méthodes de cette classe se regroupent en quatre types. D'abord, nous trouvons les opérations relatives à la manipulation de la session, à savoir, les méthodes d'ouverture et de clôture d'une session. Ensuite, nous trouvons les opérations pour gérer les actions des participants regroupant : la méthode qui effectue l'initiation de la liste de participants définis dans la description de la session, les méthodes qui permettent la connexion et la déconnexion des participants et les méthodes pour échanger le rôle de chairman entre les participants. Puis, nous trouvons les opérations qui effectuent la notification des événements au service de notification awareness. Un événement est le résultat de l'occurrence d'une action effectuée par le gestionnaire de session, par les participants ou par les applications. Ces méthodes de notification sont invoquées après l'occurrence d'une action qui provoque un changement ou une mise à jour dans l'état actuel de la session.

Classe SessionManagementService	
Attributs	
relatedSession :SessionDescription	Cet attribut garde la définition de la session à exécuter.
status :SessionState	Cet attribut contient l'état actuel de la session.
Methodes	
Primitives de manipulation de session	
openSession()	Méthode qui ouvre la session en cours, c'est à dire le début de la phase de coopération.
closeSession()	Méthode qui clôt la session en cours, c'est à dire la fin de la phase de coopération.
getStatus()	Cette méthode retourne l'état actuel de la session.
Primitives de manipulation des participants	
invite()	Méthode qui envoie un email à tous les participants contenus dans la liste expectedParticipants dans la description de la session associée (relatedSession).
joinSession(String participantName)	Cette méthode connecte le participant participantName à la session. Ceci signifie que l'état du participant identifié par participantName passe à <i>connected</i> .
leaveSession(String participantName)	Cette méthode déconnecte le participant participantName de la session. Ceci signifie que l'état du participant identifié par participantName

SessionManagementServiceFactory	
Attributs	
SMS : SessionManagementService	Liste des objets SMS gérée par le service SMSF. La liste peut contenir zéro ou n objets SMS
RCS : ReponseCollectorService	Liste des objets RCS gérée par le service SMSF. La liste peut contenir zéro ou n objets RCS.
Méthodes	
createSessionManager(SessionDescription theSession)	Cette méthode crée un couple d'objets SMS et RCS afin de gérer une session.
deleteSessionManager(SessionDescription theSession)	Cette méthode termine les instances des objets SMS et RCS associés à la session.
isSMSCreated(String sessionName)	Cette méthode renvoie vrai si la session identifiée par le nom sessionName a été créée.
getSessionManager(SessionDescription theSession)	Cette méthode retourne une référence du type SMS de la session identifiée par theSession.
getSMSStatus(String sname)	Cette méthode renvoie l'état actuel de l'instance SMS identifiée par sname.

Tableau 4-2 Classe SMSF

4.3.1.3 Primitives de service de la classe *Reponse Collector Service*

Cette classe complète les mécanismes offerts par la classe SMS afin de fournir un ensemble complet d'opérations de gestion de sessions. La classe RCS a pour objectif principal de traiter toutes les actions relatives à la réception des réponses des participants convoqués à une session. La classe se compose de trois attributs et quatre méthodes (Tableau 4-3) : l'attribut `relatedSession` est un objet qui est obtenu lors de la création d'un objet RCS. Cet attribut représente la description de la session associée à l'objet RCS. Les attributs `accepted` et `refused` sont deux listes qui contiennent les noms des participants qui ont accepté ou refusé, respectivement, l'invitation à la session.

La première méthode, `acceptInvitation`, permet au participant identifié par `partName` d'accepter la convocation à la session. De manière symétrique, la méthode `rejectInvitation`, effectue le refus de l'invitation du participant identifié par `partName` de la session. Enfin, la méthode `sendEmailWithResultats` est utilisée pour envoyer un email au chairman avec un bilan des réponses contenant la liste de participants qui ont accepté, la liste des participants qui ont refusé, et la liste de participants qui n'ont pas encore répondu.

ReponseCollectorService	
Attributs	
<code>relatedSession</code> : SessionDescription	La valeur de cet attribut est initialisée par l'argument du constructeur. Il contient la description de la session.
<code>accepted</code> : Vector	Liste des participants qui ont accepté l'invitation à la session.
<code>refused</code> : Vector	Liste des participants qui ont refusé l'invitation à la

	session.
Méthodes	
acceptInvitation(String partName)	Cette méthode accepte l'invitation à la session du participant identifié par partName .
rejectInvitation(String partName)	Cette méthode refuse l'invitation à la session du participant identifié par partName .
sendEmailWithResultats()	Cette méthode envoie par email au chairman un compte rendu des invitations à la session courante.
MailCollectorServer (SessionDescription asession)	Constructeur de la classe. Il définit un paramètre asession avec la description de la session.

Tableau 4-3 Classe RCS

4.3.1.4 Description de la classe *Reponse Collector Client*

La classe *ReponseCollectorClient* se compose de deux méthodes (Tableau 4-4). La première méthode, *acceptConvocation*, permet au participant qui l'appelle d'enregistrer côté serveur l'acceptation de l'invitation à la session associée par l'objet RCS. De manière symétrique, la méthode *rejectConvocation* permet au participant appelant d'enregistrer côté serveur le refus de l'invitation de la session auprès de l'objet RCS.

ReponseCollectorClient	
Méthodes	
acceptConvocation()	Méthode qui enregistre côté serveur l'acceptation de la convocation.
rejectConvocation()	Méthode qui enregistre côté serveur le refus de la convocation.

Tableau 4-4 Classe ReponseCollectorClient

4.3.1.5 Description de la classe *Session State*

L'état courant d'une session est une structure complexe qui contient l'ensemble des données qui définit le contexte d'une session à un instant donné. Le Tableau 4-5 illustre les différents attributs qui forment l'état courant d'une session.

SessionState	
Attributs	
SessionStateId : integer	Identificateur unique de l'état de la session
AssociatedSession : string	Nom de la session
Status:string	Etat actuel
Duration:integer	Durée de la session.
StartedTime:Integer	Heure de démarrage.
StartedDate:Date	Date de démarrage.
ActualTools:string	Liste d'applications démarrées.
ConnectedParticipants	Liste de participants connectés.

Tableau 4-5 Classe SessionState

4.3.1.6 Description de la classe *Awareness Service*

Cette classe a été définie comme une API pour signaler les événements au service de notification awareness. Nous avons créé cette API pour garantir l'interopérabilité avec différents supports du service de notification car plusieurs implémentations de CORBA ont été utilisées au long du projet DSE. L'API est composée de quatre méthodes (Tableau 4-6) :

La méthode, `pushSessionEvent`, envoie un événement au format IDL CORBA concernant une action relative à l'état de la session, e.g. l'ouverture ou la clôture de la session. La méthode, `pushParticipantEvent`, envoie un événement au format IDL CORBA concernant une action réalisée par une des primitives définies dans le Tableau 4-1 pour la manipulation des participants. La méthode, `pushMessageEvent`, est utilisée par les participants pour envoyer des messages. Ces messages apparaissent dans l'interface graphique de la console de notification et non dans l'interface graphique du gestionnaire de sessions SMS. Enfin, la méthode, `pushApplicationEvent`, prend en charge la notification du démarrage ou d'arrêt d'une application dans la session.

AwarenessService	
Méthodes	
<code>pushParticipantEvent(Structure dEvent ev)</code>	Cette méthode envoie un événement relatif à l'état des participants de la session au service de notification au format IDL CORBA.
<code>pushSessionEvent(Structure dEvent ev)</code>	Cette méthode envoie un événement relatif à l'état de la session au service de notification au format IDL CORBA.
<code>pushMessageEvent(Structure dEvent ev)</code>	Cette méthode envoie un message texte au service de notification au format IDL CORBA.
<code>pushApplicationEvent(Structure dEvent ev)</code>	Cette méthode envoie un événement relatif à l'état des applications de la session au service de notification au format IDL CORBA.

Tableau 4-6 Classe AwarenessService

4.3.2 Diagrammes de séquences UML du service SMS

La Figure 4-3 illustre le détail du cas d'initialisation d'une session. Le chairman emploie le service RMS pour définir et/ou trouver une session. Ensuite, le service RMS envoie une requête de création de session au SMSF, qui, à son tour, crée les objets SMS et RCS associés. L'initialisation de la session chez l'objet SMS provoque l'envoi de l'événement CREATE au service de notification awareness.

Les diagrammes de séquences UML permettent de représenter des collaborations entre objets selon un point de vue temporel, on y met l'accent sur la chronologie des envois de messages. Contrairement aux diagrammes de collaboration, où l'on ne décrit pas le contexte ou l'état des objets, et où la représentation se concentre sur l'expression des interactions. Les diagrammes de séquences peuvent servir à illustrer un cas d'utilisation. L'ordre d'envoi d'un message est déterminé par sa position sur l'axe vertical du diagramme ; le temps s'écoule « de haut en bas » de cet axe. La disposition des objets sur l'axe horizontal n'a pas de conséquence pour la

sémantique du diagramme. Les diagrammes de séquences et les diagrammes d'états-transitions sont les vues dynamiques les plus importantes d'UML. En utilisant ce type de diagrammes, nous avons décrit le déroulement des cas d'utilisation présentés en 3.2.2.1 et servent aussi à détailler la dynamique des sessions présentée en 3.2.2.2.

La Figure 4-4 montre le cas d'utilisation pour annoncer la session. Le diagramme se compose de deux aspects. D'une part, la requête d'invitation effectuée par le chairman est réalisée par l'objet SMS. D'autre part, les participants invités peuvent accepter ou refuser l'invitation en utilisant l'interface graphique du `ReponseCollectorClient`. Les réponses sont sauvegardées dans l'objet `ReponseCollectorService` associé à la session.

La Figure 4-5 illustre le cas de l'ouverture d'une session, et la connexion des participants invités. Le diagramme montre que les participants doivent se connecter, même le chairman, avant d'ouvrir la session. Les participants connectés après l'ouverture seront considérés comme participants retardataires.

La Figure 4-6 présente le cas de déconnexion des participants. La procédure de déconnexion est entreprise par le participant, celui-ci invoque la méthode `leaveSession` dans l'objet SMS de la session. Le chairman et les participants sont informés de l'action de déconnexion par la réception d'un événement LEAVE. Dans certaines conditions, le chairman doit autoriser le départ d'un participant, par exemple quand le participant héberge des applications qui sont utilisées pendant la session.

La Figure 4-7 détaille le cas de la clôture de la session. La clôture d'une session est sous la responsabilité du chairman, celui-ci la réalise par l'appel de la méthode `closeSession` de l'objet SMS. L'objet SMS lance un événement CLOSE dans le canal de communication du service de notification.

La Figure 4-8 montre le cas de la terminaison de la session. La terminaison de la session représente la dernière transition de l'état d'une session. Elle est réalisée par le chairman par l'appel de la méthode `delteSession` chez l'objet SMSF. Cette action représente l'élimination de l'objet SMS qui avait en charge la gestion de la session.

Formalisation de la gestion des sessions multi-applications

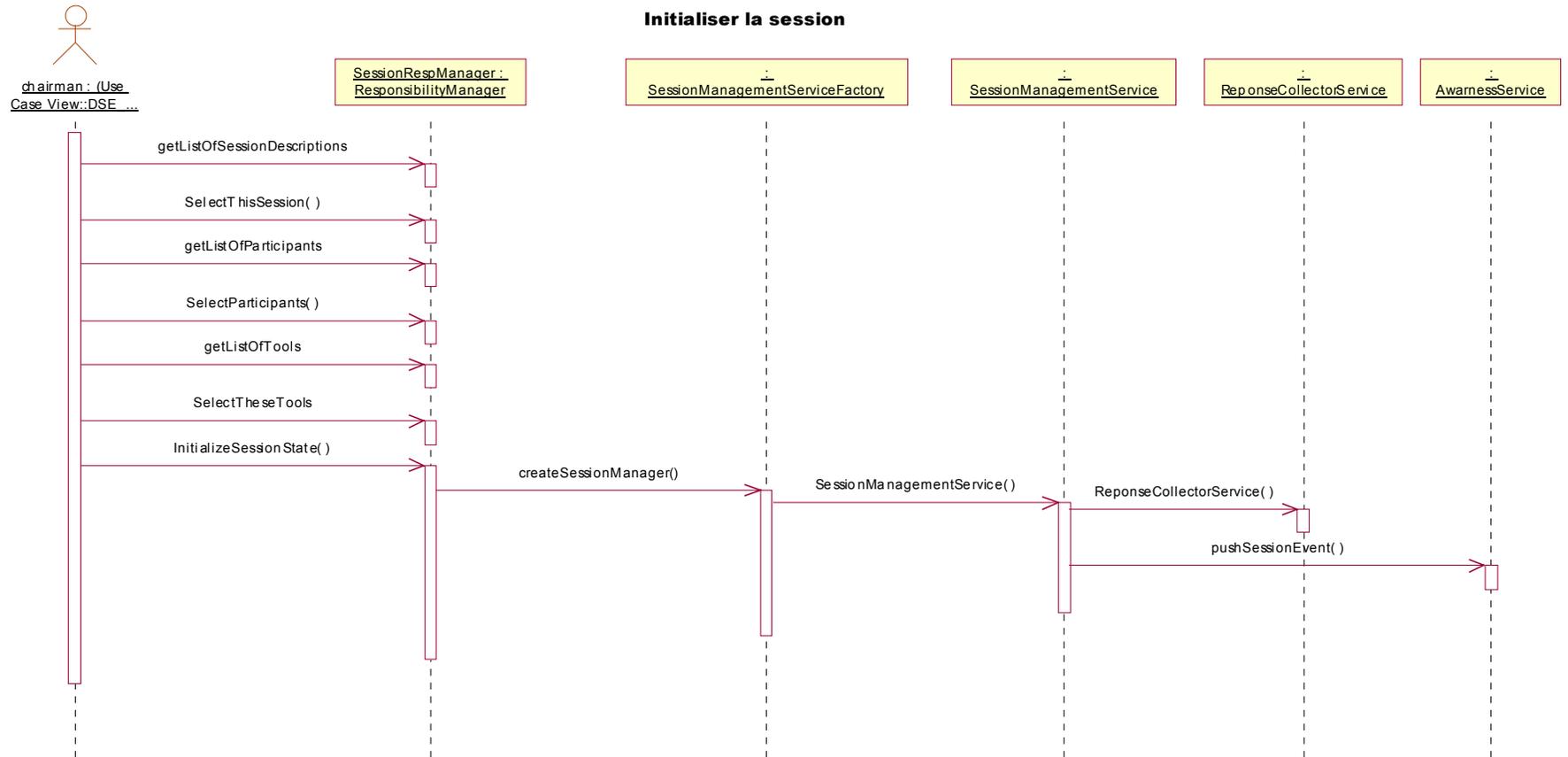


Figure 4-3 Diagramme de séquence pour l'initialisation d'une session

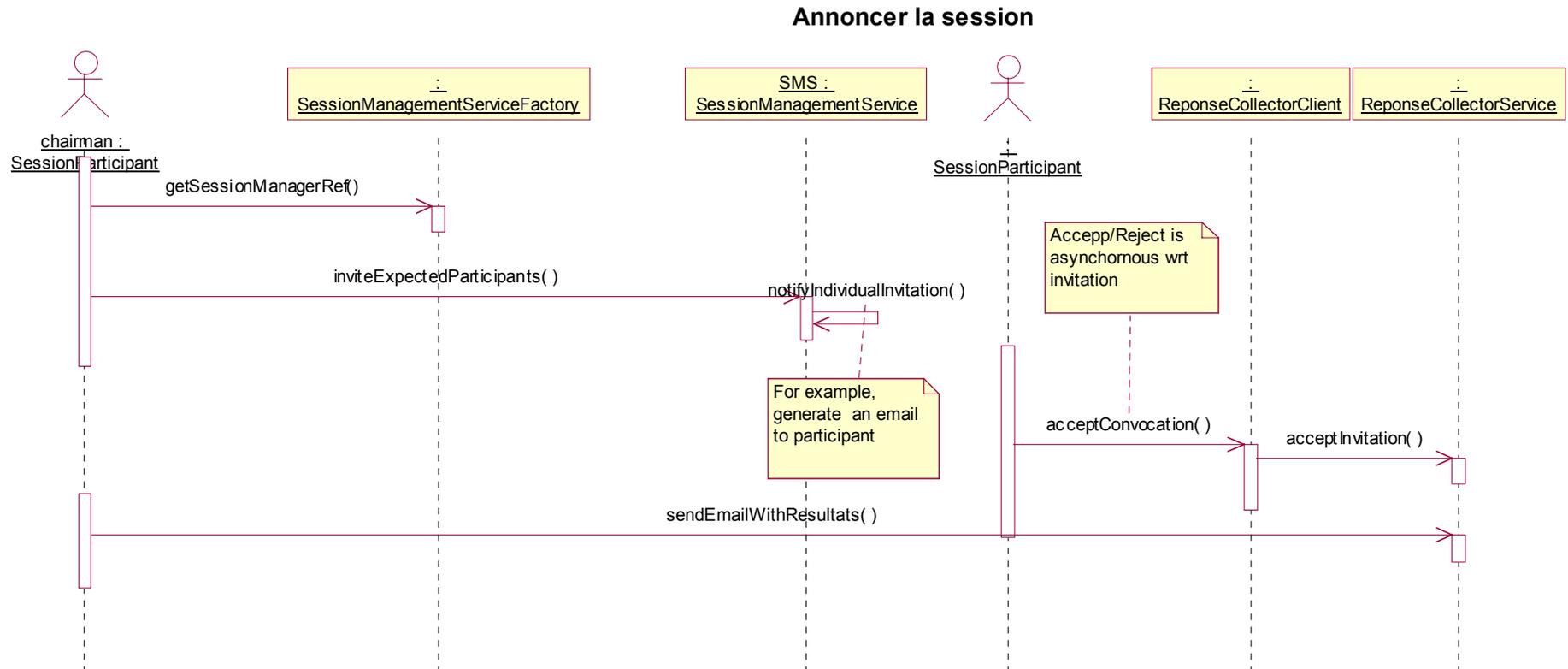


Figure 4-4 Diagramme de séquence pour la convocation des participants à une session

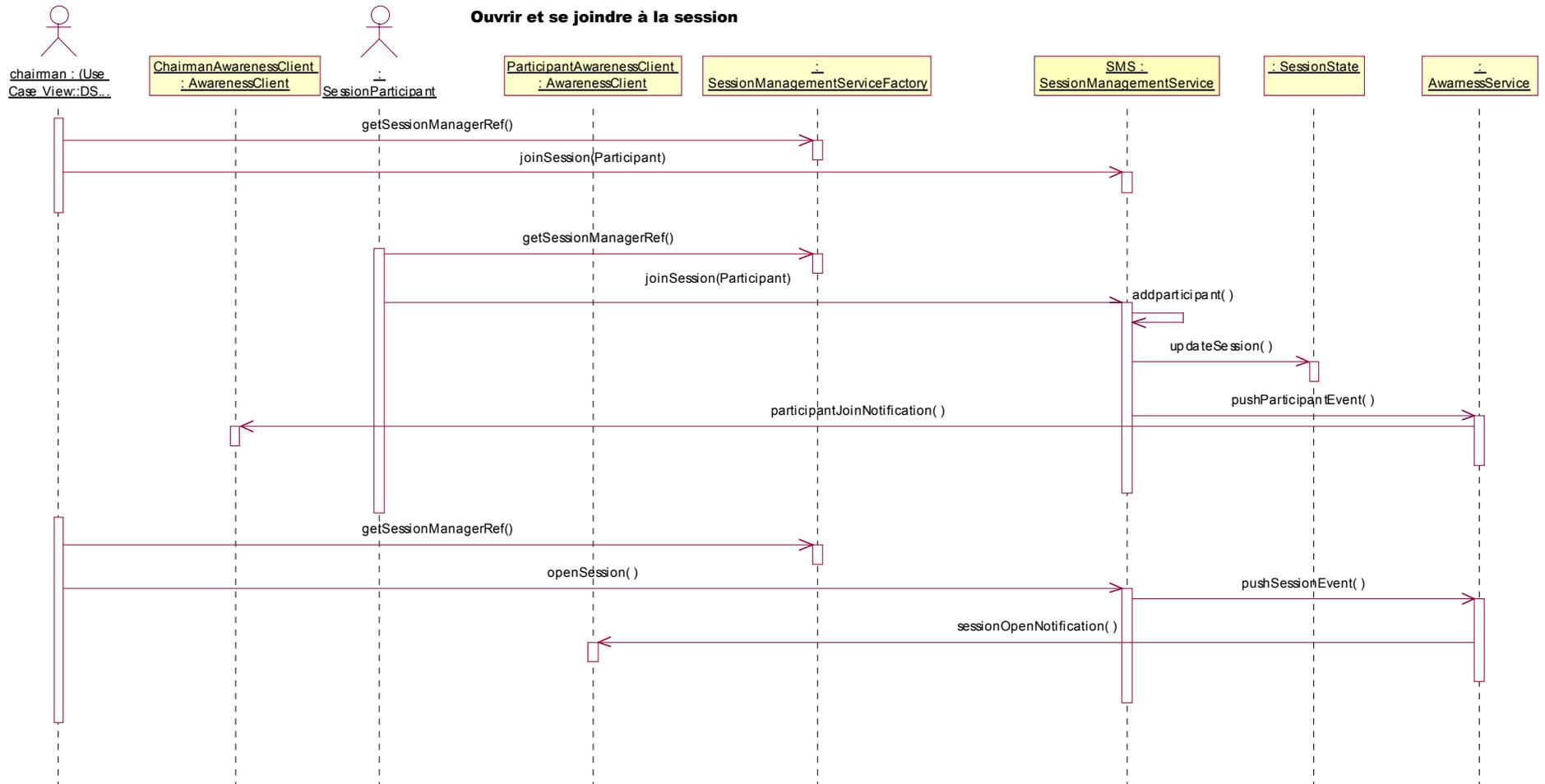


Figure 4-5 Diagramme de séquence pour l'ouverture d'une session et la connexion des participants

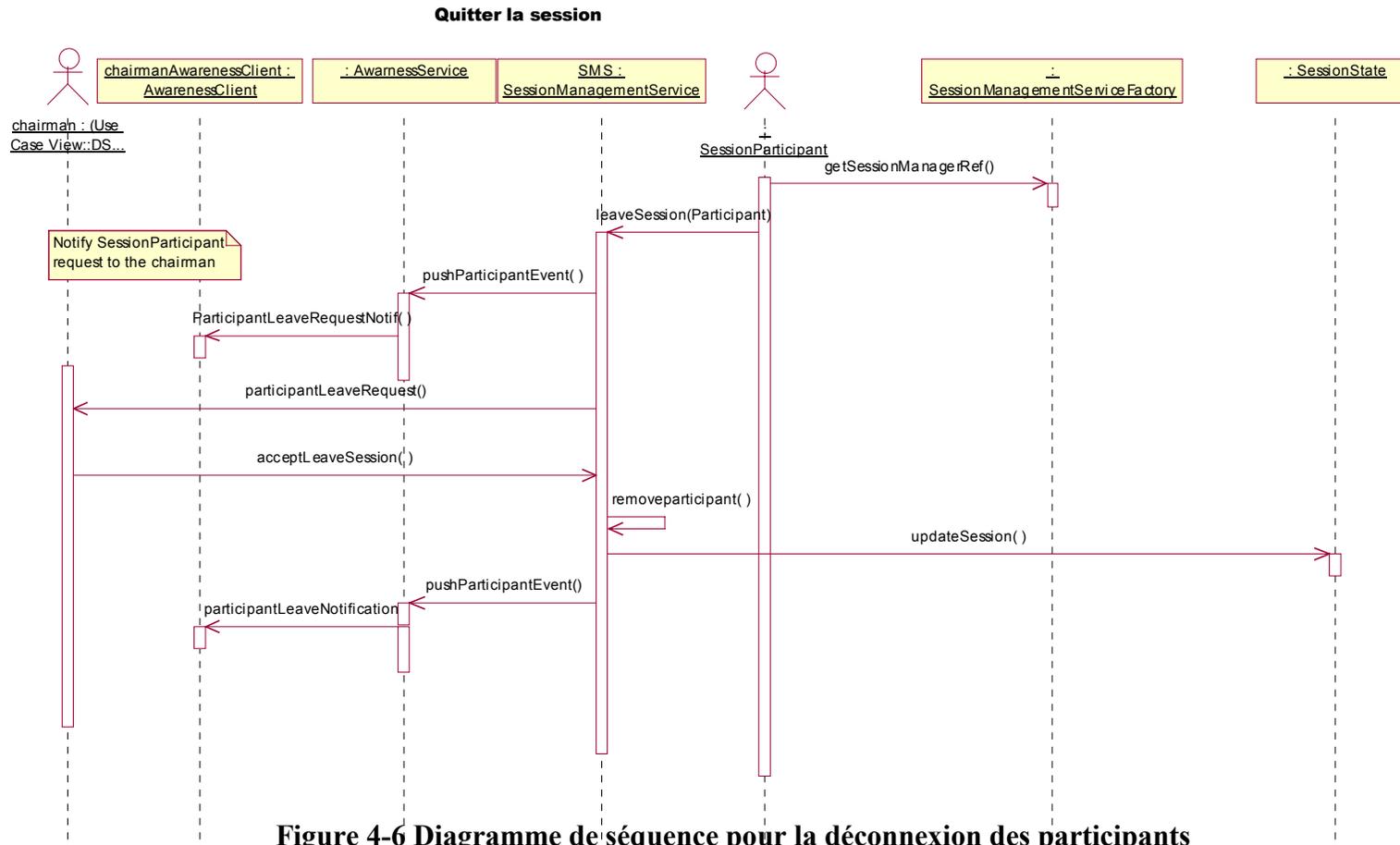


Figure 4-6 Diagramme de séquence pour la déconnexion des participants

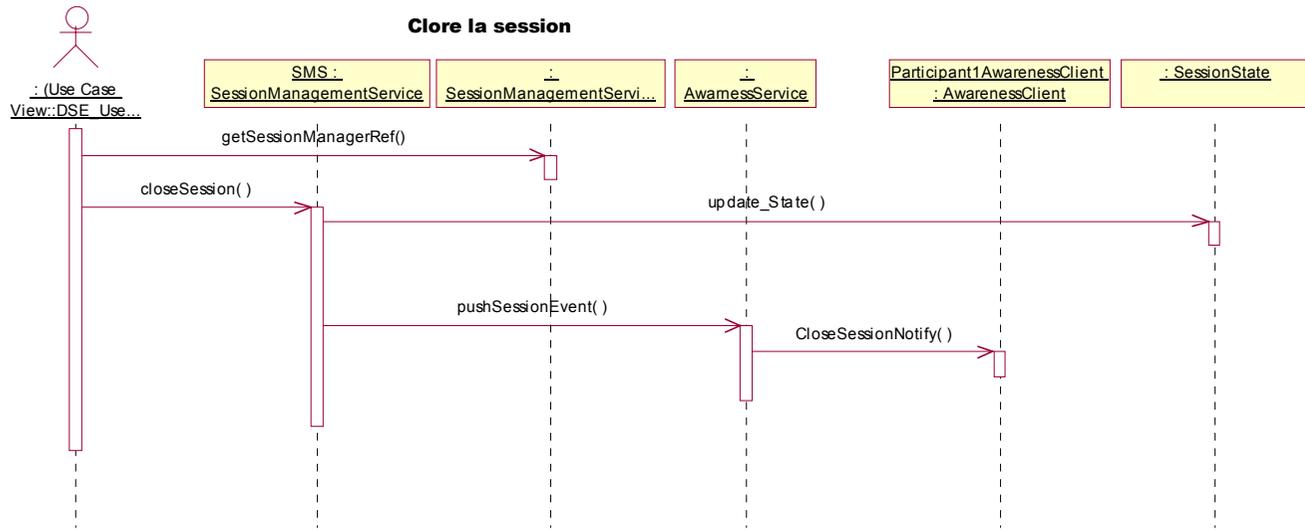


Figure 4-7 Diagramme de séquence pour la clôture d'une session

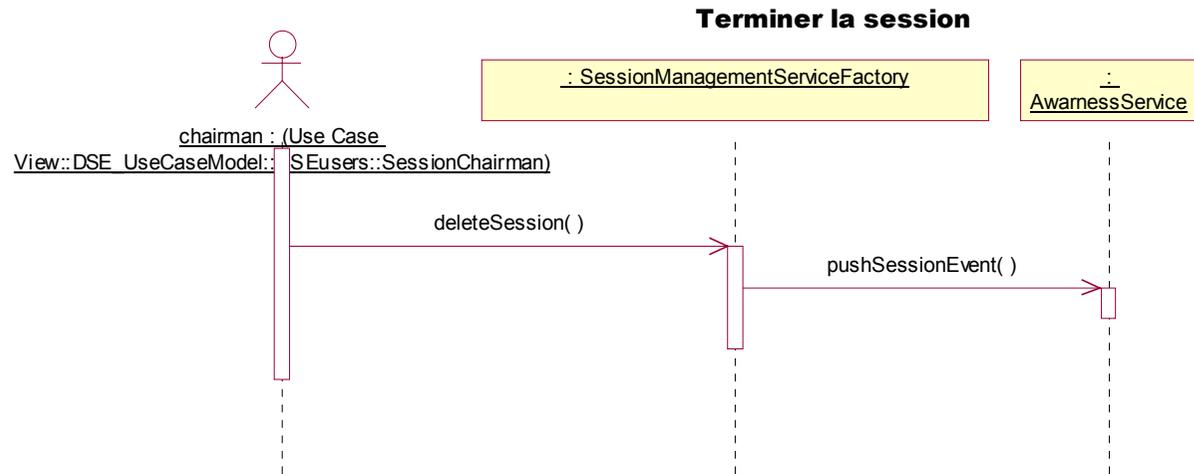


Figure 4-8 Diagramme de séquence pour la terminaison d'une session

4.3.3 Définition des règles de coordination par des contraintes OCL

4.3.3.1 OCL

UML formalise l'expression des contraintes avec OCL¹ [UML1997]. Les contraintes sont des expressions qui précisent le rôle ou la portée d'un élément de modélisation (elles permettent d'étendre ou de préciser sa sémantique).

OCL est largement utilisé dans la définition du métamodèle UML. Ce langage formel est volontairement simple d'accès et possède une grammaire élémentaire (OCL peut être interprété par des outils). Il représente un juste milieu, entre langage naturel et langage mathématique. OCL permet ainsi de limiter les ambiguïtés, tout en restant accessible. OCL permet de décrire des invariants dans un modèle, sous forme de pseudo-code :

- pré et post-conditions pour une opération,
- expressions de navigation,
- expressions booléennes, etc.

4.3.3.2 Règles de coordination de base en OCL

Afin d'intégrer les règles de coordination définies au Chapitre 3 dans le service de gestion des sessions SMS, nous avons employé le langage OCL.

Le Tableau 4-7 illustre l'ensemble des contraintes de coordination spécifiées en OCL issues des 11 règles de coordination de base présentées en 3.5.

Description	Contrainte décrite en OCL
Règles d'appartenance au groupe et de gestion de la cohérence	
Un participant peut se connecter à une session : <ul style="list-style-type: none"> ➤ s'il a été invité (Équation 3.19) ➤ et s'il n'est pas déjà connecté et ➤ si la session n'est pas terminée (Équation 3.20) 	Context SMS : :joinSession(Participant p) : Boolean Pre : <i>SMS.sessionDefinition.expectedParticipant s->includes(p) and Not SMS.connectedParticipants->includes(p) and SMS.oclInState(ANNOUNCED)</i>
Un participant peut se déconnecter d'une session : <ul style="list-style-type: none"> ➤ s'il est actuellement connecté (Équation 3.16). 	Context SMS : :leaveSession(Participant p) : Boolean Pre : <i>SMS.connectedParticipants->includes(p)</i>
Une session peut être terminée : <ul style="list-style-type: none"> ➤ les participants actuels sont déconnectés (Équation 3.21) ➤ et les applications en exécution 	<i>Context SMS : :deleteSession(): Boolean Post : SMS.ConnectedParticipants->forAll(p :Participant SMS.leaveSession(p)) and SMS.processList-</i>

¹ Object Constraint Language

sont arrêtées. (Équation 3.18)	<i>forall(t :ToolsControl SMS.stopTool(t))</i>
Une application peut être arrêtée : <ul style="list-style-type: none"> ➤ si l'application est actuellement exécuté (Équation 3.17) 	<i>Context SMS : :stopTool(ToolsControl aTool) : Boolean</i> <i>Pre : SMS.sessionDefinition.sessionTools->includes(aTool) and SMS.processList->includes(aTool)</i>
Règles de spécification de la portée des communications dans les groupes	
Un message peut être envoyé : <ul style="list-style-type: none"> ➤ si le participant qui l'envoie est connecté (Équation 3.22 et Équation 3.23) 	<i>Context SMS : :sendMessage(Participant p, String header, String msgBody, boolean fullListofConnectedParticipants) : Boolean</i> <i>Pre : SMS.connectedParticipants->includes(p)</i>
Règles de consistance des rôles	
Un participant peut accepter le rôle du chairman de la session : <ul style="list-style-type: none"> ➤ si le participant récepteur est connecté (Équation 3.25) et ➤ s'il peut jouer le rôle de chairman selon son profil et ➤ si la proposition provient du chairman effectif et ➤ après que le chairman effectif ait effectivement cédé le rôle (Équation 3.24) 	<i>Context Participant: :acceptChairmanRole(Participant p) : Boolean</i> <i>Pre : self.oclInState(CONNECTED) and self.roles->includes("chairman") and p.isChairman() and p.oclInState(GRANTED)</i>

Tableau 4-7 Règles de coordination de base exprimées en OCL

Le Tableau 4-8 illustre l'ensemble des contraintes de coordination spécifiées en OCL issus des 11 règles de coordination des sessions PDR présentées en 3.8.

Règles de coordination PDR	
<p>Un participant peut se connecter à une session PDR :</p> <ul style="list-style-type: none"> ➤ Si la session n'est pas clôturée (Équation 3.42) ➤ et <ul style="list-style-type: none"> ○ si le chairman est connecté (Équation 3.32) ○ ou ○ s'il s'agit d'un participant reviewer et le secrétaire est connecté (Équation 3.33) 	<p><i>Context pdrSms :joinSession(Participant p) : Boolean</i> <i>Pre :</i> <i>not (pdrSession.ocllnState(CLOSED) and (p.type=PARTICIPANT)and(pdrSMS.ConnectedParticipants->includes(select(p : Participant p.type = CHAIRMAN))))</i> <i>or (p.type=REVIEWER)and(pdrSMS.ConnectedParticipants->includes(select(p : Participant p.type = SECRETARY))))</i></p>
<p>Une application peut être arrêtée :</p> <ul style="list-style-type: none"> ➤ s'il s'agit du proxy et ➤ si tous les clients peuvent être arrêtés (Équation 3.37) <p>ou</p> <ul style="list-style-type: none"> ➤ s'il s'agit d'un serveur et ➤ si c'est le seul qui reste en exécution et ➤ si le proxy peut être arrêté (Équation 3.36) 	<p><i>Context</i> <i>CoVerSms : :stopTool(ToolsControl aTool) : Boolean</i> <i>Pre : (</i> <i>(aTool.type=AST_PROXY) and self.stopTool (self.processList->select(t : ToolsControl t.type = AST_CLIENT)))</i> <i>or (</i> <i>(aTool.type=AST_SERVER) and self.processList->one(t : ToolsControl t.type = AST_SERVER) and self.stopTool (self.processList->select(t : ToolsControl t.type = AST_PROXY)) and)</i></p>
<p>Une session PDR peut être ouverte :</p> <ul style="list-style-type: none"> ➤ si le chairman est connecté(Équation 3.34) et ➤ le secrétaire est connecté(Équation 3.34) ➤ et ➤ au moins un critique est connecté (Équation 3.34) ➤ et ➤ si toutes les applications peuvent être démarrées (Équation 3.39) 	<p><i>Context pdrSms : :openSession(): Boolean</i> <i>Pre :</i> <i>(pdrSMS.ConnectedParticipants->one(p : Participant p.type = CHAIRMAN)) and (pdrSMS.ConnectedParticipants->one(p : Participant p.type = SECRETARY)) and (pdrSMS.ConnectedParticipants->one(p : Participant p.type = REVIEWER)) and SMS.SessionDefinition.sessionTools->forall(t :ToolsControl SMS.startTool(t))</i></p>

<p>Une session PDR peut être clôturée :</p> <ul style="list-style-type: none"> ➤ si toutes les applications peuvent être arrêtées. (Équation 3.40) 	<p><i>Context pdrSms : :closeSession(): Boolean</i> <i>Post : SMS.processList->forall(t :ToolsControl SMS.stopTool(t))</i></p>
<p>Une application peut être démarrée :</p> <ul style="list-style-type: none"> ➤ si la session PDR n'est pas clôturée (Équation 3.41) ➤ s'il s'agit du proxy et ➤ si au moins un serveur a été démarré (Équation 3.35) <p>ou</p> <ul style="list-style-type: none"> ➤ s'il s'agit d'un client et ➤ le proxy a été démarré (Équation 3.38) 	<p><i>Context</i> <i>CoVerSms : :startTool(ToolsControl aTool) : Boolean</i> <i>Pre : not (self.oclInState(CLOSED))</i> <i>Post : (aTool.type=AST_PROXY) and self.processList->exists(t : ToolsControl t.type = AST_SERVER)</i> <i>or</i> <i>(aTool.type=AST_CLIENT) and self.processList->exists(t : ToolsControl t.type = AST_PROXY) - - rule 7</i></p>

Tableau 4-8 Règles de coordination du scénario PDR

4.4 Architecture des services de gestion des sessions coopératives multi-applications RMS et SMS

Nous avons réalisé la mise en œuvre des services de gestion de sessions coopératives décrits précédemment. Nous avons fait la réalisation des classes RMS et SMS en langage Java. Nous avons employé un ensemble de technologies Java afin de proposer des services flexibles, ouverts et distribués. Avant de présenter la description globale de l'architecture proposée, nous décrivons sommairement les technologies employées.

4.4.1 CORBA

CORBA² est la réponse de l'OMG³ aux besoins d'interopérabilité d'un nombre grandissant de matériels et logiciels disponibles aujourd'hui [LBS1998]. CORBA permet aux applications de communiquer les unes avec les autres où qu'elles soient et quel que soit leurs concepteurs. L'ORB est le middleware qui établit la relation client-serveur entre les objets. En utilisant un ORB, une application cliente peut de façon transparente invoquer une méthode sur un serveur d'objets, qui peut être sur la même machine ou sur une autre machine à travers un réseau. L'ORB intercepte l'appel et il est responsable de trouver un objet pour implémenter la demande, de passer les paramètres, d'invoquer la méthode, et de retourner le résultat. L'application cliente n'a pas besoin de savoir où se trouve l'objet invoqué, dans quel langage il a été programmé, et sur quel système d'exploitation il est hébergé, ni aucun autre aspect systèmes qui ne fait pas partie d'une interface Objet. De fait, l'ORB procure l'interopérabilité entre des applications hébergées sur des machines différentes dans

² Common Object Request Broker Architecture

³ Object Management Group

un environnement distribué hétérogène, et interconnecte de façon transparente de multiples systèmes objets. CORBA fournit un ensemble de services (CORBAservices) sous forme d'objets CORBA. Ces services sont spécifiés grâce au langage OMG-IDL, ils fournissent les fonctions systèmes nécessaires à la plupart des applications réparties. Actuellement, l'OMG a défini des services pour les annuaires (Nommage et Vendeur), le cycle de vie des objets, les relations entre objets, les événements, les transactions, la sécurité, la persistance, etc. Au fur et à mesure des besoins, l'OMG ajoute de nouveaux services communs.

Dans le cadre de notre travail, nous avons utilisé spécialement les services d'événements et de notification. Le service Événements (Event Service) permet aux objets de produire des événements asynchrones à destination d'objets consommateurs à travers des canaux d'événements. Les canaux fournissent deux modes de fonctionnement. Dans le mode « push », le producteur a l'initiative de la production, le consommateur est notifié des événements. Dans le mode « pull », le consommateur demande explicitement les événements, le producteur est alors sollicité.

Le service Notification (Notification Service) est une extension du service précédent. Les consommateurs sont uniquement notifiés des événements les intéressant. Pour cela, ils posent des filtres sur le canal réduisant ainsi le trafic réseau engendré par la propagation des événements inintéressants.

4.4.2 Java Mail

L'API Java Mail [Sun1998] est un ensemble de classes pour ajouter des fonctionnalités de messagerie électronique aux applications. L'API permet la création, l'envoi et la réception de messages. L'API supporte plusieurs implémentations de messagerie : SMTP, POP3 et IMAP4. Elle supporte aussi plusieurs types de codages : texte, MIME et RFC822.

4.4.3 JDBC

JDBC⁴ est une librairie de classes et d'interfaces Java pour l'accès aux bases de données [Ree1997]. C'est l'équivalent d'ODBC (Open Database Connectivity) dans l'architecture Microsoft. Cette API est intégrée dans le noyau Java depuis la version 1.1. Elle a été développée de façon à permettre à un programme de se connecter à n'importe quelle base de données en utilisant la même syntaxe, elle est donc indépendante du SGBD. De plus, JDBC bénéficie des avantages de Java, entre autres la portabilité du code, ce qui lui vaut, en plus d'être indépendante de la base de données, d'être indépendante de la plate-forme sur laquelle elle s'exécute.

L'accès aux bases de données offert par cette API permet de se concentrer sur la traduction des données relationnelles en objets et non sur l'acquisition des données. JDBC est basée sur SQL ANSI. Elle permet d'utiliser des instructions SQL comme paramètres de méthodes Java.

4.4.4 JSDT

L'environnement *Java Shared Data Toolkit* [Bur1999] est un ensemble de services créés pour faciliter le développement d'applications distribuées ayant de

⁴ Java DataBase Connectivity

grands besoins d'interactivité. JSDT fournit une abstraction basique de session (c'est à dire un groupe d'objets associés par une conception commune de leurs moyens de communication). JSDT permet d'établir une communication full-duplex multipoint entre un nombre arbitraire de composants connectés, ceci sur différents types de réseaux. De plus, cet outil propose un support efficace pour l'envoi de messages multicast, avec la capacité de garantir la livraison de messages uniformément ordonnés, ainsi qu'un mécanisme synchronisé de circulation de jeton.

JSDT est basé sur les protocoles TCP, UDP, HTTP et LRMP. Sa structure générale est constituée de trois parties principales : session, canal et client.

Une session est un ensemble de clients rattachés pouvant échanger des données par différents canaux de communication. L'état de la session inclut la collection de clients et leurs canaux de communication. Une application ou un applet JSDT peuvent avoir plusieurs objets associés avec les mêmes (ou différents) objets dans une session.

Un canal est un cas spécifique d'une voie potentielle multiparti entre deux ou plusieurs clients dans une session donnée. Tous les clients connectés à un canal recevront les données envoyées sur ce canal. N'importe quel client possédant une référence sur un canal est capable d'y envoyer des données. Un client peut avoir des références sur des canaux différents.

Un client est un objet ou un composant qui fait partie d'une application ou d'un applet JSDT. Des clients connectés à une même session peuvent transmettre des données point-à-point ou de façon multipoint.

4.4.5 Java Web Start

Java Web Start [Sch2001] est une solution de déploiement pour des applications basées sur la technologie Java. Il s'agit d'un mécanisme basé sur l'Internet qui permet à l'utilisateur de lancer et contrôler des applications comme si elles étaient des applications basées sur HTTP. Java Web Start fournit l'activation facile des applications, et garantit que la version exécutée est toujours la dernière version de l'application en éliminant des procédures compliquées d'installation ou de mise à jour. Java Web Start permet de trouver, télécharger des applications à partir de n'importe quel serveur Web et de les exécuter sur n'importe quel browser.

Java Web Start a les avantages suivants:

- Une interface utilisateur très interactive, comparable à celles des applications traditionnelles, telles que des traitements de textes et des tableurs.
- Faible consommation de la bande passante. Une application ne doit pas nécessairement se relier de nouveau au serveur web sur chaque clic de l'utilisateur, elle utilise l'information déjà téléchargée (tampon de mémoire). Ainsi, l'application peut fournir une meilleure interactivité sur des connexions à faible débit.
- Les applications peuvent travailler hors ligne.

4.4.6 Remote Method Invocation (RMI)

RMI est la solution d'objets distribués fournie par SUN Microsystem[CWH1998]. Le package RMI comprend des objets et des méthodes qui fournissent les mécanismes assurant la communication entre le client et le serveur et le

passage d'informations de l'un à l'autre. Les applications structurées ainsi, sont parfois appelées « applications d'objets répartis ».

De la même manière que CORBA utilise l'IDL et IIOP pour faire communiquer des objets écrits en différents langages, RMI emploie JRMP (Java Remote Method Protocol) pour faire communiquer des objets Java. Il utilise la sérialisation pour transporter ces objets, permettant ainsi le passage par valeur (ce que ne fait pas CORBA aujourd'hui). Sun propose, depuis la version 1.3 de Java, une passerelle vers CORBA au travers du support du protocole de transport IIOP.

RMI est architecturé en trois couches :

- Le stub (côté client) et le skeleton (côté serveur) assurent la conversion des communications avec l'objet distant.
- La couche de référence (Remote Reference Layer) est chargée de fournir aux objets, un moyen d'obtenir la référence à l'objet distant. Elle est assurée par le package `java.rmi.Naming`, appelé registre RMI car elle référence les objets.
- La couche de transport permet d'écouter les appels entrants ainsi que d'établir les connexions et le transport des données sur le réseau par l'intermédiaire du protocole TCP/IP.

Les applications distribuées programmées en RMI bénéficient des avantages et des services proposés par Java (robustesse, portabilité, sécurité, simple à apprendre, gestion mémoire automatique).

4.4.7 XML

XML⁵ (Langage de balisage extensible) n'est pas un langage de programmation mais un métalangage exploitable pour créer un langage[W3C2000]. XML est issu de SGML (Standard Generalized Markup Language) et a été standardisé en 1998 par la spécification W3C XML 1.0. XML est destiné à l'échange de documents, c'est une solution pour la modélisation des contenus et la standardisation de modèles de contenus. Le but de XML est de faciliter le traitement automatisé de documents et de données. L'idée est de pouvoir structurer les informations de telle manière qu'elles puissent être à la fois lues par des personnes sur le web et traitées par des applications qui exploiteront de manière automatisée les informations en question.

Le HTML et le XML sont différents en de très nombreux points dont certains ont trait à l'essence même du langage.

- Le XML décrit, structure, échange des données tandis que le Html ne fait qu'afficher des données.
- Le XML est extensible et permet de créer ses propres balises en fonction des données traitées. En Html, les balises sont prédéfinies et donc figées.

4.4.8 Architecture élaborée

La Figure 4-9 illustre l'architecture de réalisation des services développés par cette thèse: SMS et RMS. La combinaison des différentes technologies décrites ci-dessus, nous a permis de surmonter les limites de chacune de ces technologies . Par exemple, JDBC ne permet que l'accès de base de données locales, alors que nous avons le besoin d'interroger une base de données à travers des réseaux LAN et WAN. Nous avons créé une application client - serveur dédié à cette tâche. Nous avons placé

⁵ eXtensible Markup Language

le serveur dans le même site que la base de données. Ainsi, l'interrogation via JDBC de la BD se fait par la communication entre le serveur et la base de données locale, tandis que la connexion entre le client et le serveur utilise RMI et donc passe par des réseaux LAN et WAN et propage les requêtes et leurs résultats.

La centralisation des serveurs responsables des services RMS et SMS sur le même site a répondu plutôt à des critères contractuels dans le projet DSE que des contraintes techniques.

Comme il est cité ci-dessus, le service RMS est implanté sous une approche client - serveur. La communication entre les clients et le serveur est réalisée par JAVA RMI. Le client RMS possède une interface graphique, celle-ci fait des appels locaux aux méthodes du composant « Responsibility Managemnt Client » qui à son tour les transforme en appels JAVA RMI. De son côté, le serveur reçoit les requêtes des clients et accède localement à une base de données via JDBC. Nous avons développé notre système sur une base de données relationnelle MS- Access.

Le service SMS aussi est implanté selon le modèle client - serveur. La fonctionnalité côté serveur de SMS est implantée par trois familles d'objets JAVA. Il y a un objet unique du type SMSF qui a en charge la gestion des deux autres familles : SMS et RCS. Chaque client SMS est associé à un couple d'objets SMS-RCS. L'objet SMS a pour objectif la coordination des actions effectuées par les participants et les applications. L'objet RCS prend a charge le gestion des réponses après l'invitation des participants.

En plus des requêtes des clients SMS vers le serveur via l'invocation des méthodes via JAVA RMI, le service SMS a besoin de transmettre des informations vers les clients. Ceci parce que si une action l'état global de la session, alors le nouvel état doit être communiqué aux clients. Ce type de communication permet aux participants d'effectuer des actions de manière simultanée et concurrente. Ce besoin de communication a été satisfait par l'utilisation des événements. Dans le cadre du projet DSE, un service de notification basé sur les services d'événements et de notification CORBA a été développé pour ce but.

Le système awareness de l'environnement DSE a pour principal objectif d'intégrer les différents services et systèmes qui le composent. Le système suit le modèle sous l'approche publier/souscrire. Le format des événements dans l'environnement a été défini en utilisant le langage XML. Ceci a permis une définition des événements standardisé.

Nous avons défini l'ensemble des événements relatifs à l'utilisation des nos services de gestion de session, que nous présentons ci-dessous dans la section 4.6. Le serveur SMS joue un rôle principal car il est l'unique producteur des événements du domaine SESSION dans l'environnement DSE.

Nos avons utilisé le service awareness DSE pour la communication des événements *globaux* relatifs à la gestion des sessions. Par contre, les événements *internes*, c'est à dire les événements qui ne concernent que le fonctionnement du service SMS, ont été réalisés par l'utilisation de JSPT. Les fonctionnalités de gestion de présence offertes par le service SMS entraînent des besoins de communication synchrone internes au service SMS. Les clients SMS ont besoin de communiquer entre eux via le serveur SMS. Par exemple, le changement de la position d'un avatar dans la salle de réunion virtuelle provoque un événement qui doit être communiqué à tous les clients de la session et aussi à l'objet SMS, car la position des avatars fait partie de

l'état courante de la session. Pour répondre à ces besoins, nous avons créé trois canaux JSDT par session. Un canal pour la gestion des avatars, un autre pour l'envoi des messages instantanés et un dernier canal qui fait circuler un jeton pour la gestion du rôle de chairman.

La gestion des réponses des participants invités à une session est une activité asynchrone qui s'effectue avant et pendant le déroulement d'une session gérée par les objets SMS. Le chairman d'une session invite les participants par l'envoi d'un email. Ce email véhicule l'URL de l'application de réponse d'invitation (RCS) qui peut être téléchargée en utilisant le protocole de JNLP Web Start avant d'être employée pour répondre à l'invitation. L'application client du RCS montre dans son interface graphique le statut du groupe en termes de réponses d'acceptation, de refus déjà effectués par les autres participants.

La combinaison d'une architecture d'exécution orientée événements et l'implémentation des règles de coordination dans le service SMS, nous a permis la réalisation d'une architecture dite hybride [Tro1997]. Une telle architecture possède un objet central qui garde l'état global de chaque session (objets SMS). A l'arrivée d'un nouveau participant, cet état (objet SessionState) est envoyé au participant. L'évolution de l'état de la session dans chaque site participant est réalisée par la réception des événements et l'application des règles de coordination déclenchées.

Grâce à la réalisation de l'architecture orientée événements nous avons obtenu trois avantages, d'abord la facilité dans l'intégration des services développés dans l'environnement DSE, ensuite, la possibilité de gérer la connexion des participants retardataires et enfin l'ouverture des services. Dans ce dernier point, nous pouvons citer l'exemple du modèle de la gestion du contexte des sessions MSDC [RVDM2002] qui a été facilement intégré. Ce modèle consomme les événements produits par notre service SMS et construit un diagramme de coordination basé sur un graphe qui représente les dépendances producteur - consommateur entre participants, applications et données.

Le Tableau 4-9 résume la liste des versions des technologies employées dans la réalisation de l'architecture d'exécution des services RMS et SMS.

Technologie	Version
CORBA	2.1
Java 2 Standard Edition qui inclut : JDK, JDBC, JFC Swing, Java RMI.	1.3
Java Mail	1.1
JSDT	2.0
Java Web Start	1.0
	.1
XML	1.0

Tableau 4-9 Liste des versions des technologies employées

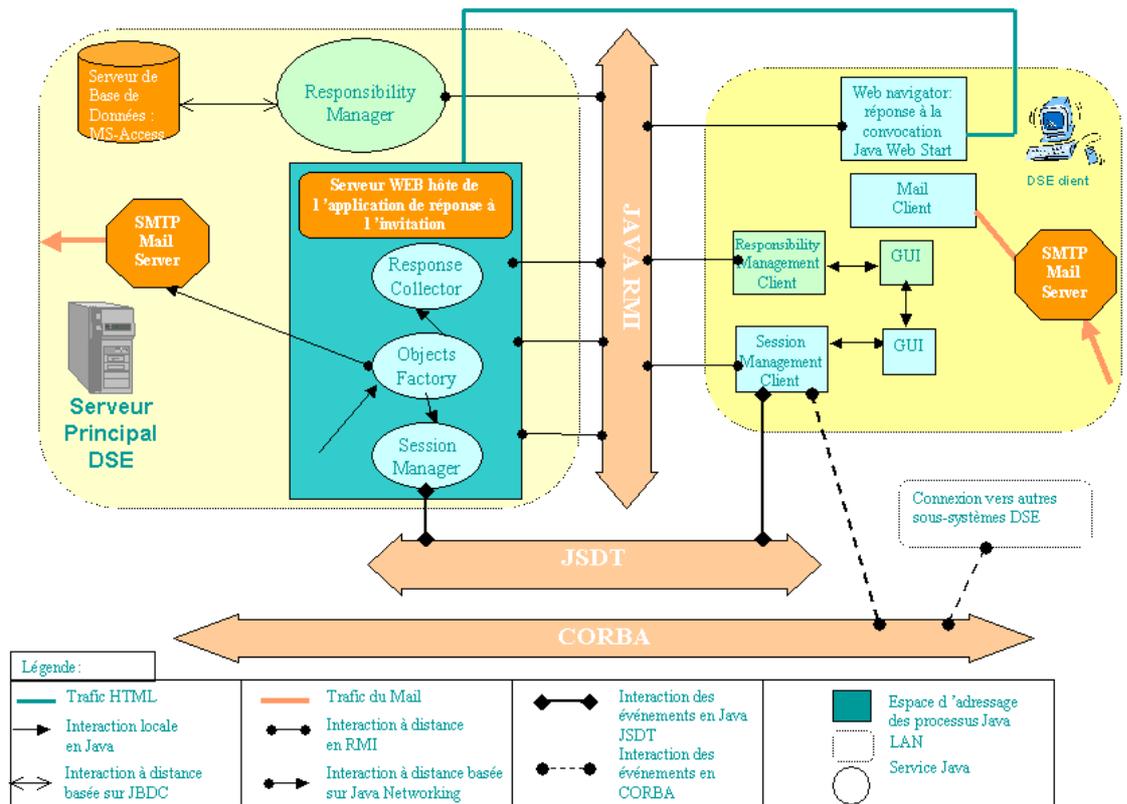


Figure 4-9 Architecture des services RMS et SMS

4.5 Systèmes orientés événements

Etant donné que le terme « événement » est largement répandu dans le domaine de l'informatique, il s'avère nécessaire de considérer les caractéristiques des événements dans le contexte de la gestion de sessions coopératives ainsi que dans les applications coopératives distribuées. Dans la suite, nous allons présenter les généralités des systèmes distribués orientés événements et ensuite nous développerons la définition, et les caractéristiques d'un modèle de gestion de sessions de coopération orienté événements.

4.5.1 Systèmes distribués orientés événements

Un cadre de distribution basé sur les événements nous donne les avantages suivants:

- Les fournisseurs et les consommateurs peuvent être mis en marche indépendamment.
- Plusieurs fournisseurs et consommateurs peuvent être impliqués dans la même interaction.

- Les consommateurs sont reliés aux serveurs au moyen d'une politique publier/souscrire, ainsi par exemple les consommateurs n'ont pas besoin de savoir à l'avance l'adresse IP du fournisseur et réciproquement.
- La communication peut être lancée autant de la part des consommateurs que des fournisseurs.

4.5.2 Définition d'un événement

Un événement est défini comme une paire d'éléments : un *déclenchement* – c'est à dire la cause, et une *action* – c'est à dire la réponse. Le déclenchement est l'identification d'un certain ensemble prédéfini de circonstances liées à l'opération du système qui fait qu'une action particulière est entreprise. L'action est la réaction prédéfinie du système qui suit l'état de déclenchement.

4.5.3 Types d'architectures des systèmes distribués orientés événements

L'architecture générique d'un système distribué orienté événements suite une politique de type publier/souscrire qui est définie par trois genres de composants:

- les fournisseurs qui sont les composants qui produisent des événements.
- les consommateurs qui sont les composants qui utilisent les événements.
- le canal des événements qui est responsable de contrôler les événements et qui est le moyen de communication entre les fournisseurs et les consommateurs.

Les composants peuvent agir simultanément comme des fournisseurs et des consommateurs. Les fournisseurs génèrent les données sur le canal des événements afin que celles-ci soient consommées par les consommateurs. La manière de l'interaction entre les fournisseurs et les consommateurs dépend du modèle de coopération utilisé. On distingue 4 modèles de coopération :

- Le modèle « annonceur ». Dans ce cas, les fournisseurs envoient les données (événements) sur le canal d'événement qui va jouer le rôle d'annonceur, c'est à dire, le canal annonce aux consommateurs qu'il y a des données disponibles et les leur livre. Ce modèle attribue l'initiative dans un premier temps aux fournisseurs puis au canal.
- Le modèle « courtier ». Dans ce modèle, c'est au canal d'événements d'interroger les fournisseurs en leur demandant de lui livrer un ensemble d'événements. Ensuite, les consommateurs à leur tour demandent au canal de leur livrer cet ensemble.
- Le modèle « file ». Ce modèle permet aux fournisseurs d'initier le transfert de données vers le canal, puis les consommateurs demandent ces données au canal.
- Le modèle « agent ». Dans ce cas, le canal des événements demande aux fournisseurs les événements disponibles, puis il prend l'initiative du transfert de données aux consommateurs. Ce modèle n'attribue l'initiative qu'au canal.

Tous ces modèles ont un comportement différent du schéma client/serveur des systèmes distribués. Selon ce dernier, le client demande un service au serveur et celui répond à sa demande. Le modèle publier/souscrire se résume par le fait que le fournisseur (les serveurs ou les clients) génère des données et les rend disponibles à la consommation et c'est à ce moment que le consommateur (les serveurs ou les clients) peut les utiliser.

4.6 Service de gestion des sessions coopératives orienté événements

En utilisant la définition de session coopérative donnée en 3.2.2 et suivant une approche « annonceur », nous avons défini quatre types d'événements de session : les événements concernant l'état de la session, les événements concernant les actions des participants et les événements concernant les actions des outils et les événements d'information.

4.6.1 Définition des événements de l'état de la session

Ce type d'événements est associé à la manipulation de l'état de la session. Ce genre d'événements est défini par la DTD représentée par la Figure 4-10, où un événement de session se compose par l'identificateur de la session (**Session_ID**) et le type de l'événement (**evName**). L'attribut **NotifDateTime** fait une référence à une entité DTD générique externe qui définit la date et l'heure où l'événement a été notifié. La structure de conférence, qui est également définie dans une DTD générique externe, contient les données de la conférence associée (identification, nom, date et heure de démarrage).

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!-- DSE State Related Event definition -->
<!ELEMENT Session_Event (Conference*)>
<!ATTLIST Session_Event
    Session_ID CDATA #REQUIRED
    %NotifDateTime;
    evName (OPEN|CLOSE|CREATE|DELETE|INVITE) #REQUIRED>
```

Figure 4-10 DTD pour la définition des événements relatifs à la session.

Les événements définis sont les suivants:

1. **CREATE**. Cet événement est déclenché lors de l'initialisation d'une nouvelle session de coopération. Dans le cadre du projet DSE, cet événement est consommé par le système *Domain Specific Applications* pour créer un référentiel de documents qui contiendra les documents employés pendant la session. En outre, le système de gestion du GroupWare réagit à cet événement en envoyant des commandes de configuration par l'intermédiaire de telnet au serveur de

- vidéoconférence afin d'installer les paramètres de conférence (débit, QoS, nombre maximal des participants, message de bienvenu, message de rejet, date et temps de début, durée, etc.).
2. DELETE. Cet événement est déclenché lors de la terminaison d'une session de coopération.
 3. OPEN. Cet événement est déclenché au moment où la session de coopération est ouverte. Cet événement marque le début de la phase du travail collaboratif. Dans DSE, cet événement est consommé par le système de gestion du GroupWare lequel envoie une requête au serveur de vidéoconférence afin de permettre la connexion à la conférence associée.
 4. CLOSE. Cet événement se produit quand la session de coopération est clôturée. A la réception de cet événement, le système de gestion du GroupWare du DSE envoie une requête au serveur de vidéoconférence afin de désactiver la conférence associée à la session.
 5. INVITE. Cet événement est utilisé pour inviter les participants à une session de coopération.

4.6.2 Définition des événements des participants.

Ce type d'événements concerne les activités des participants. La Figure 4-11 montre la DTD qui définit ce type d'événements. Les événements des participants sont constitués de trois attributs: l'identifiant de la session (**Session ID**), la date et l'heure de son envoi (**NotifDateTime**) et le type d'événement (**evName**). L'élément User est décrit dans une DTD générique externe qui contient le nom et le prénom du participant qui a produit l'événement. Cette DTD externe contient en plus le login, le mot de passe, l'adresse IP et l'e-mail du participant ainsi que l'organisation à laquelle il appartient et le rôle qu'il joue dans la session.

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!-- DSE Participant Related Event definition -->
<!ELEMENT Participant_Event (User, Conference*)>
<!ATTLIST Participant_Event
    Session_ID    CDATA #REQUIRED
    %NotifDateTime;
    evName (JOIN|LEAVE) #REQUIRED>
```

Figure 4-11 DTD pour la description des événements des participants

Les événements produits par les participants sont :

1. JOIN. Les participants déclenchent cet événement quand ils rejoignent la session de collaboration. Le SMS situé dans l'emplacement de chaque participant est concerné par ce genre d'événement car, celui-ci est employé pour mettre à jour la liste de participants en ligne dans chaque site.
2. LEAVE. Cet événement est produit quand un participant abandonne la session.

4.6.3 Définition des événements d'information

Ce type d'événements a été créé dans le cadre de DSE afin de fournir un moyen de communication via le service de notification awareness. Les événements d'information se traduisent par des messages instantanés entre les participants. Ces messages sont formatés par la DTD de la Figure 4-12, où un message d'information est composé de trois attributs : le participant qui envoie le message (**Originator**), l'identificateur de la session (**Session_ID**) et le type d'événement (**evName**).

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!-- DSE Instant Message Event declaration -->
<!ELEMENT Info_Ses_Event (Originator)>
<!ATTLIST Info_Ses_Event
  Session_ID CDATA #IMPLIED
  evName (MESSAGE | WARNING) #REQUIRED>
```

Figure 4-12 DTD pour des événements de transmission de messages

Ce genre d'événements est défini par les déclenchements et les actions suivantes:

1. MESSAGE. Ce type d'événement est déclenché par les participants pour envoyer un message de type texte à d'autres participants. Ce genre d'événements est montré dans la console consommateur du service de notification.
2. WARNING. Ce type d'événements est envoyé aux participants pour avertir de possibles erreurs ou dysfonctionnement pendant une session, e.g. des problèmes pendant le démarrage des applications, des problèmes liés au réseau.

4.6.4 Définition des événements des applications

Ce type d'événements est lancé afin de démarrer ou arrêter une application en particulier. Les événements des applications sont définis en suivant la DTD décrite par la Figure 4-13. Ce type d'événements est composé par trois éléments : une ou plusieurs conférences (**Conference**), le participant qui détient l'outil (**Originator**), et un ou plusieurs participants consommateurs (**User+**). Un événement d'outil contient aussi l'identificateur de session (**Session_ID**), l'heure et la date de notification (**NotifDateTime**), l'identificateur de l'outil (**Tool_id**), l'identificateur du flux de données associé à l'outil (**Flow_id**) et le type d'événement (**evName**).

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!-- DSE Tool Mangement Event declaration -->
<!ELEMENT Tool_Event (Originator, User+, Conference+)>
<!ATTLIST Tool_Event
  Session_ID CDATA #REQUIRED
  %NotifDateTime ;
  Tool_id CDATA #REQUIRED
  Flow_id CDATA #REQUIRED
  evName (START | STOP) #REQUIRED>
```

Figure 4-13 DTD pour la définition des événements des outils

Les actions associées aux événements des outils sont :

1. START. Cet événement commande le démarrage d'une application particulière.
2. STOP. Cet événement commande l'arrêt d'une application.

4.7 Conclusion

Dans ce chapitre, nous avons présenté la conception et la réalisation des services RMS et SMS pour la configuration et la gestion des sessions coopératives multi-applications. La conception et la mise en œuvre de ces services a été effectuée dans le cadre des études menés durant le projet DSE. D'un total de 227 exigences qui ont composé le cahier de charges du projet DSE concernant la fonctionnalité de l'environnement global, nos services RMS et SMS ont répondu à 49 de ces exigences. Nous ne nous sommes pas limités à répondre à ces exigences, nous avons étendu nos travaux afin de proposer des modèles génériques et des architectures ouvertes qui seraient utilisables dans d'autres travaux de développement des systèmes coopératifs distribués.

Le modèle de structuration des groupes de coopération ouverts présenté dans la section 4.2.1 a fait l'objet d'une publication [MDV2002]. Le modèle UML qui représente la conception du service de gestion de sessions SMS décrit en 4.3 a été publié en [MND2001]. L'architecture orienté événements du service SMS qui contient les définitions des événements relatifs à la session, aux participants, à l'information et aux applications, qui a été présentée en 4.6, a aussi fait l'objet d'une publication [MDR2002].

Chapitre 5. Scénario de validation : application du gestionnaire des sessions dans un cas de l'ingénierie coopérative distribuée

5.1 Introduction

Ce chapitre détaille le processus de validation du service de configuration des groupes de collaboration appelé RMS et du service de gestion de sessions appelé SMS dans le cadre du projet européen DSE. Nous avons expérimenté nos services présentés dans le chapitre précédent dans deux cas d'utilisation réels. Nos services ont été utilisés dans les deux scénarios de validation du projet mais nous nous focaliserons et ne présentons que le plus représentatif vis-à-vis des travaux de l'ingénierie coopérative distribuée : la revue de conception préliminaire (PDR) du véhicule de transfert automatique (ATV).

Ce chapitre est organisé de la façon suivante : d'abord nous introduisons le projet DSE. Ensuite, nous montrons l'exécution de la revue préliminaire de conception du véhicule de transfert automatique en utilisant l'environnement DSE. Nous mettons spécialement l'accent sur les rôles exécutés par nos services développés RMS et SMS. Enfin, nous présentons une analyse des bénéfices obtenus dans la revue du programme ATV en utilisant la plate-forme DSE.

5.2 Le projet DSE

DSE (Distributed Systems Engineering) est un projet de recherche Européen cofinancé par la Commission Européenne IST-1999-10302. Il a été réalisé par un consortium européen comprenant des sociétés industrielles ayant une forte implication dans des activités du domaine spatial. Elles sont intervenues en tant qu'utilisateurs de

l'environnement réalisé et qui port le même nom que le projet, DSE. Le consortium a été aussi composé par des sociétés fournissant des compétences technologiques et des centres de recherche. Ces deniers ont contribué à la définition et au développement de l'environnement. Le projet a duré de janvier 2000 à janvier 2002.

Le consortium a été composé par :

Sociétés industrielles	Location
ALENIA AEROSPAZIO Divisione Spazio - Un'Azienda FINMECCANICA S.p.A.,	Turin, Italie
AEROSPATIALE MATRA Lanceurs Stratégiques & Spatiaux (EADS-LV)	Les Mureaux, France
INDUSTRIEANLAGEN-BETRIEBSGESELLSCHAFT GmbH (IABG)	Berlin, Allemagne
Centres de recherche	
LAAS-CNRS	Toulouse, France
Université P&M Curie - Laboratoire (LIP6)	Paris, France
Sociétés technologiques	
SILOGIC	Toulouse, France
SOCIETA ITALIANA AVIONICA S.p.A. (SIA)	Turin, Italie.
D3 GROUP Softwareentwicklung, Forschung	Berlin, Allemagne

5.2.1 Objectifs du projet

Les principaux objectifs du projet sont concentrés sur l'amélioration du processus d'ingénierie coopérative en permettant à différentes équipes éloignées géographiquement d'agir réciproquement à partir de leur propre emplacement, ayant simultanément accès et contrôle sur des applications à distance, des répertoires de données ou de fichiers partagés. Le projet DSE a été prévu pour définir et construire un environnement modulaire capable de supporter des applications utilisées sur une plateforme d'ingénierie coopérative, à grande échelle, dans l'industrie européenne. Ce système doit supporter efficacement les activités d'ingénierie effectuées pendant toute la durée de son cycle de vie. Il prend en compte essentiellement les aspects de conception et de vérification des activités dans le cadre de projets internationaux, impliquant un réseau complexe avec plusieurs niveaux de fournisseurs et de consommateurs.

Les principaux objectifs de ce projet ont été basés sur l'amélioration des processus d'ingénierie coopérative permettant :

- L'analyse des tâches et la conception du système, en exigeant la distribution et l'évaluation coopérative des résultats de simulations et leur analyse par les utilisateurs, les donneurs d'ordre (commission Européenne), les responsables et les gestionnaires du projet ainsi que les partenaires qui sont chargés du développement.
- La révision de la conception, impliquant différents partenaires, afin de consolider et d'évaluer les systèmes conçus par chaque équipe.

- La distribution des processus de vérification. Ceci doit permettre le suivi des tests des phases préparatoires jusqu'aux résultats des essais. Durant cette étape les principaux sites où le système a été conçu et intégré, sont connectés afin de permettre à des équipes spécialistes d'effectuer des tests spécifiques.

Ce projet a permis de construire un environnement basé sur des processus d'ingénierie distribuée, qui est capable d'être utilisé sur des réseaux LAN ou WAN. Il a été développé selon une architecture multicouche avec une infrastructure générique et ouverte prévue pour les communications et le travail de groupe (ou groupware), et une couche spécifique aux utilisateurs, permettant de mettre en commun des données et des applications provenant d'outils hétérogènes.

L'environnement développé a été validé deux fois au milieu et à la fin du projet DSE à l'aide de scénarios réels issus du domaine spatial. Pendant cette phase, les avantages apportés par le projet aux différents profils d'utilisateurs ont été mesurés et les processus de « re-engineering » renforcés pour profiter pleinement des paradigmes de collaboration. Deux types de scénarios ont été utilisés comme référence pour construire et valider l'environnement DSE : « Collaborative Design and Analysis » (Co-Des) et « Collaborative Distributed Verification » Co-Ver. Ils font tous les deux références à l'activité spatiale.

Dans le Co-Des, la revue préliminaire de conception du véhicule de transfert automatique (PDR-ATV) a été prise comme référence et utilisée pour présenter la vérification de la conception par des clients, des entrepreneurs et des critiques. Dans le Co-Ver, l'utilisation de plates-formes et d'applications distribuées a été expérimentée pour la surveillance et le contrôle d'un système spatial réel testé à partir de plusieurs sites éloignés.

L'utilisation de l'environnement DSE dans le cadre d'une activité réelle spatiale ou non, doit fournir des gains considérables en termes de :

- Diminution du cycle de développement et des coûts, dès la première participation, des différentes équipes d'intervenants.
- Bénéfice tiré de la synergie/ des opinions des divers groupes et des différents spécialistes
- Anticipation des issues du développement et détection des limites liées aux techniques employées pour la mise en œuvre des solutions, dès les phases de conception, afin de réduire les risques.
- Utilisation du prototypage rapide comme support de conception de l'avant-projet et du système
- Partage de la conception et de la simulation des données, en assurant simultanément la gestion des droits de propriété de chaque entreprise.

L'architecture de l'environnement DSE est composée d'un maximum de produits (solutions commerciales ou développées par les partenaires) appropriés et réutilisables. Ces derniers s'appuient sur des standards internationaux (CORBA, JAVA, la norme H323 pour la visioconférence, etc.) afin d'assurer l'interopérabilité entre les outils, les échanges des données et des simulations distribuées.

5.2.2 Le système : architecture et composants

5.2.2.1 L'architecture générale

Le point fort de l'environnement DSE est d'intégrer différentes technologies émergentes telles que Java/RMI, CORBA, HLA ou XML, pour réaliser des opérations sur des données distribuées, effectuer des simulations distribuées et définir et exécuter des sessions multipoints sur des plates-formes (UNIX, Windows NT...) et des réseaux hétérogènes. Les technologies utilisées pour construire le projet, ont été choisies de telle sorte que les différents outils soient intégrables et faciles à configurer, que le déploiement de l'application soit modulaire et que les interfaces utilisateurs soient communes.

L'environnement DSE fournit aux utilisateurs un ensemble de services intégrés et étendus tels que la visioconférence « multicast », le partage d'application, la définition et la gestion des sessions, des opérations sur le réseau, et des outils pour mettre des documents à la disposition des autres utilisateurs.

5.2.2.2 Les composants du projet

L'environnement DSE est basé sur une structure modulaire permettant de l'améliorer et de le maintenir pendant son cycle de vie, mais surtout de déployer et de gérer facilement chacun de ses composants. Nous avons adopté une approche orientée objet pour l'élaboration du projet. Quatre composants principaux ont été identifiés, ils sont illustrés par la Figure 5-1 : Middleware, Process Session and Groupware, Domain Specific Application et Network Layer. Ils sont eux-mêmes découpés en sous-systèmes effectuant des opérations spécifiques.

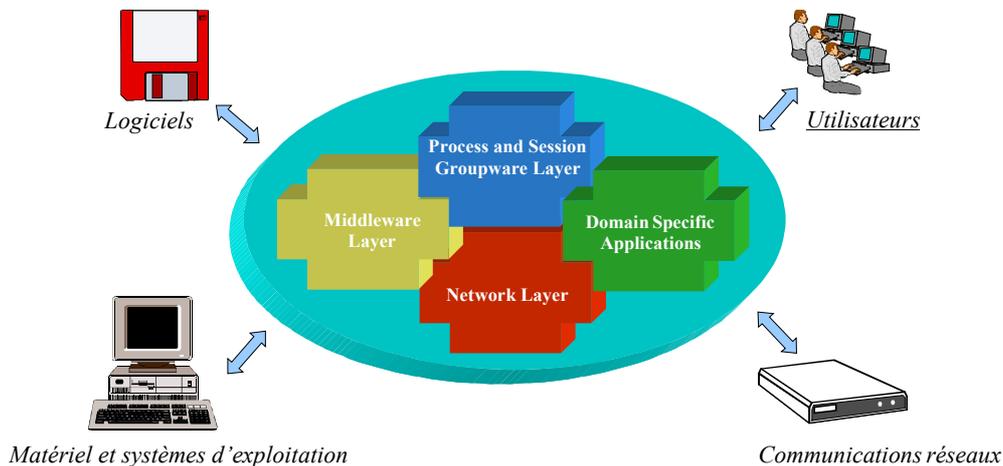


Figure 5-1 Composants de l'environnement DSE

5.2.2.2.1 Middleware

C'est l'ensemble des couches logicielles qui se trouvent entre l'application et le réseau. La tâche principale du Middleware est de supporter le partage des données et des applications du système. Il s'occupe de la distribution des services et gère l'awareness, c'est à dire qu'il notifie les événements produits par les différents outils pendant les sessions synchrones. Le service d'awareness a été conçu pour utiliser CORBA, HLA et XML. La distribution du système et l'interopérabilité des outils sont fondées sur le standard CORBA, son protocole de communication (IIOP) et ses services (notification, gestion des événements).

Le Middleware comprend également les systèmes de simulation et leurs technologies standards (HLA/RTI). Les objets CORBA utilisés dans DSE ont été développés par plusieurs partenaires. Un graphe de nommage unique a été défini pour assurer l'homogénéité et l'interopérabilité des références CORBA. Il a été créé en utilisant Orbix 3.0.1. Le service de nommage CORBA permet de générer une structure de nommage du contexte dans laquelle les noms d'objets et leurs références sont stockés dans un canal de persistance.

5.2.2.2 Process, Session and Groupware

La couche « Process, Session et Groupware » permet de définir les sessions et leurs attributs, de gérer la session, d'effectuer des interactions homme-machine, de contrôler les outils et de gérer la base de données de documents qui constitue la base de référence de la revue. Les principaux composants de cette couche ont été développés dans le cadre de cette thèse. Quatre composants ont été développés pour réaliser ces opérations :

- Responsibility & Session management : ce composant est divisé en deux parties : «Responsibility & User Management Service RMS» et « Session Management Service SMS». La conception et la réalisation de ces deux services ont été amplement décrites dans le Chapitre 4. Le premier module fournit les processus et les interfaces graphiques qui permettent au RMS de définir les participants, les profils des participants (rôles, compétences, dates d'indisponibilité...), les groupes de collaboration, les sessions, les outils, etc. Le service « Session Management Service » permet de gérer l'exécution de la session synchrone. Le participant qui joue le rôle de *chairman* de la session est responsable de cette dernière et peut l'initialiser, l'annoncer, l'ouvrir, la clore et la terminer. Les personnes inscrites dans une session peuvent, après avoir été invitées par le chairman (réception d'un e-mail), accepter ou non de participer à la session et éventuellement se joindre à cette dernière. Les principaux états de la session (initialisation, ouverture, clôture et terminaison) et les actions effectuées par les participants (se joindre ou quitter la session) sont affichés en temps réel au niveau des écrans de gestion des sessions. Ces services sont développées en Java/RMI. Elles fournissent les GUI qui permettent de définir et d'exécuter les sessions collaboratives, mais aussi de préparer le paramétrage des autres applications de travail coopératif (« GroupWare ») qui vont être utilisées.
- Tool Control Management : ce composant permet aux participants de la session d'utiliser des applications de travail coopératif fournis par le système. Le contrôle des applications fournit différents dispositifs, basés sur l'intégration de produits disponibles dans le commerce, destinés à effectuer des communications

synchrones, au moyen d'outils de visioconférence (audio et vidéo), du chat, du tableau blanc, d'utilitaire de partage d'applications et d'outils de travail collaboratif. La visioconférence, s'appuyant sur la norme H323, utilise Meeting Point comme serveur MCU (Multipoint Control Unit). Les outils de dialogue textuel et graphique sont basés sur le standard T120 et sont fournis par Microsoft NetMeeting. L'application VNC, disponible en Open Source, a été choisie afin d'effectuer le partage d'application. Elle a été adaptée pour que le chairman puisse gérer la prise de contrôle de l'application partagée et puisse changer de serveur en cours de session.

- Engineering Data Repository (EDR) : il fournit, aux utilisateurs de DSE, un support sécurisé pour le stockage et le partage d'information tel que des documents, des fichiers, des images, du son et de la vidéo. Les services disponibles dépendent du profil des utilisateurs et de leurs droits d'accès. Ils sont tous référencés au niveau d'un portail. Un serveur Web (Apache) a été mis en place pour permettre aux utilisateurs d'accéder au système de gestion de la base de données via un browser Web. Le système de gestion de base de données est basé sur Oracle (8i Release2) et le portail sur Oracle Portal (ORACLE Portal 3.0)

5.2.2.2.3 Domain Specific Application

La couche Domain Specific Application intègre toutes les applications identifiées qui sont requises pour réaliser les activités d'ingénierie spécifiques au domaine.

Le composant de contrôle des outils autorise une utilisation concurrente des applications externes et, selon le scénario de référence de DSE, il inclut, par exemple, des outils pour traiter la revue PDR-ATV tel que l'utilitaire d'analyse EGSE (Electrical Ground Support Equipment) employé pour la surveillance et l'exécution des activités de test. Il doit également permettre le contrôle interactif et la surveillance des sessions de simulations distribuées.

5.2.2.2.4 Network Layer

Une topologie de réseaux hétérogènes a été identifiée et les services standard TCP/IP ont été adoptés, afin de pouvoir déployer facilement l'environnement DSE dans des environnements techniques actuels et diversifiés.

La couche réseau de DSE permet :

- d'effectuer des communications multicast entre différentes infrastructures réseaux
- de faire communiquer des réseaux privés comme l'ATM (Asynchronous Transfer Mode) avec l'Internet public et réciproquement, via un processus de transformation des adresses privées en adresses publiques et de routage de celles-ci.

5.3 Scénario Co-Des : PDR-ATV

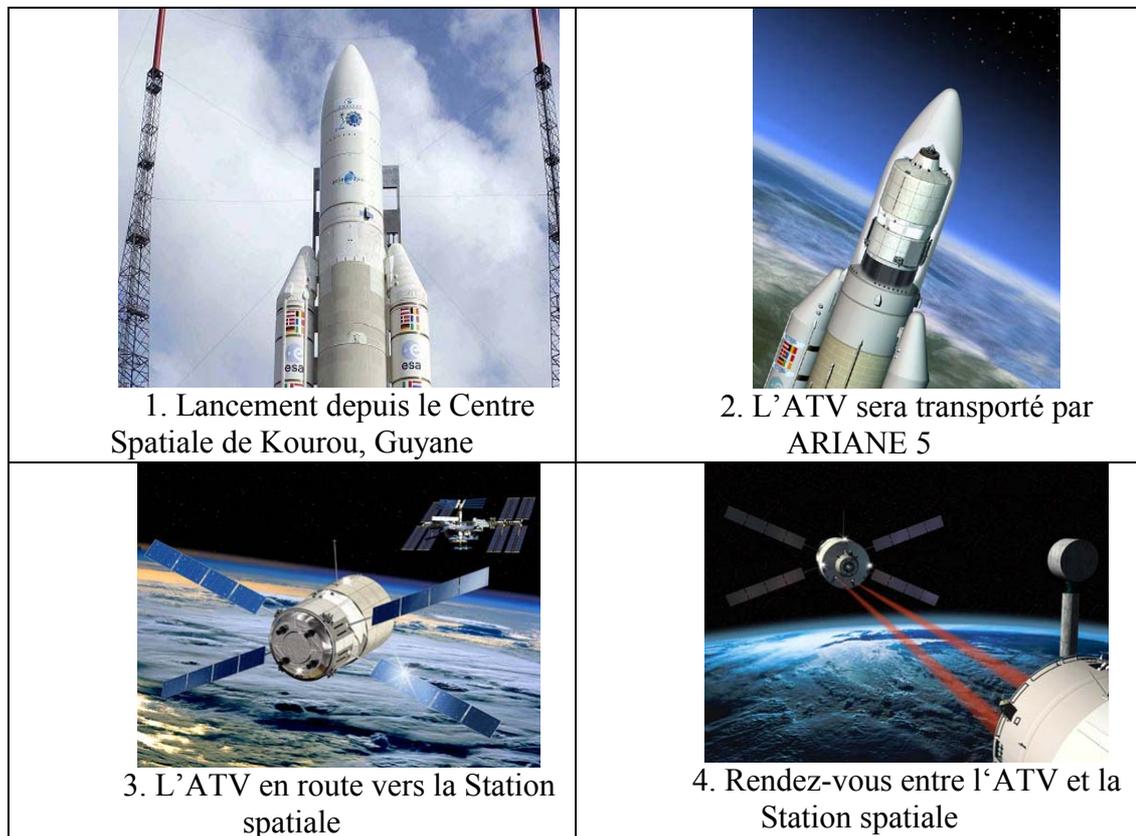
La validation de l'environnement DSE a été réalisée dans un scénario de conception réel. Le scénario de validation choisi est l'application de la revue PDR au

programme de Segment de Vol du véhicule de transfert automatique (ATV). Le véhicule de transfert automatique de l'Agence Spatiale Européenne (ESA) ravitaillera la Station spatiale en ergols, nourriture et autres fournitures, lors de chacune de ses missions.

Le caractère polyvalent de ce véhicule représente une contribution essentielle de l'Europe à l'exploitation régulière de la Station spatiale. L'ESA fournira neuf ATV qui seront lancés par Ariane 5 à douze mois d'intervalle. Le premier lancement est prévu pour 2004.

Chaque ATV restera amarré à la Station pendant plusieurs mois. Au cours de cette période, ses moteurs pourront être allumés pour rehausser l'altitude la Station spatiale, compensant la baisse progressive de l'orbite causée par l'atmosphère de la Terre et la pression solaire.

A la fin de sa vie utile, l'ATV sera rempli de déchets et désamarré du complexe orbital. Il se désintégrera en rentrant dans l'atmosphère terrestre. Le tableau 5-1 montre la séquence en images de la mission de l'ATV. Ces images ont été extraits du site Web de l'ESA¹.



¹ <http://www.esa.int>

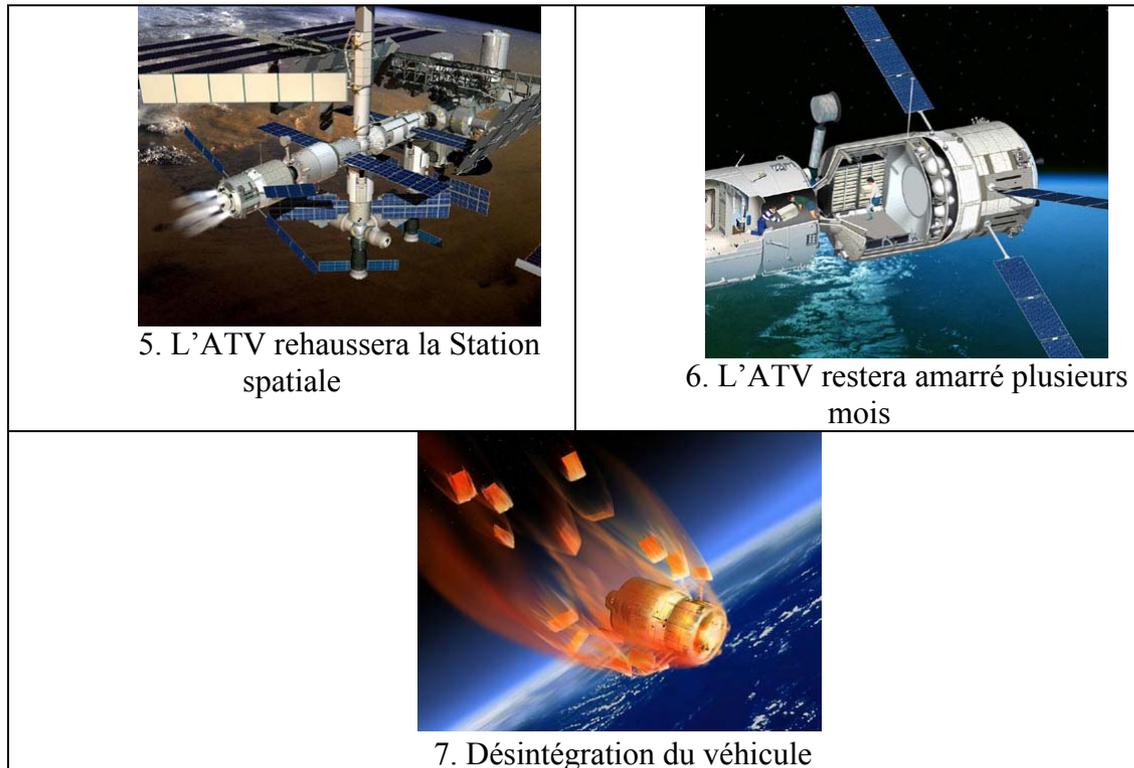


Tableau 5-1 Séquence graphique de la mission du Véhicule de transfert Automatique

La revue PDR-ATV a été tenue pendant l'année 2000. Les principales difficultés, pour organiser ce processus, étaient à traiter le grand nombre de personnes impliquées et le volume des documents. Pour cette revue, il y a eu plus de 120 critiques qui se sont réunies dans la même location pendant 3 semaines. La base de référence a été constituée de plus de 40000 pages. Enfin plus de 2500 RID et de 30 à 50 recommandations ont été établies.

Ces éléments exposent les contraintes quantitatives d'une grande revue telle que PDR-ATV :

- réservation d'une douzaine de salles pendant une longue période (10 personnes par salle),
- besoin de moyens de transport entre les salles,
- approvisionnement de 25 copies en papier de la documentation passée en revue (base de référence de revue) et
- difficultés pour contrôler 2500 RID.

5.3.1 Exécution de la revue PDR-ATV en utilisant l'environnement DSE

L'objectif du scénario PDR-ATV est de démontrer comment la fonctionnalité de l'environnement DSE peut réduire en grande partie les contraintes rencontrées pendant la revue PDR-ATV. L'environnement DSE insiste sur l'amélioration de l'efficacité opérationnelle du processus dans les sujets suivants:

- 1) Distribution automatisée de la base de référence :** DSE est employé pour donner un accès électronique facile et sécurisé aux documents en évitant le grand nombre de copies en papier.

Les outils employés pour la lecture de ces documents électroniques sont configurés par le service RMS et contrôlés depuis le service SMS.

- 2) Analyse coopérative et distribuée des issues techniques par groupes de dix personnes :** DSE est employé pour exécuter des réunions d'analyse par une douzaine d'experts situés à leur bureau.

Les unités de collaboration dans le système sont spécifiées par les sessions de coopérations définies par les services RMS et SMS. Le service RMS est capable de former des groupes et des sessions de coopération réutilisables.

- 3) La gestion et le contrôle de la revue de manière distribuée coopérative:** ceci signifie donner aux personnes impliquées dans la revue une vue claire de ce qu'elles doivent faire et quand, ce que les autres font et ce qui est déjà réalisé. Cet aspect du système comprend aussi le maintien d'une mémoire global (conscience ou *awareness*) pour les participants de l'occurrence d'événements pendant la revue (par exemple: la création d'un nouveau RID).

Le service SMS apporte plusieurs fonctions qui répondent à ce type de besoins. D'abord il y a les fonctions de présence qui fournissent la réunion virtuelle et la messagerie instantanée. En utilisant la salle de réunion virtuelle, les participants peuvent déterminer en temps réel, quels sont les participants présents (connectés), quels sont les participants invités, quels sont les participants qui ont accepté l'invitation et ceux qui l'ont refusée. Le service de messagerie instantanée permet aux utilisateurs d'échanger des messages pendant la session. Cette fonctionnalité se révèle très importante en cas de panne d'autres services de communications, e.g. la vidéoconférence.

Ensuite, le service SMS fournit un outil de messagerie asynchrone véhiculé par e-mail qui permet aux utilisateurs d'envoyer des messages soit aux participants actifs (connectés), soit aux participants invités.

Enfin, le service SMS fournit la capacité de concéder le rôle du chairman de la session d'un participant à un autre.

- 4) La validation dans un environnement multi-sites :** Le scénario PDR-ATV démontrera que DSE peut être employé sur des réseaux hétérogènes LAN et WAN.

Grâce à sa réalisation en Java et à l'utilisation d'une politique d'intégration producteur - consommateur réalisé en Java-RMI et CORBA, les services RMS et SMS peuvent être employés sur des plate-formes hétérogènes.

- 5) Analyse des facteurs humains:** les facteurs humains représentent un aspect primordial dans la performance de l'environnement DSE. Les facteurs humains incluent: les aspects de formation et de support, la rentabilité, la transparence de la technologie aux utilisateurs et la qualité de vie au travail.

La conception et l'implémentation incrementale en deux étapes nous ont permis de tenir compte des facteurs humains via l'évaluation par des utilisateurs finals des services que nous avons développés et mis à leur disposition.

Les participants aux scénarios de validation du projet ont répondu à un ensemble de questionnaires d'évaluation dont deux spécialement dédiés aux services RMS et SMS.

5.3.2 Tâches PDR-ATV avec l'environnement DSE

Le processus de revue PDR consiste en trois tâches, décrites dans la section 3.7.3, effectuées l'une après l'autre. L'environnement DSE permet de soutenir les trois tâches. Les activités relatives à la première tâche, la préparation de la revue, sont soutenues par le système EDR et le service RMS. Le système EDR s'occupe de la formation, de la diffusion et de l'accès à la base de référence. Le service RMS permet de constituer les groupes de la revue, d'effectuer sa programmation et d'affecter les tâches de vérification aux différents groupes.

Les activités effectuées pendant les deux tâches suivantes, l'exécution et le traitement des résultats de la revue, sont soutenues à travers la configuration et l'exécution des sessions de coopération, réalisées à l'aide de l'ensemble de services DSE. Les sessions coopératives constituent les unités basiques de travail dans l'environnement DSE. Chaque session coopérative est configurée par le service RMS et exécutée par le service SMS. Pendant l'exécution de chaque session, elle fait appel aux autres services du système, e.g. pendant la configuration d'une session, le service RMS crée les fichiers de configuration qui sont utilisés par le système *Tool Control Management*, à savoir les fichiers de configuration du sous-système E-VNC (partage d'applications), et la configuration à distance du serveur de conférence multi-point MCU. En raison du fait que les sessions sont la base principale du travail coopératif dans l'environnement DSE et du fait qu'elles regardent de la même façon leur fonctionnement, ce qui change à chaque session est essentiellement le but. On présentera un exemple qui représente la forme d'utilisation de l'environnement DSE pour la réalisation de la revue PDR-ATV. Cet exemple met l'accent sur la configuration et l'exécution d'une session de coopération qui a pour objectif de réunir des critiques du comité de revue, des membres de l'équipe de projet et des membres du comité d'examen autour d'un RID spécifique issu de la revue réelle PDR-ATV effectué en 2000.

La réalisation des trois tâches qui composent le processus PDR peut être décomposée en 5 étapes sous l'utilisation de l'environnement DSE :

1. Configuration et préparation de l'environnement DSE,
2. Préparation de la revue PDR,
3. Les critiques analysent les documents et publient des RID,
4. L'équipe de projet répond aux RID,
5. Exécution d'une session de coopération pour une analyse approfondie sur un RID particulier.

5.3.3 Configuration et préparation de l'environnement DSE

Les activités envisagées dans cette tâche permettent l'installation et la vérification des composants distribués qui conforment l'environnement DSE.

5.3.3.1 Installation des serveurs DSE

La Figure 5-2 détaille les serveurs de l'environnement DSE qui sont repartis en trois pays et cinq lieux d'exploitation. Au LAAS-CNRS nous avons installé deux serveurs : le serveur de configuration et gestion de groupes « Responsibility Management System » et le serveur de gestion de sessions de coopération « Session Management System ». Au LIP6 se situe le serveur de conférence multipoint. Chez

Silogic est installé le serveur de la base de référence de la revue (EDR). Le serveur du service d'awareness est localisé à SIA. A EADS-LV, on trouve le serveur d'applications du type *domain specific applications*, décrite en 5.2.2.2.3. La Figure 5-2 montre aussi les connections nécessaires entre les serveurs.

5.3.3.2 Installation des clients DSE

Cinq clients ont été déployés chez EADS-LV, ELV, LAAS, LIP6 et D3. Chaque client intègre les logiciels suivants, montrés par la Figure 5-3 :

- un web browser comme Internet Explorer 5.x sur Windows NT,
- un client H.323 (Microsoft NetMeeting, CUSeeMe)
- un client RMS et SMS,
- un client du système awareness
- une application de messagerie électronique e-mail
- un client du système E-VNC pour le partage des outils d'ingénierie

5.3.3.3 Distribution d'acteurs

La Figure 2-7 présentée précédemment montre la répartition des cinq acteurs impliqués dans la revue PDR-ATV.

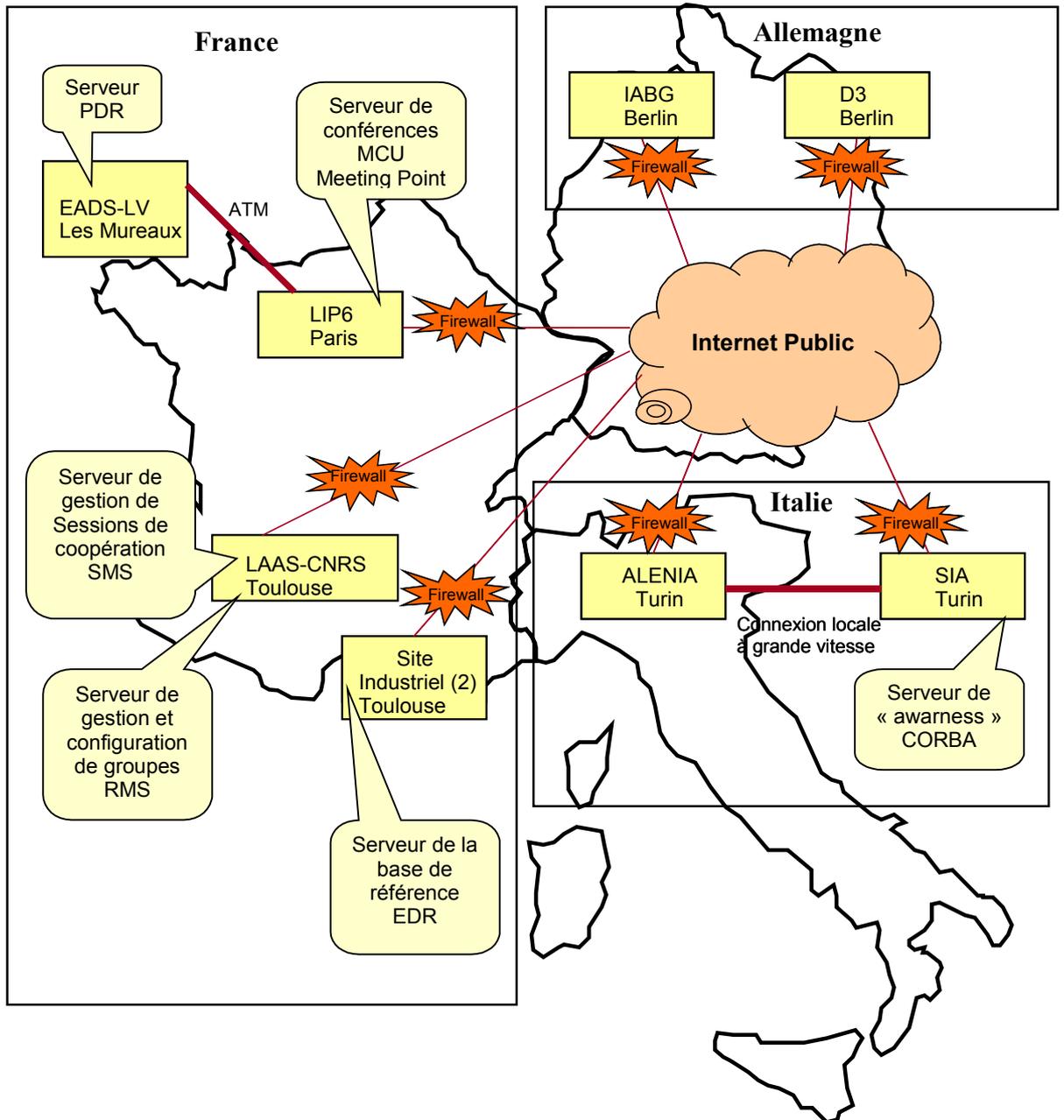


Figure 5-2 Répartition des serveurs

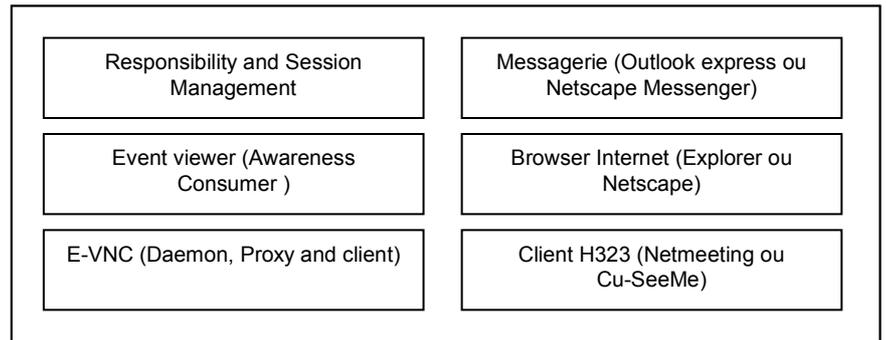


Figure 5-3 Configuration du logiciel dans chaque client DSE

5.3.4 Préparation de la revue préliminaire de conception.

Les tâches de préparation de la revue PDR-ATV sont sous la responsabilité du secrétaire du comité de revue. Ces tâches consistent à former les groupes de revue (les groupes du comité de revue, les groupes d'équipe de projet), à planifier les sessions qui forment les activités de la revue et à construire la base de référence et affecter les tâches de révision à chaque groupe.

5.3.4.1 Formation des groupes de collaboration

Le secrétaire du comité de revue est responsable de l'organisation des groupes qui interviennent pendant la revue PDR. L'objectif principal consiste à intégrer les groupes de la revue à partir des groupes de collaboration définis dans l'environnement DSE. Juste après, on énumère la liste des organisations, des compétences, des rôles, des thèmes, des outils et des tâches nécessaires pour la revue PDR-ATV :

Les organisations dans ce scénario sont: EADS-LV, Alenia Spazio, D3, LAAS-CNRS et LIP6.

Les tâches : tâche 1. Préparation de la revue, tâche 2. Exécution de la revue, tâche 3. Traitement des résultats de la revue.

Les rôles : Président du comité de revue, secrétaire du comité de revue, participant du comité de revue, participant de l'équipe de projet, membre du comité d'examen.

Les qualifications : Comité de Revue, Comité d'examen, Equipe de projet.

Les thèmes : 1. Thermal/Power, 2. Rangées solaires, 3. Trajectoires 4. Conditions d'amarrage, 5. Guidance / Navigation / Contrôle, 6. R-GPSs, 7. Communications, 8. Compas gyroscopiques, 9. Sûreté du matériel et 10. Sûreté de logiciel

En utilisant ces données, le secrétaire peut définir la liste des participants. Chaque participant est caractérisé par ses rôles, ses qualifications, ses thèmes. La Figure 5-4 montre la fenêtre pour la définition des participants à l'aide du service RMS. Dans cette figure, on note la définition des rôles, des qualifications et des thèmes du participant. Ensuite, le secrétaire définit les groupes suivants : 1. Thermal/Power, 2.Guidance/Trayectoires, 3.Contrôle et Command de l'ATV, 4.Sécurité, 5. Traitement de données et 6. Interface Russe. Ces groupes correspondent à la classification pour repartir la révision de documents.

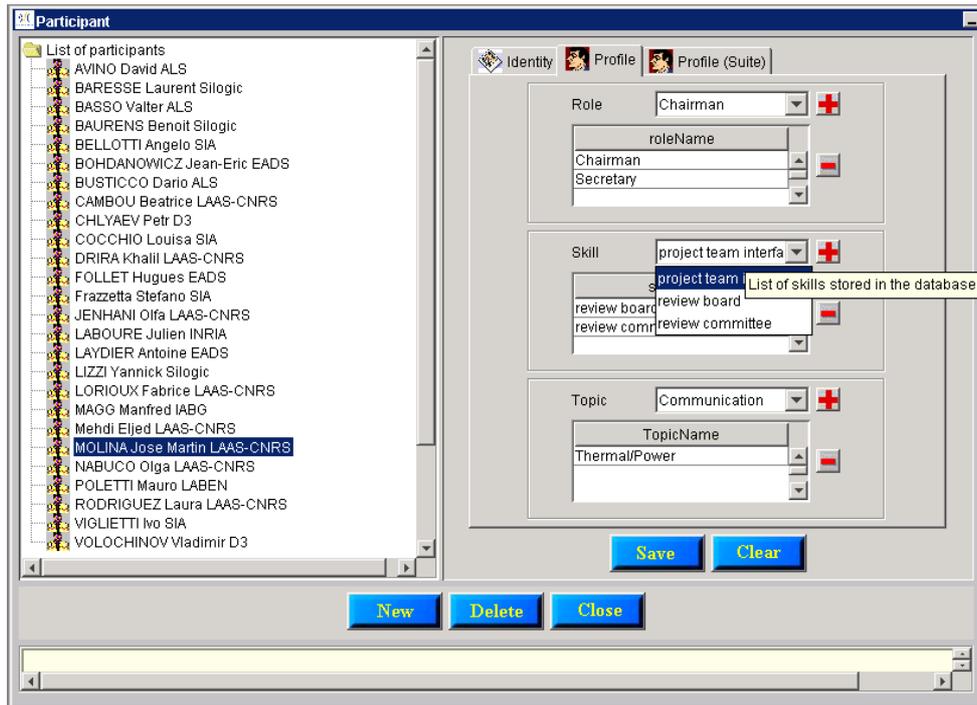


Figure 5-4 Définition de participants dans le service RMS

La Figure 5-5 montre la définition des groupes de collaboration en utilisant le service RMS. Chaque groupe est constitué d'un ensemble de participants, dont un possède le statut de chairman du groupe.

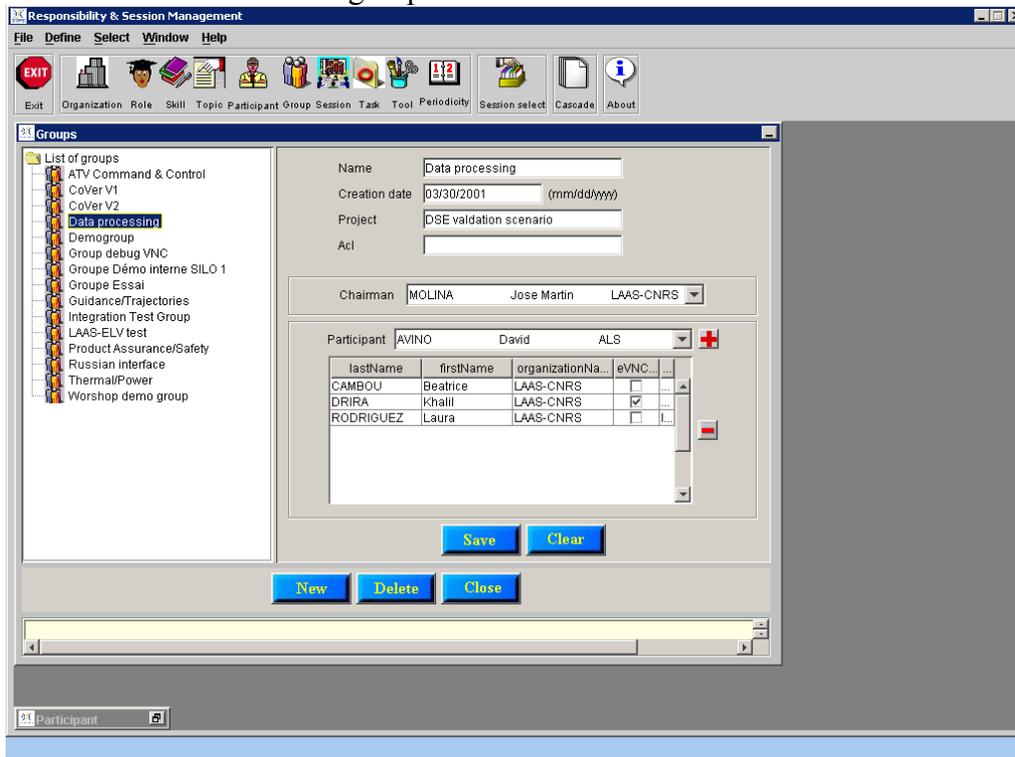


Figure 5-5 Définition des groupes de collaboration à l'aide du service RMS

5.3.4.2 Programmation de la revue PDR-ATV

Le secrétaire du comité de revue en est aussi responsable. Il s'agit de construire le calendrier des sessions qui forment la revue et d'affecter les tâches de révision de documents aux groupes de collaboration formés ci-dessus.

La tâche antérieure a déjà préparé la liste de participants et la formation des groupes de collaboration. A partir de ces groupes, l'environnement DSE peut définir des sessions de coopération. En effet, dans chaque groupe de collaboration, on distingue un ensemble de participants, dont un possède le rôle de responsable de groupe (chairman) et qui par la suite deviendra le chairman de la session. Ainsi, la définition d'une session de coopération se compose par les données générales (nom, but, heure, date, etc.), par le groupe de collaboration et par la liste des outils à employer pendant la session. La Figure 5-6, nous montre la définition d'une session de coopération à l'aide du service RMS. Cette figure nous montre en particulier l'onglet de données générales de la session « PDR preparation ».

Dans la Figure 5-7, on peut distinguer le groupe de collaboration qui a été associé à la session. On peut aussi remarquer que le participant « Martin MOLINA » est le chairman du groupe en conséquence il sera le chairman de cette session. Finalement, la Figure 5-8 nous montre la liste des applications qui seront utilisées au cours de cette session.

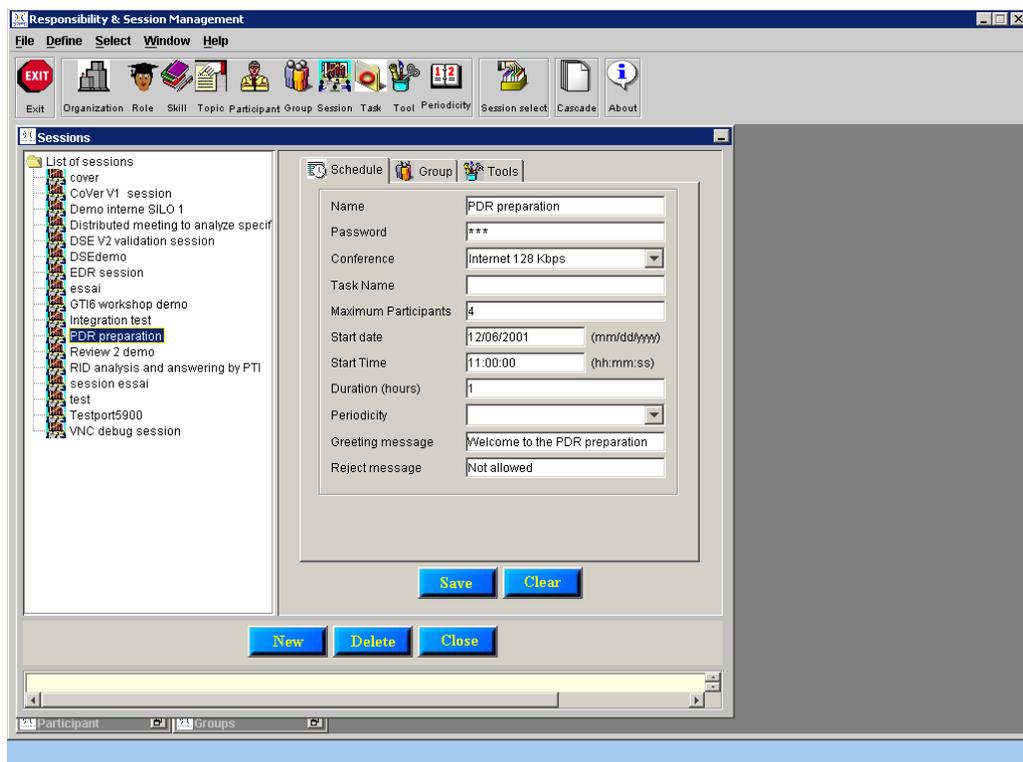


Figure 5-6 Définition d'une session de coopération avec le service RMS

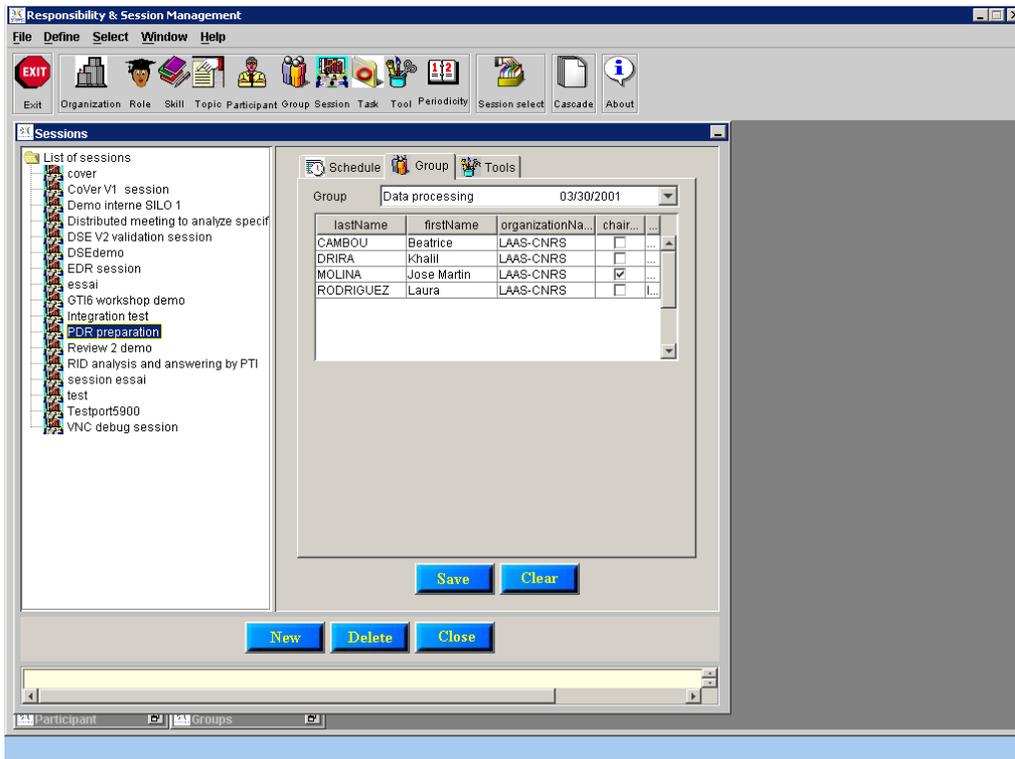


Figure 5-7 Association d'un groupe de collaboration à une session

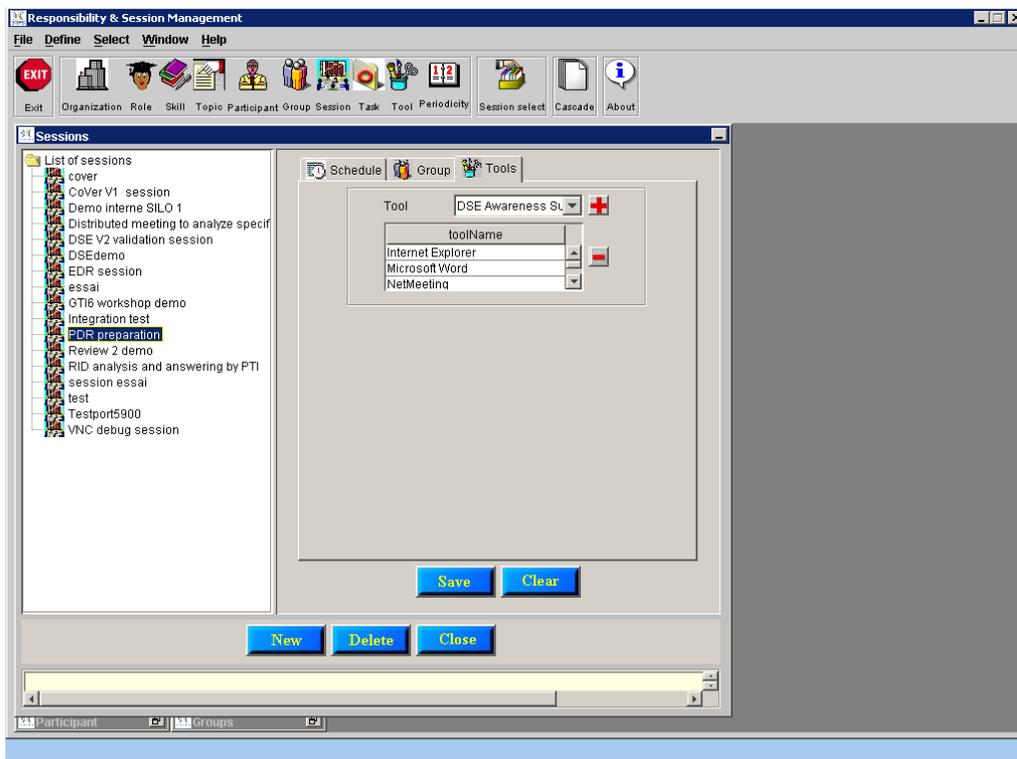


Figure 5-8 Définition des outils à employer durant une session de coopération

5.3.5 Les critiques analysent des documents de la base de référence et publient RID.

Dans cette tâche, les groupes du comité de revue révisent l'ensemble des documents affectés, et ensuite ils génèrent des RID. La révision des documents par les groupes peut se réaliser par zéro, une ou plusieurs sessions de coopération. Dans la première possibilité, on peut supposer que tous les critiques ont analysé les documents et qu'ils n'ont pas trouvé d'anomalies, alors il n'y a pas besoin de réaliser une session coopérative. Par contre, si une ou plusieurs anomalies sont trouvées, alors il faut réaliser une ou plusieurs sessions de coopération entre les critiques afin de discuter et élaborer le(s) RID associé(s). Dans l'exemple que nous sommes en train d'exposer, nous avons supposé que les critiques ont examiné les résultats mais qu'ils n'ont pas eu besoin d'effectuer de sessions de coopération.

5.3.6 L'équipe de projet répond aux RID

Dès qu'un RID a été établi, l'équipe de projet répond au comité de revue sur le RID en question. Dans cette tâche, l'équipe de projet rassemble les informations et les documents nécessaires afin de répondre aux questions posées par le comité de revue. Ce rassemblement peut être réalisé par zéro, une ou plusieurs sessions, selon le degré de l'anomalie. Comme dans le cas précédent, on a supposé qu'il n'y a pas eu de sessions de coopération.

5.3.7 Exécution d'une session pour analyser un RID particulier

Une fois qu'un RID a été établi, et que l'équipe de projet a répondu, il est nécessaire d'organiser une session entre les membres du comité de revue qui ont initié le RID et les membres de l'équipe de projet qui ont répondu. Cette session a pour objectif de fermer le RID ou bien de créer une recommandation. Cette session correspond bien à l'illustration des fonctionnalités que possède l'environnement DSE pour aider les participants à réaliser de manière facile et efficace le travail coopératif distribué. Cette tâche est divisé en:

- La session est initialisé par le président de la session (chairman),
- Le chairman invite les membres du comité de revue, de l'équipe de projet et du comité d'examen (ceux-ci pour valider la réunion),
- Les participants acceptent ou refusent l'invitation de la session,
- Les participants qui ont accepté se joignent à la session,
- Le chairman ouvre la session. A partir de ce moment les applications sont démarrées automatiquement, et les participants peuvent commencer a interagir. Pendant cette phase les participants discutent entre eux. Les critiques du comité de revue posent des questions et les participants de l'équipe de projet répondent et expliquent les sujets du RID.

5.3.7.1 Initialisation de la session

La première étape d'une session de coopération consiste à l'initialiser, c'est à dire, à créer le serveur de la session. Cette activité est réalisée par le chairman à l'aide du service SMS et constitue le premier pas dans le cycle de vide de la session.

Une fois que la session a été initialisée, alors le chairman invite les participants. Cette étape est réalisée de manière automatique et illustrée par la Figure 5-9. Cette

invitation se traduit par l'envoi à chaque participant d'une convocation via la messagerie électronique e-mail qui contient les informations sur la session, illustré par la Figure 5-10. La convocation est formatée selon le standard du Web HTML et contient un lien personnalisé qui permet à chaque invité de répondre en acceptant ou en refusant l'invitation. De cette manière les réponses sont collectées automatiquement, et le chairman de la session est tenu au courant en temps réel des décisions des participants par notification via sa messagerie électronique.

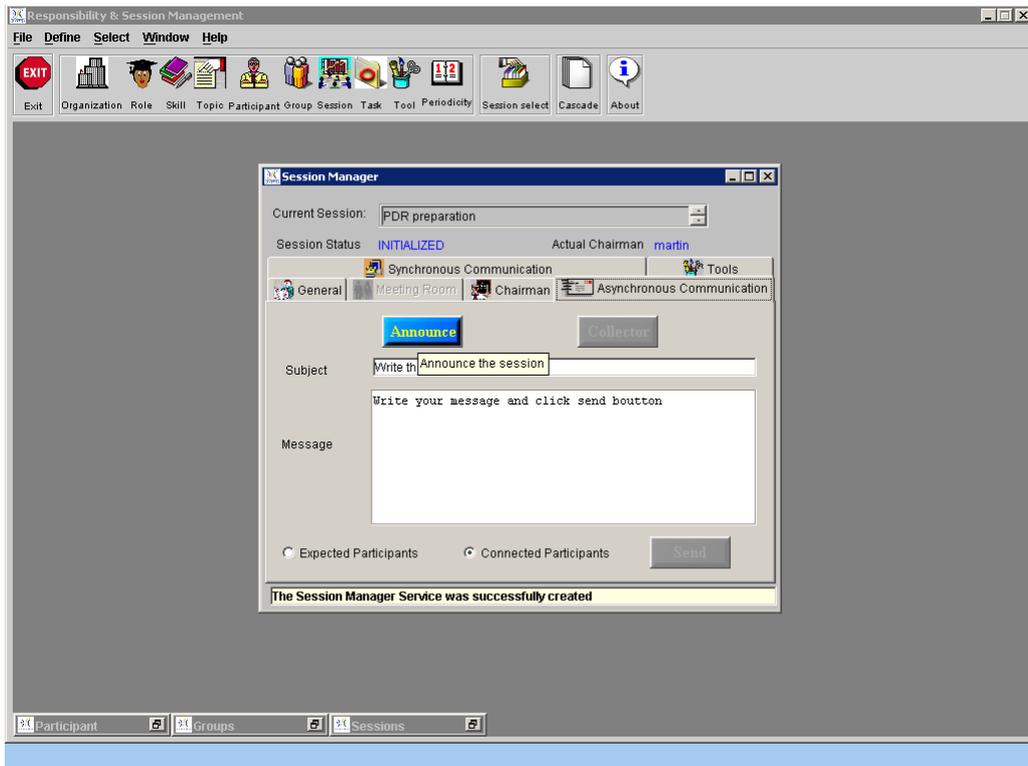


Figure 5-9 Invitation des participants à une sessions en utilisant le service SMS

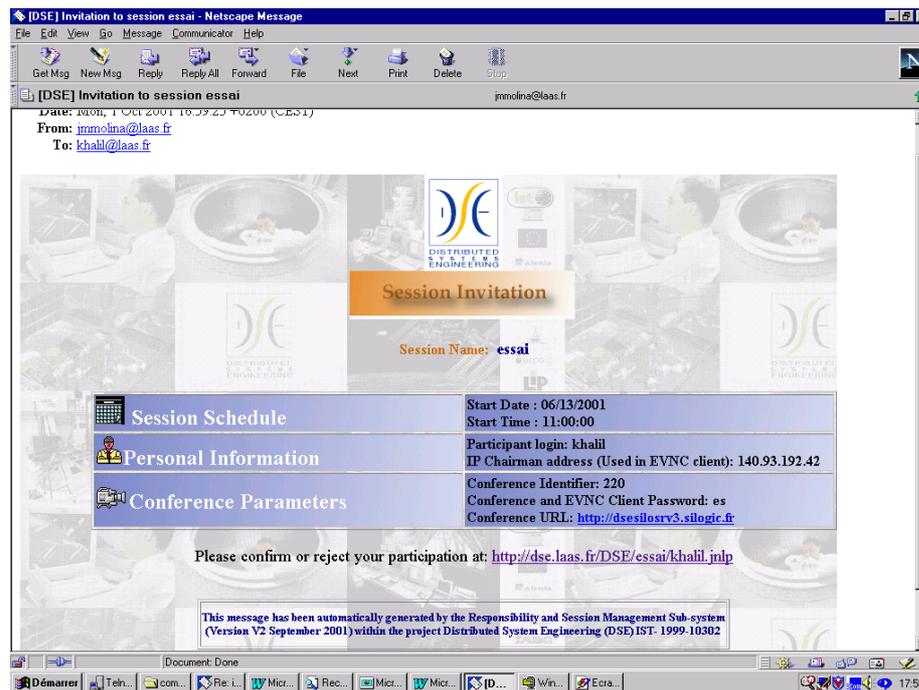


Figure 5-10 Convocation formatée HTML pour l'invitation à une session de coopération

5.3.7.2 Le participant se joint à la session

Si un participant a accepté l'invitation, alors il peut se joindre à la session. Ceci est réalisé à l'aide du service SMS, illustré par la Figure 5-11. Une fois connecté, le participant peut échanger des messages instantanés avec les autres participants, ou bien des messages asynchrones avec les participants reliés ou attendus.

Scénario de validation : application du gestionnaire des sessions dans un cas de l'ingénierie coopérative distribuée

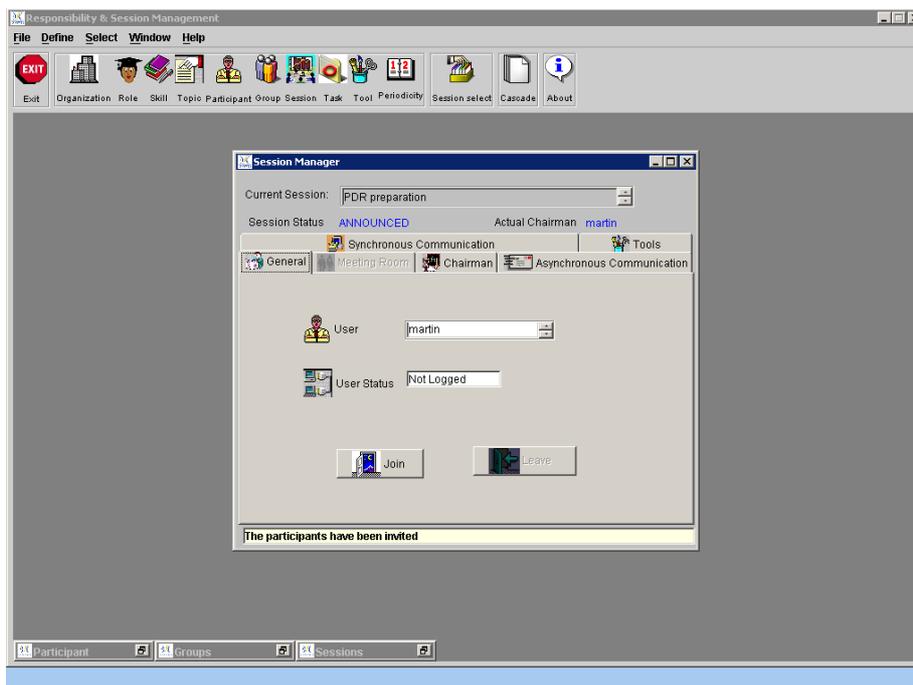


Figure 5-11 Connexion du participant à l'aide du service SMS

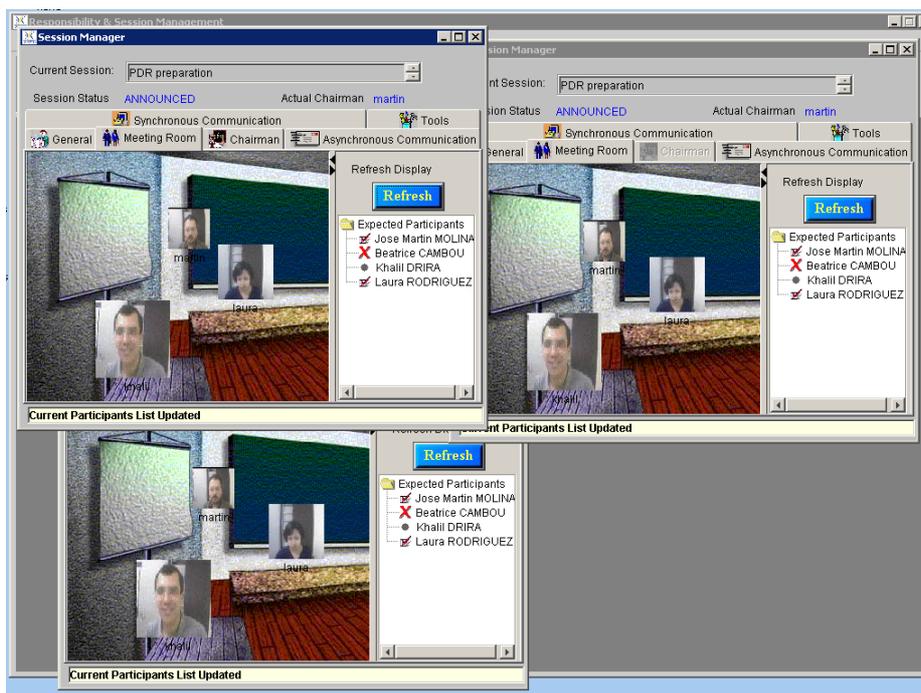


Figure 5-12 Trois SMS correspondants à une session à trois

La représentation de la présence des participants est réalisée par des *avatars* (dans notre cas, on a choisi une photo du participant). La Figure 5-12 illustre les fenêtres correspondantes d'une session à trois participants, où la salle de réunion virtuelle du service SMS est mise en évidence. La salle de réunion est constituée de deux parties. Au centre de la fenêtre se trouvent les avatars des participants connectés. A droite de la fenêtre se trouve la liste de participants invités. Chaque nom de participant possède une icône (à gauche) qui représente la réponse du participant à l'invitation. L'icône « » indique que le participant a accepté l'invitation, « », indique que le participant l'a refusée et « ● » indique que le participant n'a pas encore répondu. Cette information est utilisée par le chairman pour prendre la décision sur l'ouverture ou l'annulation de la session.

5.3.7.3 Les participants analysent RID

Le chairman de la session détermine le moment d'ouvrir la session par le nombre de participants connectés ou bien par leurs rôles. L'ouverture de la session signifie le moment où les participants se mettent à travailler ensemble. Afin de prendre cette décision, le chairman utilise l'information fournie par la fonctionnalité de la salle de réunion virtuelle.

L'ouverture de la session provoque le lancement automatique des applications chez tous les participants.

Le chairman identifie le RID qui sera analysé pendant la session. Chaque participant accède aux documents associés. Le travail coopératif commence au moment où le chairman donne la parole au participant qui a créé le RID, celui-ci explique aux participants les causes du RID. La communication est réalisée, généralement, par une vidéoconférence. Le participant s'appuie sur l'utilisation des diagrammes ou des documents qui sont vus par l'ensemble des participants en utilisant le partage d'applications.

Pendant cette présentation, d'autres participants peuvent poser des questions ou donner des explications en utilisant la vidéoconférence ou bien par la demande du droit de parole sur les outils partagés. Ainsi, les participants de l'équipe de projet peuvent répondre aux questions posées sur le RID.

La Figure 5-13 illustre l'aspect du bureau d'un participant pendant la session. Dans la figure on illustre l'application de vidéoconférence, le service SMS, le service d'awareness et le partage d'applications.

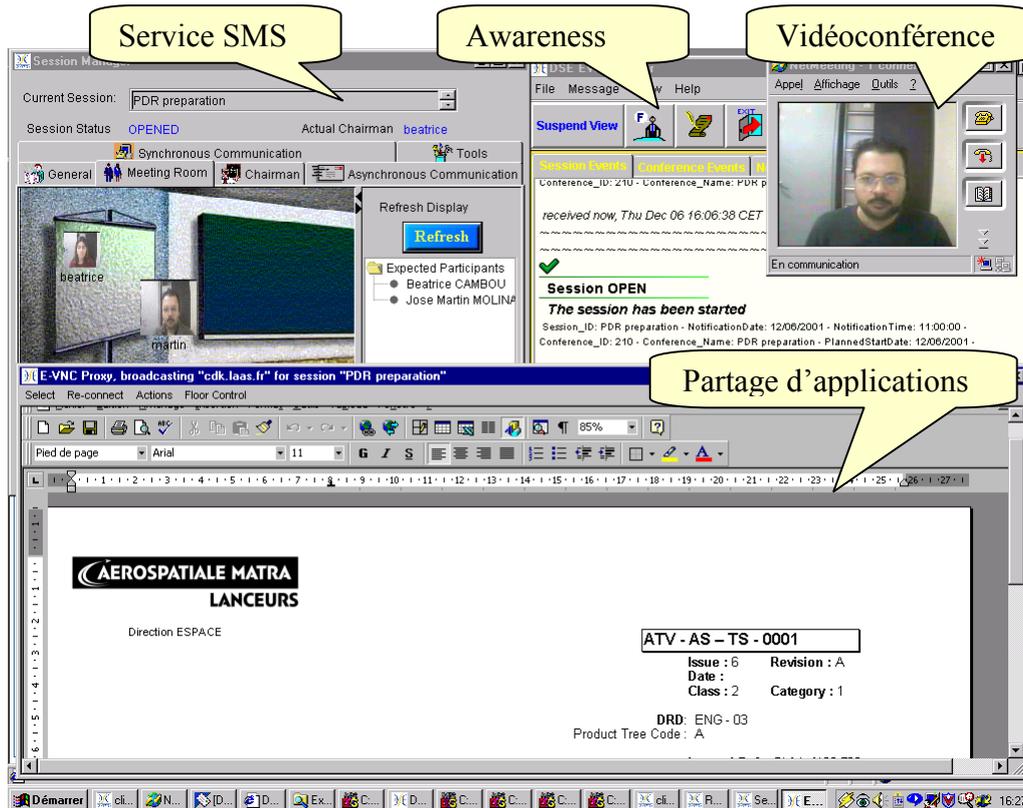


Figure 5-13 Bureau de participant lors d'une session de coopération

5.4 Analyse des bénéfices dans PDR-ATV en utilisant l'environnement DSE

Les résultats des phases de validation sont exprimés par une comparaison de ce qui est une PDR sans l'utilisation du DSE et ce qui est avec l'aide des fonctions DSE. Cette comparaison est faite dans le tableau suivant.

Le tableau est organisé selon les différentes phases du processus PDR-ATV. La dernière phase du processus n'est pas comparée car elle ne fait pas partie des scénarios de validation.

Cette comparaison permet d'évaluer les avantages et les limites dans la mise en œuvre de l'environnement DSE pour soutenir le processus PDR-ATV, spécialement dans les trois premiers objectifs évoqués en 5.3.1 : la distribution automatisée de la base de référence, l'analyse et distribution des questions techniques par groupes de revue et la gestion et le contrôle du processus distribué et coopératif.

Activité	En dehors DSE	Avec DSE
1. Préparation de la revue PDR-ATV		
Le secrétaire et le président du comité de revue s'occupent de choisir des critiques potentiels extérieures au projet, de	Ils emploient le téléphone et le fax.	Idem car l'environnement n'est pas déployé chez les experts potentiels à ce stade du processus.

Scénario de validation : application du gestionnaire des sessions dans un cas de l'ingénierie coopérative distribuée

Activité différents domaines d'expertise et venant de différentes compagnies et consortiums.	En dehors DSE	Avec DSE
Le secrétaire du comité de revue établit le programme de la revue	Il emploie un logiciel de gestion de programme (MS-project ou un logiciel semblable). Cette programmation est incluse dans la note d'organisation PDR.	Idem, mais le secrétaire peut éditer le programme avec le système EDR. Ce qui lui permet de mettre à jour plus fréquemment et facilement les modifications sans attendre une nouvelle version de la note d'organisation PDR.
Ils créent les groupes du comité de revue	Les groupes de revue sont décrits dans la note d'organisation de PDR	En utilisant le système RMS les profils des participants sont créés (nom, organisation, adresse, rôles, qualifications, matières, téléphone, fax, E-mail, etc.). Des groupes de collaboration sont aussi définis. Ces définitions sont réutilisées lors de la définition des sessions. Ceci permet la consultation facile des données toujours actualisées.
Le secrétaire et le président du comité de revue proposent à l'ESA un premier paquet des données industrielles (la base de référence est composée de ce paquet de données industrielles et d'un paquet de documents de conception).	Le paquet des données industrielles est intégré à la note d'organisation PDR. Ce paquet est progressivement raffiné et négocié avec l'ESA par des lettres, des conversations téléphoniques et des réunions face à face. Finalement ce paquet est fixé et mis à jour dans une nouvelle issue de la note d'organisation.	Le paquet de données industrielles est progressivement défini et rendu disponible en utilisant les applications propriétaires PDR et le système EDR de DSE. Les services RMS et SMS sont employés pour organiser de sessions de coopération distribuées entre l'ESA et l'entrepreneur principal.
Des réunions sont organisées entre EADS LV et ESA/ESTEC pour valider la note d'organisation PDR-ATV.	Ces réunions sont organisées dans les locaux d'EADS-LV ou de l'ESA.	Certaines de ces réunions ont lieu en utilisant les services RMS et SMS.

Activité	En dehors DSE	Avec DSE
La note d'organisation PDR-ATV est éditée.	La note d'organisation est publiée (document en papier).	La note d'organisation est éditée et publiée sur EDR (document électronique).
La base de référence est rendue disponible aux critiques.	Plus de 25 copies de papier de 60 documents sont envoyés en Europe, aux Etats-Unis (la NASA) et en Russie.	La base de référence est disponible à travers un portail Internet du système EDR.
Le président et le secrétaire du comité de revue affectent les tâches de revue aux critiques (en termes de documents à passer en revue).	La liste des documents à passer en revue par les groupes du comité de revue est mentionnée dans la note d'organisation.	Les documents sont organisés dans le système EDR, selon les groupes de revue qui doivent les analyser. Evidemment, le même document peut être passé en revue par plusieurs groupes.
Le secrétaire du comité de revue réserve les salles de réunion, les repas et le transport entre les salles afin de préparer les réunions d'analyse de documents.	Les réunions sont réalisées dans les locaux d'EADS-LV aux Mureaux. Pendant 2 semaines, plus de 100 personnes assistent aux réunions. Celles-ci ont lieu dans 12 salles de réunion. EADS-LV loue un bus pour le transport des critiques entre les salles.	Le nombre et la durée des réunions peuvent être sensiblement réduits par l'automatisation avec l'utilisation des services RMS et SMS. La logistique d'organisation du secrétaire de PDR est aussi réduite.
Le secrétaire met à disposition les canevas de la revue (RID, Recommandation).	Les canevas sont disponibles dans la note d'organisation et sur le service web PDR de EADS-LV.	Les canevas sont mis à disposition dans le service EDR.
Le secrétaire prépare le système pour traiter les RID (création, enregistrement, etc.)	Ces services sont supportés par le service web PDR de EADS-LV.	DSE ne compte pas avec un système de traitement des RID. Ceci est partiellement couvert par le Domain Specific Applications.
La réunion pilote est organisée par EADS -LV.	Plus de 100 participants assistent à la réunion à l'ESA.	Il convient de noter que les solutions techniques DSE sont modulables et permettraient à un certain nombre de personnes de participer à ce type de réunions (non examiné dans le cadre du projet).

Activité	En dehors DSE	Avec DSE
		DSE).
2. Exécution de la revue PDR-ATV		
Les critiques(plus de 100 personnes) analysent les documents.	Les critiques appartiennent à 11 consortiums ou compagnies. Chaque critique regarde la note d'organisation pour savoir quels sont les documents qu'il doit analyser. Puis, chaque critique a besoin de localiser parmi les participants de son consortium, une personne possédant une des 25 copies de papier du paquet industriel et se mettre d'accord avec lui pour pouvoir accéder aux copies.	Chaque critique peut accéder aux documents qu'il doit analyser à travers l'application EDR. Les documents sont organisés selon des groupes de revue. Chaque critique peut avoir une vue générale de l'ensemble de documents, ensuite, télécharger et imprimer ceux qui l'intéressent vraiment.
Les critiques publient leurs RID.	Les critiques utilisent le canevas pour produire les RID. Ils ont besoin de les identifier avec un code périodique personnel (par exemple Tom Jones – Tjones001, Tjones002, etc.). Ensuite, chaque critique envoie ses RID par la poste au comité de revue.	Les critiques utilisent le canevas RID localisé au système EDR. Les RID sont stockés dans le répertoire personnel du critique. Ces RID sont accessibles par le comité de revue.
Chaque groupe du comité de revue organise des réunions pour avoir une première vue sur les RID créés. Le but est de grouper les RID semblables. Le groupe peut préparer une pré-recommandation à cette étape. Les RID sont identifiés avec une référence selon le groupe de revue. Un premier rapport de groupe de revue est préparé.	Ces réunions se déroulent dans les locaux d'ESA en Hollande, les critiques proviennent des Etats-Unis, de Russie et d'Europe. Le chairman du groupe de revue utilise le site web de l'ESA pour registrer les RID générés par son groupe.	Certaines de ces réunions peuvent avoir lieu en utilisant les services RMS et SMS (vidéoconférence, Partage d'applications). Idem pour l'enregistrement des RID.
Les groupes de l'équipe de projet accèdent aux RID et répondent selon leurs	Les groupes de l'équipe de projet accèdent au site web ESA pour obtenir les RID	Les groupes de l'équipe de projet accèdent aux RID en utilisant le système EDR.

Activité	En dehors DSE	Avec DSE
responsabilités.	et donner des réponses.	
Des réunions sont organisées pour analyser les réponses des groupes de l'équipe de projet et résoudre les RID.	Ces réunions se déroulent dans les locaux d'EADS aux Mureaux. Plus de 120 participants venant des Etats-Unis, de Russie et d'Europe se réunissent pendant 2 semaines.	Le nombre et la durée de ces réunions peuvent être réduits de manière significative en employant les services de sessions de coopération.
Chaque responsable de groupe du comité de revue écrit et publie le compte rendu des réunions du pas précédent.	Les copies papier du compte rendus sont diffusées aux participants le lendemain.	Les rapports de réunion sont édités par l'EDR.
Après chaque réunion, les RID qui ont été résolus sont mis à jour avec les décisions qui a ont été prises.	Les responsables des groupes du comité de revue accèdent et mettent à jour les RID dans le site web ESA.	Les responsables des groupes du comité de revue accèdent et mettent à jour les RID dans le système EDR. De l'information additionnelle réservée au comité d'examen peut être ajoutée.

Une simulation de cette activité à une échelle plus réduite a été expérimenté pour évaluer les fonctions offertes par les composants DSE. Ces résultats sont présentés dans la section suivante.

5.5 Résultats de l'évaluation des services RMS et SMS

Après l'exécution des scénarios de validation du projet DSE, les participants ont répondu à un ensemble de questionnaires afin d'évaluer le système. Les questionnaires ont été remplis par quatre ingénieurs qui ont participé aux tests de validation et qui travaillent dans de compagnies industrielles du domaine spatial. Ci-après, nous présentons leur bilan.

5.5.1 Questionnaire d'évaluation de gestion de responsabilité

Co-Des	Co-Ver	Question	Notes/10				Moyenne
			Ingénieur 1	Ingénieur 2	Ingénieur 3	Ingénieur 4	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q4. Croyez-vous que les fonctions disponibles pour organiser les responsabilités pendant la préparation du scénario sont appropriées? Sinon, expliquez pourquoi?	6	6	8	8	7
		Définition des rôles, des qualifications et des matières.	4	6	8	6	6
		Définition des participants et leurs attributs.	4	6	6	8	6
		Définition de l'activité commune du groupe.	4	4	6	6	5
		Définition des tâches et des sessions.	4	6	6	6	5,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q5. Trouvez-vous que les fonctions disponibles pour organiser les responsabilités pendant l'exécution du scénario sont appropriées ? Sinon, expliquez pourquoi?	6	4	8	8	6,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q6. Trouvez-vous que les fonctions disponibles pour organiser les responsabilités pendant l'évaluation du scénario sont appropriées ? Sinon, expliquez pourquoi?	4	6	8	6	6
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q7. Croyez-vous qu'il manque de fonctions de responsabilité à ajouter ?. Lesquelles ?	6	4	8	4	5,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q8. Pensez-vous que le temps d'accès aux fonctions de responsabilité est bon?	6	6	6	4	5,5

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q9. Quelle est votre opinion sur l'ergonomie de l'interface homme-machine ?	6	4	4	6	5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q10. Etes-vous satisfaite au sujet des impacts produits par DSE sur les performances dans la programmation du scénario et le travail en équipe ? Sinon, expliquez pourquoi:	6	6	4	6	5,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q11. Avez-vous eu des réductions dans le temps de réalisation du travail? Sinon, expliquez pourquoi ?	6	6	8	6	6,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q12. Avez-vous réalisé des économies? Sinon, expliquez pourquoi ?	6	6	10	8	7,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q13. Avez-vous réduit le nombre de déplacements? Sinon, expliquez pourquoi ?	6	6	10	8	7,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q14. Croyez-vous que la qualité du travail a été améliorée en utilisant DSE ?	6	6	8	6	6,5

5.5.2 Questionnaire d'évaluation de gestion de session

Co-Des	Co-Ver	Question	Notes/10				Moyenne
			Ingénieur 1	Ingénieur 2	Ingénieur 3	Ingénieur 4	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q.15 Trouvez-vous que les fonctions disponibles pour commander la session pendant l'exécution du scénario sont appropriées? Expliquez pourquoi?	6	4	6	6	5,5
		Invitation des participants à la session	4	6	6	6	5,5
		Liste des participants connectés à la session	4	6	8	8	6,5
		Manière de transférer le contrôle d'une application vers un participant.	6	4	10	4	6

		Manière d'effectuer la gestion d'une réunion distribuée en termes du passage de droit de parole à un participant, de faire converger les activités des participants à un but commun.	4	4	6	4	4,5
		Manière de finir la session.	4	6	6	4	5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q16. Quelle est votre opinion sur l'ergonomie de l'interface homme-machine ?	4	6	4	4	4,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q17. Etes-vous satisfait des performances du système (temps de réponse, réseaux) ? Sinon, expliquez pourquoi	4	4	6	4	4,5

5.5.3 Questionnaire d'évaluation des réunions distribuées

Co-Des	Co-Ver	Question	Notes/10				Moyenne
			Ingénieur 1	Ingénieur 2	Ingénieur 3	Ingénieur 3	
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q.21 Pensez-vous que la manière dont vous êtes invité par le chairman à une réunion distribuée est appropriée? Sinon, expliquez pourquoi?	6	6	8	8	7
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q.22 Pensez-vous que l'accès aux fonctions de groupware de DSE est facile (manière de lancer les fonctions, le temps d'accès aux fonctions de groupware, etc.) Sinon, expliquez pourquoi?	4	4	8	8	6
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q.23 Pensez-vous que les fonctionnalités courantes de DSE sont suffisantes pour réaliser vos activités? Sinon, expliquez pourquoi?	6	6	8	6	6,5
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Q24. Etes-vous clairement informé du statut de la session (session commencé, arrêté, liste des participants reliés, etc.). Sinon, expliquez pourquoi?	6	4	8	6	6

5.6 Conclusion

Les services RMS et SMS proposés font partie de l'environnement logiciel développé dans le cadre du projet européen Distributed Systems Engineering (DSE). Le scénario Preliminary Design Review (PDR) a servi de cadre de test. L'application de la revue préliminaire de conception au programme ATV en utilisant l'environnement DSE a été présentée en détail. Nous avons mis l'accent sur les contributions des services RMS et SMS. L'application des service RMS et SMS dans le scénario de test Co-Ver est disponible dans le rapport [MDV2002a].

Les résultats d'évaluation présentés ci-dessus concernent la première version développée au milieu du projet. La version actuelle tient compte de cette évaluation. Cette version a apporte des nouvelle solutions et des améliorations résumées ci-dessous.

Le service RMS a été amélioré au niveau de l'interface graphique homme-machine :

- La complexité dans le nombre de fenêtres ouvertes a été réduit. Toutes les fenêtres relatif aux services RMS et SMS ont été concentres sur une seul fenêtre application, de cette façon le nombre de fenêtres contrôlées par l'utilisateur a passé de huit à une seule.
- Ergonomie. L'interface a bénéficié aussi de l'intégration de barres de boutons avec des icônes, ceci a augmente significativement la facilité la tâche à l'utilisateurs pour utiliser le service.

Le service SMS a été aussi amélioré dans son interface graphique :

- Salle de réunion virtuel. Nous avons intégré une nouvelle fonctionnalité qui consiste à une salle de réunion virtuel avec la gestion de télé-présence. Cette salle est une salle virtuelle où les participants connectés sont représentés par des avatars. Cette salle fournit aussi la liste complète des participants attendus et qui contient la réponse à la convocation à la session.

Chapitre 6. Conclusion générale

Dans ce mémoire de thèse, nous avons présenté les travaux effectués autour de la définition des mécanismes pour la préparation et la coordination des sessions coopératives. La problématique que nous avons étudiée a porté essentiellement sur la conception et la mise en œuvre de services de préparation et de gestion de sessions qui fournissent les fonctions permettant aux participants de définir, d’initialiser, de chercher, de se joindre à, de quitter, d’ouvrir, de clore et de terminer une session. Ces services incluent aussi des mécanismes qui garantissent la mémoire de groupe : la messagerie instantanée et les notifications entre les participants, la gestion de la présence en temps réel et l’authentification.

6.1 Rappel des contributions

L’étude de la problématique associée à la coordination des sessions coopératives (chapitre 1) a établi les contraintes et les difficultés inhérentes à la gestion de ce type des sessions et donc les applications qui les supportent. Dans ce but, nous avons tout d’abord dégagé les besoins de la coordination des actions en général et la coordination des actions dans les sessions coopératives multi-utilisateurs multi-applications plus particulièrement. Principalement, nous avons identifié dans cette étape les différents types d’interdépendances entre participants et applications que nous avons classifié en quatre types de dépendances : inter-applications, intra-applications, inter-participants et participants-applications.

Modèle de coordination des sessions de coopération

Pour qu’une session de coopération atteigne son objectif, il faut que les interactions respectent un ensemble de règles de coordination bien établies. Ces règles doivent être

à la fois suffisamment restrictives pour éviter les interactions incohérentes et destructives et suffisamment permissives pour favoriser les interactions pertinentes et constructives.

La formalisation, la structuration et la généralisation des interdépendances, identifiées au chapitre 1, ont permis l'établissement d'un modèle de coordination de sessions. Ce modèle, nous a permis de créer un cadre formel capable de spécifier de manière générale les règles de coordination.

Notre modèle de coordination des sessions (chapitre 2) est basé sur la spécification des interactions entre les acteurs pendant une session. Il permet la définition des règles de coordination. Le modèle est fondé sur deux aspects, d'une part par la représentation des sessions de coopération par des ordres partiels étiquetés et d'autre part, l'utilisation de formules de premier ordre afin d'exprimer les règles de coordination sur les sessions de coopération. Nous avons défini trois modalités qui constituent la base de notre modèle : la *précédence*, l'*inhibition* et le *déclenchement*. Ces trois modalités sont suffisantes pour décrire les dépendances dans les sessions de coopération multi-applications.

Mécanismes de gestion des sessions de coopération

Le chapitre 3 s'est intéressé, quant à lui, au développement de deux services principaux. Le premier permet de planifier une session de coopération. Le deuxième permet de gérer le déroulement d'une session programmée en commençant par les invitations, et en terminant par la terminaison de la session tout en passant par différents états de contrôle dont l'ouverture et la clôture de la session. Des interfaces graphiques conviviales ont été développées et ont été mises à disposition des utilisateurs des nos services (administrateurs, animateurs de sessions ou simple participants).

Le service de préparation des sessions est basé sur la définition des sessions de coopération sur un modèle de structuration orienté rôle. Différents types d'attributs sont associés aux participants pour permettre de les adresser ou de les identifier de façon unique (par leur identité) ou de façon générique (par leurs propriétés).

Le service de gestion de sessions réalise la coordination des phases synchrones de coopération dans des sessions multi-applications. Il s'agit en particulier de gérer la constitution du groupe de participants, de gérer la présence des membres de ce groupe et de soutenir le chairman de la session dans sa gestion de l'évolution de l'état de la session. Les événements échangés permettent de garantir la validité de la session lors des changements relatifs à son état, relatifs à la structure de groupe ou aux rôles des participants tout au long de son cycle de vie.

Nous avons proposé une architecture d'exécution orientée événements pour soutenir le service de coordination des sessions. Cette architecture possède l'avantage de permettre la gestion des sessions distribuées au cours desquelles on n'exige pas toujours la présence de tous les participants pour le démarrage. L'arrivée tardive de participants est acceptée et l'état global est maintenu à jour et communiqué aux nouveaux arrivants.

Scénario de validation

Nous avons réalisé l'expérimentation et la validation des services développés dans le cadre d'un cas d'utilisation réel à une échelle réduite d'un scénario de

l'industrie spatiale dans le cadre de la phase de validation de l'environnement logiciel du projet DSE(chapitre 4). Ce scénario, appelé PDR, est le processus de revue préliminaire de conception « Preliminary Design Review ». Le scénario est basé sur la conception du programme de développement du segment de vol de l'ATV (Automated Transfer Vehicule) de la station spatiale international (ISS). Le PDR est une étape de révision pendant laquelle la conception préliminaire de l'ATV est contrôlée conforme aux plans industriels et aux normes internationales. La fabrication, l'assemblage, l'intégration et les plans d'essai sont vérifiés être conformes aux dates de livraison prévues. Pendant le processus PDR-ATV, des spécialistes des diverses domaines analysent un ensemble de documents de conception afin de trouver des désaccords vis-à-vis des exigences prédéfinis. Une base cohérente et vérifiée de documents de conception est un des résultats de cette activité.

L'expérimentation a été conduite en deux étapes : au milieu et en fin du projet. Cette procédure incrémentale nous a permis de prendre en compte le retour des ingénieurs de nos partenaires industriels sur la satisfaction des services développés afin d'aboutir à une version avec de nouvelles solutions et des améliorations au niveau de l'awareness (conscience du groupe) grâce à la gestion de la présence, et au niveau de la convivialité et la maîtrise de la complexité des fenêtres de l'interface graphique.

6.2 Perspectives

Pour terminer ce mémoire, nous allons classifier et détailler les perspectives du travail réalisé.

Modèle de préparation des sessions

Introduction de la QoS

Le modèle de structuration de sessions à travers des groupes génériques ouverts pourrait être étendu de par l'introduction des paramètres associés à la qualité de service requis. Ceci pourrait se faire par l'introduction de nouveaux attributs permettant de représenter les hiérarchies dans les groupes afin de pouvoir adapter, quand c'est nécessaire, la qualité de service par participant, par groupe ou par session.

Introduction de la gestion des connaissances (Knowledge Management)

Note modèle de préparation des sessions est basé sur la sélection des participants selon leurs rôles, leurs compétences et leurs thèmes d'intérêt. En étudiant et en adaptant les résultats des travaux du domaine *knowledge management* [KS2002], nous pouvons concevoir d'introduire la notion de connaissance, comme critère de recherche et de sélection, dans le référentiel de participants. Ainsi, on pourrait classifier et sélectionner les participants en répondant à des critères du type : les participants de la compagnie X ayant travaillé dans le projet Y et possédant une expérience de telle durée dans tel domaine acquise au cours des 6 derniers mois, etc.

Modèle de coordination de sessions de coopération

Le modèle de coordination des événements par l'utilisation des ordres partiels étiquetés et de la logique du premier ordre assure un cadre formel pour des développements futurs. Une première perspective de recherche est la spécification des

règles de coordination par la logique temporelle. Cette introduction nous permettrait de spécifier des règles de coordination par rapport à un point de synchronisation.

La vérification des propriétés de logique temporelle sur des spécifications à base d'ordres partiels est un domaine émergent dans le *model checking*. Nous pourrions bénéficier des résultats obtenus dans ce domaine pour déterminer la décidabilité des spécifications des sessions.

Service de préparation des sessions de coopération

Interoperabilité avec autres systèmes.

La réalisation actuelle du service de préparation des sessions RMS utilise une base de données MS-Access à travers JDBC pour stocker les configurations des participants, des groupe et des sessions. Cette implémentation ne permet pas la réutilisation des systèmes de bases de donnée préexistants sous un format différent. Pour assurer la pérennité et la réutilisation de ce service, il serait envisageable de remplacer la base de données MS-Access par un protocole d'annuaire standard, par exemple, LDAP.

Réalisation des groupes génériques

Actuellement, la version du service RMS ne prend pas en compte la structuration des groupes génériques, elle se limite à la définition des groupes de cas. Grâce à la conception des groupes génériques ouverts déjà effectuée dans cette thèse, il ne serait pas difficile d'étendre le service actuel pour implanter cette fonctionnalité.

Service de gestion des sessions de coopération

Intégration du service workflow

Les sessions de coopération se déroulent dans un cadre plus vaste, par exemple, dans des projets, des revues, des processus d'ingénierie concurrente, etc. Il serait donc intéressant de pouvoir associer la définition du flux des activités aux sessions. Nous nous sommes inspirés des nos travaux issus du DEA [Mol1999, MDD1999, MDD2000, MDD2000a]. Nous croyons qu'il est pertinent d'envisager l'intégration du service SMS avec la définition des processus workflow par des techniques de transformation de graphes. Dans ce cas, le résultat serait l'association de la définition des sessions avec la définition des activités dans un processus workflow.

Introduction d'une sémantique aux avatars

Dans la version actuelle de la réalisation du service SMS, des avatars représentant les participants connectés sont introduits dans la salle de réunion virtuelle. Ces avatars sont construits en 2D mais ils ont un comportement 3D, car leur taille dépend de leur position dans la salle virtuelle. S'ils reculent, leur taille devient plus petite, s'ils avancent, leur taille devient plus grande. Ceci donne l'impression à l'utilisateur d'une salle en 3D. Ce mécanisme pourrait être étendu par l'introduction d'une sémantique à la position des avatars. Une première sémantique pourrait être la distinction des participants actifs et des participants passifs. On peut imaginer que les avatars en premier plan, c'est à dire ceux qui se trouvent au premier plan de la scène et qui ont une taille importante représentent les participants qui sont entrain de

s'échanger des informations, ou qui sont en cours de réalisation de certaines tâches, c'est à dire des participants actifs. Par contre, les participants qui se trouveraient en arrière plan de la scène, et par conséquent associés à de plus petits avatars, seront des participants qui écoutent ou qui attendent les actions des autres, c'est à dire des participants passifs. On pourrait envisager aussi l'association des icônes au dessus des avatars pour rendre disponible des informations associées à l'état des utilisateurs, par exemple, qui a le droit de parole actuel, qui est en attente de ce droit de parole, qui possède les privilèges d'accès aux données de telle application, etc.

Intégration de la gestion des données.

Grâce à la définition des règles de coordination et à notre architecture hybride orientée événements, on peut envisager l'addition d'une nouvelle fonctionnalité relative à la gestion des données manipulées lors des sessions de coopération. Les règles de coordination pourraient servir à définir les dépendances entre ces données et les éléments déjà présents : les participants qui les produisent ou qui les consomment, les applications qui les transforment ou qui les analysent et les sessions qui les regroupent.

Sessions multi-instances multi-types

Les services RMS et SMS que nous avons réalisés supportent des sessions multi-types mais mono-instance, c'est à dire que dans une même session des applications de types différents peuvent être exécutées, mais chaque application ne comporte qu'une instance. On peut envisager l'extension du service pour traiter des sessions où coexistent plusieurs instances d'applications de types différents, c'est à dire, des sessions multi-instances multi-types.

Chapitre 7. Bibliographie de l'auteur

Contribution à ouvrages

- [DMN+2001] K. Drira, J.M.Molina Espinosa, O. Nabuco, L.M. Rodriguez Peralta, T. Villemur, «Cooperative environments for distributed systems engineering », Lecture Notes in Computer Science 2236, Springer, N°ISBN 3-540-43083-0, 2001, pp.107-151. Rapport LAAS No01599

Conférences avec actes

- [MFD2003] J.M.Molina-Espinosa, J. Fanchon, K.Drira, A Logical Model for Coordination Rule Classes in Collaborative Sessions, Rapport LAAS N°03132, IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE-2003), 9-11 juin 2003, Linz, Autriche, 5p.
- [RVDM2002] L.M. Rodriguez Peralta, T. Villemur, K. Drira, J.M. Molina Espinosa, Managing dependencies in dynamic collaborations using coordination diagrams, 6th International Conference on Principles of Distributed Systems (OPODIS'02), Reims (France), 11-13 décembre 2002, pp.29-42
- [MD2002] J.M. Molina Espinosa, K. Drira, A multi-modal coordination service model for cooperative distributed systems engineering Rapport LAAS N°02295, 2002 IEEE International Conference on Systems Man and Cybernetics (SMC'02), Hammamet (Tunisie), 6-9 Octobre 2002, 6p.

- [MDV2002] J.M. Molina Espinosa, K. Drira, T. Villemur, The responsibility management system for collaborative meetings scheduling in the distributed system engineering project Rapport LAAS No02234 IEEE International Workshop on Knowledge Media Networking (KMN'2002), Kyoto (Japon), 10-12 juillet 2002, 6p.
- [MDR2002] J.M. Molina Espinosa, K. Drira, L.M. Rodriguez Peralta, An event driven session management system for collaborative distributed systems engineering Rapport LAAS No02061 9th European Concurrent Engineering Conference (ECEC'2002), Modena (Italie), 15-17 avril 2002, pp.148-152
- [MND2001] J.M. Molina Espinosa, O. Nabuco, K. Drira, A UML model for session management in collaborative design for space activities Rapport LAAS No01087 8th European Concurrent Engineering Conference (ECEC'2001), Valence (Espagne), 18-20 avril 2001, pp.170-174
- [MDD2000] J.M. Molina Espinosa, K. Drira, M. Diaz, Modèle de description de procédures Workflow basé sur la réécriture de graphes Rapport LAAS No00118 Journées "Formalisation des Activités Concurrentes" (FAC'2000), Toulouse (France), 18-19 Mai 2000, pp.105-115
- [MDD2000a] J.M. Molina Espinosa, K. Drira, M. Diaz, A graph-grammar model for coordination definition in workflow process activities Rapport LAAS No99458 International Conference on Artificial and Computational Intelligence for Decision, Control and Automation in Engineering and Industrial Applications (ACIDCA'2000), Monastir (Tunisie), 22-24 Mars 2000, pp.168-173
- [MDD1999] J.M. Molina Espinosa, K. Drira, M. Diaz, Modelo de descripción de coordinación de actividades en procesos Workflow Rapport LAAS No99466 Taller de Sistemas Distribuidos y Paralelos del Segundo Encuentro Nacional de Computación (ENC'99), Pachuca (Mexique), 12-15 Septiembre 1999, 6p.
- Rapport DEA**
[Mol1999] J.M. Molina Espinosa, Modèles "Workflow" pour la description des procédures de coordination. Rapport de stage de DEA Programmation et Systèmes, INPT-ENSHEEIT, Rapport LAAS N°99462, Octobre 1999, 53p.
- Rapports de contrats**
[BCD+2001] B.Baurens, B. Cambou, K.Drira, J.M. Molina Espinosa, O. Nabuco, DSE V1 Integrated implementation report Rapport LAAS No01280 Project IST-1999-10302, 18 juillet 2001, 254p
- [DRN+2001] K. Drira, L.M. Rodriguez Peralta, O. Nabuco, J.M. Molina Espinosa, S. Pomares Hernandez, DSE V1 Architecture and technical specifications Rapport LAAS No00612 Project IST-

- [DRN+2000] 1999-10302, 2 mai 2001, 173p.
K. Drira, L.M. Rodriguez Peralta, O. Nabuco, J.M. Molina Espinosa, DSE Technology & products; State-of-the-art report Rapport LAAS No00614 Project IST-1999-10302, juin 2000, 226p.
- Rapports LAAS**
- [MD2003] J.M.Molina-Espinosa, K.Drira, A formal representation of the coordination requirements for the collaborative verification activity within the DSE project, Rapport LAAS N°03028, janvier 2003, 13p.
- [MDV2002a] J.M.Molina-Espinosa, K.Drira, T.Villemur, The responsibility management system for collaborative meetings scheduling in the distributed system engineering project, Rapport LAAS N°02356, septembre 2002, 11p.

Chapitre 8. Bibliographie

- [ABC+2001] D. Avino, R. Becchini, P. Chilaev, H. Follet, V. Krivtsov, A. Martelli et V. Volochinov, Chapter 3 Relevant Existing Practices, in : Cooperative Environments, LNCS 2236, 2001, pp.15-39
- [AKK+1996] H. Abdel-Wahab, B. Kvande, O. Kim, J.P. Favreau, Using Java for Multimedia Collaborative Applications, Proceedings of PROMS'96: Third International Workshop On Protocols for Multimedia Systems, 1996, pp. 49-62, Madrid, October 1996.
- [AKK+1997] H. Abdel-Wahab, B. Kvande, O. Kim, J.P. Favreau, An Internet Collaborative environment for Sharing Java Applications, Proceedings of the 5th IEEE Computer Society Workshop on Future Trends of Distributed Computing Systems (FTDCS'97), Tunis, Tunisia, pp.112-117, October 29 - 31, 1997
- [AKK+1998] H. Abdel-Wahab, P. Kabore, O. Kim, J.P. Favreau, Replication Management of Application Sharing for Multimedia Conferencing and Collaboration, Proceedings of the Second IFIP/IEEE International Conference on Management of Multimedia Networks and Services '98, Versailles, France, November 16-18, 1998
- [AKK+1999] H. Abdel-Wahab, O. Kim, P. Kabore, J.P. Favreau, Java-based Multimedia Collaboration and Application Sharing Environment, Colloque Francophone sur l'Ingenierie des Protocoles (CFIP'99), Nancy, France, April 26 - 29, 1999
- [Bas2001] Béatrice Bastier, Outils de gestion de sessions et de groupes pour l'ingénierie coopérative distribuée, Rapport LAAS No01653 Diplôme d'Ingénieur, Centre National des Arts et Métiers, Toulouse,

- 7 Décembre 2001, 99p
- [Bau2001] B. Baurens, DSE V2 Integrated Implementation Detailed Executive Summary, D3.3.2, Project IST-1999-10302, 20 decembre 2001, 81 p.
- [BCD+2001] B.Baurens, B. Cambou, K.Drira, J.M. Molina Espinosa, O. Nabuco, DSE V1 Integrated implementation report Rapport LAAS No01280 Project IST-1999-10302, 18 Juillet 2001, 254p
- [BCF+1997] Beca L., Chang G., Fox G.C., Jurga T., Olszewski K., Podgorny M., Sokolowski P., Stachowiak T. et Walczak K., « Tango – a Collaborative Environment for the World-Wide Web. », Northeast Parallel Architectures Center, Syracuse University, New York, 1997. http://www.collabworx.com/legacy/tango/Documents/Papers/WhitePaper/white_paper.html
- [BHI1993] S.A. Bly, S.R. Harrison et S. Irwin, « Mediaspace : Bringing people together in a video, audio and computing environment », Communications of the ACM, vol. 36, numéro 1, pp. 28-47, janvier 1993.
- [Bur1999] RichBurridge, Java Shared Data Toolkit 2.0, User Guide, Sun Microsystems, Inc., Octobre 1999, URL : <http://java.sun.com/products/java-media/jsdt/>
- [BZB+1997] B. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation protocol (RSVP) -- version 1 functional specification", RFC 2205, October 1997.
- [CGJ+1998] A. Chanbert, E. Grossman, L. Jackson, S. Pietrovicz, NCSA Habanero- Synchronous collaborative framework and environment, Software Development Division at the National Center for Supercomputing Applications, white paper, 1998
- [CMB+1990] T. Crowley, P. Milazzo, E. Baker, H. Forsdick, R. Tomlinson, « MMConf: An Infrastructure for Building Shared Multimedia Applications », CSCW90 : Proceedings of the Conference on Computer-Supported Cooperative Work, Los Angeles, CA : ACM 1990, pp.329-342.
- [CR1997] C. Mani Chandy et Adam Rifkin, Systematic Composition of Objets in Distributed Internet Applications : Processes and Sessions, in 30th Hawaii International Conference on System Sciences in January 1997, 10p.
- [CWH1998] Mary Campione, Kathy Walrath, Alison Huml, The Java Tutorial Continued: the rest of the CDK, Addison Wesley, ISBN 0201485583, 1998, 950p.
- [DD2000] K. Drira et M. Diaz, « Graph-grammar based coordination in inter- corporate computer supported collaborative activities », Annual Review of Scalable Computing, Series on Scalable Computing. Vol. 2, Ed. Y.C. Kwong, Singapore University Press, ISBN 981-02-4413-4, 2000, chapter 1, pp. 1-27.
- [DeMi1995] G. De Michelis, « Computer Support for Cooperative Work: Computers between Users and Social Complexity », in C. Zucchermaglio, S. Bagnara, S. Stucky (editors), Organizational Learning and Technological Change, Springer Verlag Berlin 1995,

- pp. 307-330.
- [DMN+2001] K. Drira, J.M.Molina Espinosa, O. Nabuco, L.M. Rodriguez Peralta, T. Villemur, «Cooperative environments for distributed systems engineering », Lecture Notes in Computer Science 2236, Springer, N°ISBN 3-540-43083-0, 2001, pp.107-151. Rapport LAAS No01599
- [DOM1997] H.P. Dommel, J.J. Garcia-Luna-Aceves; Floor control for multimedia conferencing and collaboration.; Springer-Verlag 1997.
- [DRN+2000] K. Drira, L.M. Rodriguez Peralta, O. Nabuco, J.M. Molina Espinosa, DSE Technology & products; State-of-the-art report Rapport LAAS No00614 Project IST-1999-10302, Juin 2000, 226p.
- [DRN+2001] K. Drira, L.M. Rodriguez Peralta, O. Nabuco, J.M. Molina Espinosa, S. Pomares Hernandez, DSE V1 Architecture and technical specifications Rapport LAAS No00612 Project IST-1999-10302, 2 Mai 2001, 173p.
- [DV1994] M.DIAZ , T.VILLEMUR. Formation of private conversation subgroups in a cooperative group of processes. Journal of the Brazilian Computer Society, Vol.1, N°1, pp.46-58, Juillet 1994
- [Edw1994] W. Keith Edwards, «Session Managemnt for Collaborative Applications », Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW), Chapel Hill, NC, Octobre 22-26 , 1994.
- [EGR1991] Clarence A. Ellis, S.J. Gibbs, et G.L. Rein, «Groupware some issues and experiences », communications of the ACM, janvier 1991, vol. 34, numéro 1.
- [Eli1998] Clarence A. Ellis, « A Framework and Mathematical Model for Collaboration Technology », Coordination Technology for Collaborative Applications : Organizations, Processes, and Agents, LNCS 1364, Springer, pp. 121-144.
- [Eri1994] Hans Eriksson, Mbone : The multicast Backbone, Communications of the ACM, 37(8), pp 54-60, 1994.
- [EW1994] Clarence A. Ellis et Jacques Wainer, « A conceptual Model of Groupware », Proccedings of ACM 1994 Conference on Computer Supported Cooperative Work, octobre 22-26, 1994, Chapel Hill, North Carolina, pp. 79-88.
- [Fed1993] Federal Information Processing Standards (FIPS) Publication 183, “ Integration Definition for Function Modeling (IDEF0) ”, National Institute of Standards and Tachnology, 21 Décembre 1993.
- [FPM+2000] H.Follet, J. Perennec, J.P Martinrichond, Ph. Lacan, A. Vankov, D.Cortese, A.Todino, Co-Des User Requirements Document, Commission of the European Comunities Directorate-General Information Society, DSE Systems Engineering, IST-1999-10302, Deliverable 1.1, juin 2000, 53p.
- [Gru1991] Jonathan Grudin, « CSCW Introduction », Communications of the ACM, Vol. 34, No. 12, decembre 1991, pp 30-34.
- [Gru1995] Jonathan Grudin, « Computer-Supported Cooperative Work : History and Focus », IEEE Computer 27(5), pp.19-26,mai 1995
- [GS1987] Irene Greif et Sarin Sunil, « Data sharing in group work », ACM

- transactions on Office Information Systems, Vol. 5 Numéro 2, pp187-211, avril 1987.
- [Hab1997] Henri Habrias, « Dictionnaire encycloédique du génie logiciel », Masson, Paris 1997.
- [Han1996] M. Handley, « SAP: Session announcement protocol, » Internet Draft, Internet Engineering Task Force, Nov. 1996.
- [HCB1996] M. Handley, J. Crowcroft et C. Bormann, The Internet Multimedia Conferencing Architecture. Internet Draft, MMUSIC Working Group, February 1996.
- [HJ1998] M. Handley, et V. Jacobson, « SDP: session description protocol », RFC 2327, April 1998.
- [HSS+1999] M. Handley, H. Schulzrinne, E. Schooler, J. Rosenberg, SIP: Session Initiation Protocol, RFC 2543, Mars 1999
- [ITU1995] International Telecommunications Union, Data Protocols for Multimedia Conferencing, Draft Recommendation T.120, 1995
- [Joh1982] Peter Johnson-Lenz et Trudy Johnson-Lenz, « Groupware : The process and impacts of design choices. In E. B. Kerr & S. R. Hiltz editors, Computer-Mediated Communication Systems : Status and Evaluation, pp. 45-55. Academic Press, N.Y., 1982
- [Joha1988] R. Johansen, « Groupware, Computer support dor business teams », New York : the free press, Macmilan Inc. 1988.
- [Jur1997] T. Jurga, Session Management in Web Collaboratory Systems, Master thesis Northeast Parrallel Architectures Center, Syracuse University, New York, 1997
<http://trurl.npac.syr.edu/tomj/mt/thesis/html/index.html>
- [KKF+1997] O. Kim, P. Kabore, J.P. Favreua et H. Abdel-Wahab, Issues in Platform-Independent Support for Multimedia Desktop Conferencing and Application Sharing, Proceedings of the Seventh IFIP Conference on High Performance Networking (HPN'97), White Plains, NY, pp. 115-139, April 18 - May 2, 1997
- [Kle1998] Mark Klein, « Coordination Science : Challenges and Directions », Coordination Tecnology for Collaborative Applications : Organizations, Processes, and Agents, LNCS 1364, Springer, pp. 161-176.
- [Kli1991] Rob Kling, « Cooperation, Coordination and Control in Computer-Supported Work », Communications of the ACM, vol.34, numéro 12, decembre 1991
- [KS2002] Koji Kida et Hideo Shimazu, Ubiquitous Knowledge Managemnt – Enabling an office-work scheduling tool for corporate knowledge sharing, in Proccedings of the IEEE “nternational Workshop on Knowledge Media <Networking, Kyoto, Japon, 10-12 juillet, pp. 99-104
- [LBS1998] Geoffrey Lewis, Steven Barber et Ellen Siegel, Programming with Java IDL:Developing Web Applications with Java and CORBA, Wiley Computer Publishing, ISBN0471247979, 1998, 322p.
- [LFK1988] Leland M., Fish R., Kraut R., « Collaborative Document Production using Quilt », Proceedings of the Conference on Computer-Supported

- [Lju1998] Cooperative Work, CSCW 88, sep. 1988
 Fredrik Ljungberg, « Exploring CSCW Mechanisms to Realize Constant Accessibility Without Inappropriate Interaction », Scandinavian Journal of Information Systems, Vol. 11 numéro 1, 1998.
- [Mar1867] K. Marx, Le Capital. Critique de l'économie politique (1867), Livre I (le développement de la production capitaliste) (tome II), Section IV : La production de la plus-value relative, chapitre XIII : coopération, édition électronique réalisée à partir du livre de Karl Marx (1867), Le Capital. Critique de l'économie politique. Traduction française de Joseph Roy entièrement revue par Karl Marx. Livre premier. Tome II (Sections IV, V et VI). Paris: Éditions sociales, 1948, 246 pages
- [MC1990] Thomas W. Malone et Kevin Crowston, « What is Coordination Theory and How Can It Help Design Cooperative Work Systems ? », Proceedings of the Conference on Computer-Supported Cooperative Work, Octobre 7-10, 1990 Los Angeles, CA., pp 357-370.
- [MC1994] Thomas W. Malone et Kevin Crowston, « The Interdisciplinary Study of Coordination », ACM Computing Surveys, Vol. 26, No. 1, Mars 1994, pp. 87-119.
- [MD2002] J.M. Molina Espinosa, K. Drira, A multi-modal coordination service model for cooperative distributed systems engineering Rapport LAAS N°02295, 2002 IEEE International Conference on Systems Man and Cybernetics (SMC'02), Hammamet (Tunisie), 6-9 Octobre 2002, 6p.
- [MD2003] J.M.Molina-Espinosa, K.Drira, A formal representation of the coordination requirements for the collaborative verification activity within the DSE project, Rapport LAAS N°03028, Janvier 2003, 13p.
- [MDD1999] J.M. Molina Espinosa, K. Drira, M. Diaz, Modelo de descripcion de coordinacion de actividades en procesos Workflow Rapport LAAS No99466 Taller de Sistemas Distribuidos y Paralelos del Segundo Encuentro Nacional de Computacion (ENC'99), Pachuca (Mexique), 12-15 Septembre 1999, 6p.
- [MDD2000] J.M. Molina Espinosa, K. Drira, M. Diaz, Modèle de description de procédures Workflow basé sur la réécriture de graphes Rapport LAAS No00118 Journées "Formalisation des Activités Concurrentes" (FAC'2000), Toulouse (France), 18-19 Mai 2000, pp.105-115
- [MDD2000a] J.M. Molina Espinosa, K. Drira, M. Diaz, A graph-grammar model for coordination definition in workflow process activities Rapport LAAS No99458 International Conference on Artificial and Computational Intelligence for Decision, Control and Automation in Engineering and Industrial Applications (ACIDCA'2000), Monastir (Tunisie), 22-24 Mars 2000, pp.168-173
- [MFD2003] J.M.Molina-Espinosa, J. Fanchon, K.Drira, A Logical Model for Coordination Rule Classes in Collaborative Sessions, Rapport LAAS N°03132, IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE-2003), 9-11 juin, Linz, Autriche, 5p.
- [MDR2002] J.M. Molina Espinosa, K. Drira, L.M. Rodriguez Peralta, An event

- driven session management system for collaborative distributed systems engineering Rapport LAAS No02061 9th European Concurrent Engineering Conference (ECEC'2002), Modena (Italie), 15-17 Avril 2002, pp.148-152
- [MDV2002] J.M. Molina Espinosa, K. Drira, T. Villemur, The responsibility management system for collaborative meetings scheduling in the distributed system engineering project Rapport LAAS No02234 IEEE International Workshop on Knowledge Media Networking (KMN'2002), Kyoto (Japon), 10-12 Juillet 2002, 6p.
- [MDV2002a] J.M.MOLINA-ESPINOSA, K.DRIRA, T.VILLEMUR, The responsibility management system for collaborative meetings scheduling in the distributed system engineering project, Rapport LAAS N°02356, Septembre 2002, 11p.
- [MND2001] J.M. Molina Espinosa, O. Nabuco, K. Drira, A UML model for session management in collaborative design for space activities Rapport LAAS No01087 8th European Concurrent Engineering Conference (ECEC'2001), Valence (Espagne), 18-20 Avril 2001, pp.170-174
- [Mol1999] J.M. Molina Espinosa, Modèles "Workflow" pour la description des procédures de coordination. Rapport de stage de DEA Programation et Systèmes, INPT-ENSHEEIT, Rapport LAAS N°99462, Octobre 1999, 53p.
- [NCSA2001] National Center for Supercomputing Applications, NCSA Habanero – V. 3.0 and ISAAC V 1.0 Users guide, University of Illinois, 2001. <http://www.isrl.uiuc.edu/isaac/Habanero/3.0/habanero3.0.html>
- [OAB1999] João Orvalho, Tiago Andrade, Fernando Boavida, "A Conference Control Service Based on the CORBA Event Service", Proceedings of the 2nd Conference on Telecommunications, Instituto de Telecomunicações (Portugal), Sesimbra, Portugal, 15-16 April, 1999.
- [OAB1999a] João Orvalho, Tiago Andrade, Fernando Boavida, "Extension to CORBA Event Service for a Conference Control System", Proceedings of the IEEE Multimedia Systems 99 - International Conference on Multimedia Computing and Systems, ICMCS'99, IEEE TC on Multimedia Computing, University of Florence, Centro Affari, Florence, Italy, June 7-11, 1999.
- [OAFB1999] João Orvalho, Tiago Andrade, Luís Figueiredo, Fernando Boavida, "CONCHA – CONFERENCE system based on java and corba event service CHANNELS", Proceedings of SPIES's symposium on Voice, Video, and Data Communications conference on Quality of Service Issues Related to Internet II, Boston, MA, USA, September 19-22, 1999.
- [OMG2003] Object Management Group, Unified Modeling Language 1.5 specification, 2003, URL: <http://www.omg.org/technology/documents/formal/uml.htm>
- [PHRM1990] J.F. Patterson, R.D. Hill, S.L. Rohall et W.S. Meeks, « Rendezvous : An Architecture for Synchronous Multi-user Applications », Proceedings of the ACM Conference on Computer-Supported

-
- [Pra1987] Cooperative Work (CSCW), Los Angeles CA, 1990, pp.317-328.
Pratt.W. Modeling concurrency with partial orders. *Int. J. Parallel Programming* 15 (1987) 33-71.
- [PMS1997] A. Prakash, M. Raynal, M. Singhal, An Adaptive Causal Ordering Algorithm Suited to Mobile Computing Environments, *Journal of Parallel and Distributed Computing*, Mars 1997, pp.190-204.
- [Ree1997] George Reese, *Database Programming with JDBC and Java*, O'Reilly, ISBN 1565922700, 1997, 224p.
- [RG1997] Mark Roseman et Saul Greenberg, Building Groupware with GroupKit. In M. Harrison (Ed.) *Tcl/Tk Tools*, p535-564, O'Reilly Press.
- [Rou1997] Nicolas Roussel, « Au-delà du mediaspace : Un modèle pour la collaboration médiatisée », . In *Actes des neuvièmes journées francophones sur l'Interaction Homme Machine (IHM'97)*, *Futuroscope*, pages 159-166, Septembre 1997.
- [RRH2000] Anthony Ralston, Edwin D. Reilly et David Hemmendinger, « Encyclopedia of Computer Science », Nature Publishing Group, UK, 2000.
- [RSC+2002] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, SIP: Session Initiation Protocol, RFC 3261, June 2002
- [RVDM2002] L.M. Rodriguez Peralta, T. Villemur, K. Drira, J.M. Molina Espinosa, Managing dependencies in dynamic collaborations using coordination diagrams, 6th International Conference on Principles of Distributed Systems (OPODIS'02), Reims (France), 11-13 Décembre 2002, pp.29-42
- [SC+1995] Daniel Salber, Joëlle Coutaz, Dominique Decouchant, Michel Riveill, « De l'observabilité et de l'honnêteté : le cas du contrôle d'accès dans la Communication Homme-Homme Médiatisée », *Proceedins IHM 1995*, editorial CEPAD, pp. 27-34.
- [SCF+1996] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, « RTP: a transport protocol for real-time applications », RFC 1889, Internet Engineering Task Force, Jan. 1996.
- [Sch1991] Schmidt K., « Cooperative work : A conceptual framework », J. Rasmussen, B. Brehmer et J. Leplat editeurs, *Distributed Decision Making Cognitive Models for Cooperative Work*, Ed. John Wiley & Sons, pp.75-109, 1991
- [Sch2001] René W. Schmidt, *Java Network Launching Protocol & API Specification*, Sun Microsystems, version 1.0.1, 2001, 73p.
- [SLR1998] H. Schulzrinne, R. Lanphier, and A. Rao, « Real time streaming protocol (RTSP) », RFC 2326, April 1998.
- [SNN+1997] Sandeep K. Singhal, Binh Q. Nguyen, Richard Redpath, Jimmy Nguyen, Michael Fraenkel, *InVerse : Designing an Interactive Universe Architecture for Scalability and Extensibility*, 6th International Symposium on High Performance Distributed Computing (HPDC '97), Portland, OR, August 05 - 08, 1997, pp61-70,

- [Sof1981] SofTech, INC., "Integrated Computer-Aided Manufacturing (ICAM) Architecture Part II: Volume IV – Function Modeling Manual (IDEF0)," Air Force Materials Laboratory, Writing-Patterson AFB, OH, Juin 1981.
- [Sun1998] Sun, JavaMail API Design Specification, Version 1.1, Sun Microsystems Inc., 1998, 92p.
- [TN1999] S.Terzis and P. Nixon, "Building the next generation groupware: A survey of groupware and its impact on the virtual enterprise", TCD-CS-1999-08, Computer Science Department, Trinity College, February 1999.
- [TP1999] G. Texier, N. Plouzeau, Automatic Management of Sessions in Shared Spaces. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'99), Las Vegas (USA), 28 June - 1st July 1999, Vol.IV, pp.2115-2121.
- [Tro1997] Christophe Tronche, « Applications collaboratives : exemples, architectures et services », Calculateurs Parallèles, Vol. 9 n° 2,1997, pp 161-182.
- [UML1997] Unified Modeling Language, Object Constraint Language Specification, version 1.1, septembre 1997, 32 p. URL : <http://www.rational.com/uml>
- [W3C2000] W3C, Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 October 2000, URL : <http://www.w3.org/TR/REC-xml>
- [WFK+1996] Erik Wilde, Pascal Freiburghaus, Daniel Koller, Bernhard Plattner, « A Group and Session Management System for Distributed Multimedia Applications » Giorgio Ventre, Jordi Domingo-Pascual, André Dantine (Eds.): Multimedia Telecommunications and Applications, Third International COST 237 Workshop, Barcelona, Spain, November 25-27, 1996, Proceedings. Lecture Notes in Computer Science 1185 Springer 1996, ISBN 3-540-62096-6, pp. 1-22