Une architecture pour la collaboration synchrone à distance appliquée à la visualisation et à la modélisation dans les géosciences :

Luciano Pereira dos Reis

Institut National
Polytechnique de Lorraine

École Nationale Supérieure de Géologie

École doctorale IAE + M

UNE ARCHITECTURE POUR LA COLLABORATION SYNCHRONE À DISTANCE APPLIQUÉE À LA VISUALISATION ET À LA MODÉLISATION DANS LES GÉOSCIENCES

THÈSE

Présentée et soutenue publiquement le 9 mai 2006
À l'École Nationale Supérieure de Géologie

pour l'obtention du

Doctorat de l'Institut National Polytechnique de Lorraine
(Spécialité Informatique)

par

Luciano PEREIRA DOS REIS

Composition du jury :

Rapporteurs:	Marcelo GATTASS
	Pascal GUITTON
Examinateurs:	Christine FERRARIS
	Claude GODART
	Jean-Laurent MALLET
Invité:	Fabien BOSQUET
Directeur de Thèse :	Jean-Claude PAUL

LORIA
Laboratoire Informatique et Analyse des Données
Rue du Doyen Marcel Roubault – 54500 VANDOEUVRE – Tél. 03 83 59 64 35

Remerciements

Merci à toutes les nombreuses personnes qui ont rendu mon travail possible pendant cette longue période, avec leur collaboration et soutien scientifique, moral ou affectif.

En premier lieu, je tiens à remercier Jean-Claude Paul et Jean-Laurent Mallet pour la confiance qu'ils m'ont accordée en acceptant de diriger cette thèse, et pour avoir toujours soutenu mon travail.

Je remercie également les membres du jury, Marcelo Gattass, Pascal Guitton, Christine Ferraris et Claude Godart de m'avoir fait l'honneur d'être les rapporteurs et examinateurs de cette thèse.

Je tiens spécialement à remercier Fabien Bosquet pour son aide permanente, sans qui ce travail n'aurait pas été possible.

Un grand merci à tous mes compagnons de labeur du LIAD, du Loria, d'Earth Decision Sciences, de VSP Technologies (Nancy), de Petrobras et de PUC/Tecgraf (Rio), pour leurs importantes contributions. Merci particulièrement aux copains du groupe de recherche Gocad pour l'amitié, et a Alberto Raposo et Marcelo Gattass pour leur orientation académique lorsque j'étais au Brésil.

Un merci tout particulier à Mme Cugurno pour son aide très amicale, bien au-delà de ses responsabilités formelles, pour la résolution des problèmes pratiques tout au long de ce travail et de mon séjour à Nancy.

Je remercie Petrobras de m'avoir donné l'opportunité de mener cette thèse, et en particulier mes chefs directs pendant cette période, Anelise Quintão Lara et Antônio Carlos Capeleiro Pinto, pour leur soutien continu.

Enfin, merci spécialement à ma famille et à mes amis au Brésil pour supporter mes longues absences, même lorsque de retour à Rio quand je partageais mon temps entre la thèse et le travail chez Petrobras.

Table des matières

Résumé **7**
Abstract **8**

Introduction **9**

1 Modélisation et visualisation collaboratives dans les géosciences **13**
1.1 Introduction 13
1.2 Géomodèles 14
1.3 Flux des travaux en amont 16
1.4 Activités coopératives 18
1.4.1 Modélisation géologique 18
1.4.2 Planification et guidage des puits 21
1.4.3 Réalité Virtuelle 22
1.5 Systèmes de géosciences associés 24
1.6 Cas d'usage pour la collaboration à distance 25
1.6.1 Consultation et modélisation collaborative 25
1.6.2 Formation 26
1.6.3 Planification et guidage de puits 27
1.6.4 Géoguidage avec réalité virtuelle 27
1.7 Besoins et conséquences 28
1.7.1 Petits groupes 29
1.7.2 Haute performance graphique 29
1.7.3 Communication multimédia intégrée 29
1.7.4 Réutilisation de fonctionnalité mono-utilisateur 30
1.7.5 Contrôle de concurrence 30
1.7.6 Conscience de groupe 31
1.7.7 Interfaces hétérogènes 31

1.7.7 Conditions d'opération hétérogènes 31

2 Collaboration à Distance 33
2.1 Introduction 33
2.2 Classification des systèmes coopératifs 36
2.3 Collaboration synchrone à distance 37
2.3.1 Systèmes à collaboration implicite 37
2.3.2 Systèmes à collaboration explicite 40
2.4 Caractérisation architecturale 42
2.4.1 Réplication 45
2.4.2 Degré de conscience de collaboration 46
2.4.3 Modules externes 47
2.5 Questions de systèmes coopératifs 47
2.5.1 Contrôle d'utilisation 47
2.5.2 Contrôle de concurrence 49
2.5.3 Coordination 52
2.5.4 Gestion de session 53
2.5.5 Conscience de groupe 54
2.5.6 Mise en réseau 55
2.5.7 Sécurité 56
2.6 Systèmes de collaboration associés 57
2.6.1 Boîtes à outils et plateformes 57
2.6.2 Composants de coordination 58
2.6.3 Visualisation scientifique collaborative 61
2.6.4 CAO collaborative 62
2.6.5 Communication vidéo 62

3 Architecture de collaboration 65
3.1 Introduction 65
3.2 Architecture Gocad 66
3.2.1 Motifs de Conception 66
3.2.2 Modules d'extension 67

3.3 Architecture NetGocad 70
3.3.1 Fonctionnement 72
3.3.2 Canaux de collaboration 74
3.3.3 Contrôle de canaux 80
3.3.4 Spécification de collaborations 83
3.3.5 LuaConférence 86
3.4 Visioconférence 89
3.4.1 Outils associés 90
3.4.2 Java Media Framework 91
3.4.3 Modes d'opération CSVTool 93
3.4.4 Contrôle de Participation 97
3.4.5 Transmission d'écran 104
3.4.6 Contrôle de concurrence 104
3.5 Réalité virtuelle 105
3.5.1 Fonctionnalité du plugin RV 105
3.5.2 Collaboration à distance 106
3.5.3 Collaboration hétérogène 107
3.5.4 Haptique 109
3.6 Acquisition de données à distance 110

4 Application 113
4.1 Introduction 113
4.2 Géoguidage 1 114
4.3 Géoguidage 2 115
4.4 Formation 120
4.5 Ordinateur de bureau - réalité virtuelle 122
4.6 Discussion 123

Conclusion 125

Bibliographie 129

Appendix 1: NetGocad's user interface 141

Appendix 2: NetGocad IDL..... 143

Appendix 3: Lua and related plugins 147

Résumé

Les systèmes de modélisation collaborative sont encore rares dans la plupart des domaines techniques, notamment dans les géosciences, parce que leur développement implique beaucoup de questions technologiques et organisationnelles et généralement demande une nouvelle conception majeure des applications mono-utilisateur déjà opérationnelles.

Dans cette thèse, nous proposons une architecture permettant la transformation d'une application existante dans un système coopératif, exclusivement au moyen de la composition dynamique de modules d'extension (*runtime plugins*). L'architecture tient compte des demandes de modélisation et visualisation dans les géosciences et permet l'intégration dans l'application de base de services de coopération, de communication multimédia et d'un schéma de coordination simple et extensible basé sur les rôles, fournissant des mécanismes intégrés de diffusion et de contrôle d'utilisation (*floor control*) pour les canaux de collaboration présentés.

Nous envisageons aussi un modèle de travail dans lequel des sessions de collaboration peuvent impliquer l'utilisation d'interfaces d'utilisateurs hétérogènes (ordinateur de bureau et réalité virtuelle) par les participants à distance, et nous décrivons l'application du système dans des cas d'usage opérationnels.

Abstract

Collaborative modeling systems are still rare in most technical domains, particularly in the geosciences, because their development involves many technological and organizational issues and usually requires major redesign of operational single-user applications.

In this thesis we propose an architecture that enables the transformation of an existing application into a collaborative system, exclusively through the dynamic composition of run-time plugins. It takes into account the requirements of modeling and visualization in the geosciences, and supports the integration into the base application of cooperation services, multimedia communication, and a simple and extensible role-based coordination scheme that provides integrated broadcasting and floor-control mechanisms for the introduced collaboration channels.

We also consider a work model in which collaboration sessions may involve the use of heterogeneous user interfaces by the remote participants (desktop and virtual reality), and discuss the application of the system in operational scenarios.

Introduction

La création de modèles tridimensionnels en sciences et ingénierie est une activité complexe, demandant une collaboration étroite entre divers spécialistes et l'intégration d'informations à partir de nombreuses sources. Traditionnellement ceci impliquait l'utilisation de différentes applications et le transfert successif des résultats d'un spécialiste à l'autre. Cependant, l'évolution de l'informatique et de l'infographie au cours de la dernière décennie, combinée avec le progrès des méthodes de modélisation géométrique et de visualisation scientifique, a encouragé l'émergence d'une nouvelle génération d'outils intégrés de conception assistée par ordinateur qui permettent davantage de collaboration efficace pour mettre en place des modèles unifiés.

La prochaine étape dans cette évolution est l'utilisation de systèmes de collaboration à distance pour permettre à différentes personnes à des endroits différents de travailler ensemble en même temps, à partir de leurs propres postes de travail avec des modèles partagés. Ceci est maintenant un but important dans beaucoup de branches de la science et de l'ingénierie. Certaines industries en particulier - aéronautique, automobile, pétrole et gaz - qui ont fait de gros investissements en modélisation tridimensionnelle et visualisation haut de gamme, recherchent actuellement fermement à oeuvrer vers une collaboration à distance en "temps réel" (synchrone), étant donné les bénéfices élevés en jeu.

Dans ce travail, qui correspond à un plus large programme de recherche de la compagnie brésilienne de pétrole et de gaz (Petrobras), notre attention particulière est dirigée sur le problème de la collaboration à distance appliquée à la modélisation et visualisation de modèles géoscientifiques de la subsurface. La capacité d'employer efficacement les technologies de collaboration dans ces activités constitue un atout stratégique pour les compagnies de pétrole et de gaz dans un marché global, puisque cela leur permet de profiter pleinement de leur meilleure compétence, facilitant une intense coopération parmi des professionnels répartis géographiquement, dans des situations où cela ne serait pas faisable autrement.

Une collaboration à distance peut étendre à des équipes éloignées les bénéfices considérables atteints durant les dernières années grâce à l'utilisation de la modélisation et de la visualisation tridimensionnelle pour la collaboration en colloque. Ainsi, ceci augmente la capacité d'organisation pour négocier quelques problèmes urgents dans cette industrie: la demande pour la réduction du temps de cycle des projets, dans le but de minimiser les coûts et accélérer le retour sur investissements; le besoin de modèles avec une précision plus élevée et moins d'incertitude, intégrant

efficacement des informations pour des disciplines multiples; le besoin d'incorporation dans les modèles d'une quantité croissante de données dynamiques capturées par les dispositifs à senseur et reçues d'endroits éloignés en temps réel; et le besoin d'accélérer le transfert de connaissance en raison de la pénurie de spécialistes seniors.

Cependant, malgré le fort intérêt que cette technologie suscite, actuellement à peine quelques outils de visualisation et modélisation géoscientifiques commencent à offrir des ressources limitées de collaboration à distance, en raison de l'effort de développement que cela représente. Ainsi, la question principale dans ce travail est l'étude d'une solution architecturale qui facilite l'introduction des fonctionnalités de collaboration à distance dans un système existant. Le défi réside dans le fait qu'il est plus facile de construire une application pour la collaboration dès le début que d'essayer d'adapter une déjà existante [LYS+01].

Contrairement aux boîtes à outils (*toolkits*) et architectures cadres (*frameworks*) existants, nous n'attendons pas le développement de nouveaux outils conformes à une architecture prédéfinie. Notre but est plutôt de permettre la transformation d'applications mono-utilisateur opérationnelles en applications coopératives, à travers l'intégration de mécanismes modulaires de coopération, communication et coordination, tenant compte des exigences spécifiques dans le domaine d'application pour la modélisation collaborative, l'interaction graphique et la communication multimédia.

Nous basant sur une application de modélisation géologique pionnière développée au départ par le Projet Gocad [GOC] suivant des principes de conception établis [GHJV95], nous présentons une architecture qui permet la création de sessions de collaboration exclusivement au moyen de la composition de modules d'extension (*plugins*). En plus de l'application principale et des modules d'extension fonctionnels, le schéma proposé implique l'intégration de :

- Un plugin de collaboration (NetGocad), responsable de la fourniture de services de gestion de session et du support de la coopération synchrone au moyen de la création de multiples "canaux de collaboration" parmi les instances distribuées de l'application, avec des mécanismes intégrés de diffusion (*broadcasting*) et de contrôle d'utilisation (*floor control*). En chargeant ce plugin, des versions standards mono-utilisateur de l'application peuvent être jointes à des sessions de collaboration.
- Un composant de coordination (LuaConference), implémenté dans un langage interprété simple et extensible, qui permet la séparation d'aspects de coordination du reste du système et leur spécification basée sur des types de conférences et des rôles de participants.
- Un composant de communication multimédia (CSVTool), fournissant des canaux vidéo et audio intégrés, soumis aux politiques de contrôle définies.
- Un plugin (Go2VR) de réalité virtuelle (RV) compatible avec la fonctionnalité de collaboration fournie, de sorte que dans certaines activités quelques tâches peuvent

être accomplies au moyen de l'utilisation de la RV, alors que d'autres sont réalisées synchroniquement au moyen de l'utilisation des interfaces d'ordinateur de bureau.

- Un plugin d'acquisition de données en temps réel (gWLog), utilisé pour l'intégration automatique dans le modèle de données recueillies dans des sites à distance.

Nous considérons un modèle de travail qui peut impliquer l'utilisation d'interfaces d'utilisateurs hétérogènes par les participants à distance et nous analysons l'application du système dans des cas d'usage opérationnels, basés sur des études de champ des activités clés à supporter.

Des chercheurs du Travail Coopératif Assisté par Ordinateur (TCAO; en anglais CSCW, *Computer Supported Cooperative Work*) ont remarqué qu'une collaboration à distance est aujourd'hui autant un défi social que technique [Ehr99, Gru92]. Un aspect fondamental pour la conception et le développement d'un système coopératif est l'analyse de processus de travail réels et de besoins associés de communication et de coordination [OO91].

Dans les géosciences en particulier, en comparant avec les secteurs d'application associés comme la CAO collaborative et la visualisation scientifique collaborative, la multidisciplinarité du travail et les problèmes de modélisation et de visualisation difficiles traités créent un riche domaine pour l'application et l'étude de la collaboration à distance, avec des exigences spécifiques qui ont besoin de se refléter dans l'architecture du système.

Pour l'analyse de ces questions, cette dissertation est organisée de la manière suivante. Au chapitre 1, nous fournissons une vue d'ensemble des activités du domaine d'application. Nous analysons les besoins et opportunités pour une collaboration à distance impliquant la visualisation et la modélisation, et less d'application considérés. Sur cette base, nous définissons les besoins pour notre solution. Au chapitre 2 nous analysons les concepts de collaboration à distance et leur rapport avec l'architecture proposée, qui est présentée au chapitre 3. Finalement, au chapitre 4 nous presentons des examples d'application du système.

Chapitre 1

Modélisation et visualisation collaboratives dans les géosciences

1.1 Introduction

Certaines des principales décisions techniques et d'affaires dans le secteur d'Exploration et Production ("E&P", ou "amont"; en anglais, "*upstream*") de l'industrie du pétrole et du gaz sont basées sur des modèles tridimensionnels de la subsurface. La création de ces modèles comprend une série de disciplines et activités, avec des problèmes et besoins particulièrement exigeants en termes de modélisation tridimensionnelle assistée par ordinateur, et est fortement appuyée par une visualisation de pointe.

De nombreux problèmes difficiles persistent dans ce domaine, en raison de plusieurs facteurs: la complexité géométrique et topologique des modèles géoscientifiques; les divers types d'objets graphiques employés; le besoin pour l'incorporation de concepts géologiques dans le processus de modélisation; le besoin pour l'intégration d'informations clairsemées, hétérogènes, incertaines et à multi-échelle de différentes disciplines; et la taille des modèles et ensembles de données, en termes de conditions informatiques. Un modèle géologique numérique évolue aussi continuellement, au moyen d'un effort d'équipe. Pendant l'exploitation et la production d'un champ de pétrole, de nouvelles données sont constamment acquises et de nouvelles analyses réalisées, ce qui fait que le modèle a besoin d'être fréquemment actualisé pour pouvoir appuyer les nouvelles décisions sur les mesures à prendre et sur les nouveaux investissements dans un projet. Une forte tendance dans l'industrie est l'acquisition en continu de données à partir d'endroits éloignés. Ces données ont besoin d'être reçues, analysées et incorporées dans le modèle en "temps réel" (relativement aux décisions engagées).

Une modélisation efficace dans les géosciences est fortement basée sur l'infographie. L'infographie à haute performance rend possible, pour les professionnels

impliqués dans le processus, la visualisation interactive et la mise en forme de grands modèles tridimensionnels intégrés: la visualisation est utilisée comme un outil puissant pour la compréhension et l'aperçu des données, comme un support pour une modélisation interactive et comme un langage commun pour la communication au sein d'équipes multidisciplinaires.

L'importance de la modélisation et de la visualisation tridimensionnelles a mené les compagnies de pétrole à adopter de plus en plus l'utilisation de diverses configurations de salles équipées d'écrans de projection immersifs pour favoriser la communication visuelle et la collaboration de groupe lors de sessions de travail techniques et de réunions de prises de décisions. Dans ce type d'environnement, la collaboration est nettement améliorée, si on la compare à l'utilisation d'écrans d'ordinateurs de bureau, principalement par le fait que les gens partagent le même espace physique, tout en ayant leur attention tournée vers les représentations de grande taille de leurs modèles, facilitant ainsi la communication de concepts et réduisant les malentendus. Chez Petrobras, par exemple, ceci se reflète par l'existence aujourd'hui de plus d'une douzaine de "salles de visualisation" à grands écrans, utilisées par des groupes de géosciences à différents endroits du Brésil, et dans des projets continus pour la construction de nouveaux systèmes d'immersion et pour leur intégration.

Habituellement, la plupart du temps ces environnements sont utilisés juste pour une collaboration en colloque avec des applications mono-utilisateur standard d'ordinateurs de bureau. Récemment, des applications de réalité virtuelle employant la visualisation stéréoscopique immersive et dispositifs d'interaction tridimensionnelle commencent aussi à être utilisées, avec un potentiel prometteur d'augmenter la compréhension et d'améliorer la précision dans certaines tâches de modélisation (par ex. dans la conception de puits horizontaux complexes, une opération stratégique et à coût élevé [Lei05]).

Etant donné la dispersion géographique des opérations et des professionnels et la pénurie d'experts en géomodélisation dans l'industrie, la collaboration à distance est maintenant reconnue comme technologie fondamentale pour améliorer la coopération distribuée et le transfert de connaissance dans diverses activités du flux de travaux E&P [EVD+02, TH02].

Aux paragraphes suivants, nous proposons une vue d'ensemble de ces activités, avec une emphase sur leurs besoins en termes de collaboration assistée par ordinateur.

1.2 Géomodèles

Avant de poursuivre, nous passons en revue rapidement la terminologie et les concepts utilisés pour décrire les objets de la modélisation collaborative dans les géosciences, d'un point de vue informatique.

Les modèles numériques tridimensionnels de la subsurface représentant des objets géophysiques, géologiques ou réservoirs sont appelés "géomodèles"ou "modèles de la terre" (*earth models*). La création d'outils et de méthodes pour la conception assistée

par ordinateur de ces objets et de leurs propriétés est le sujet de la "géomodélisation" [Mal02], et implique une grande gamme de compétences, englobant différents domaines: science informatique, mathématiques, géologie, géophysique et ingénierie des réservoirs.

Les objets mathématiques utilisés en géomodélisation sont hétérogènes, ce qui rend cette discipline particulièrement motivante. Les modèles peuvent être représentés par des points, lignes, surfaces, mailles, volumes, peuvent avoir des attributs multi-échelle et multi-dimensionnels associés et peuvent être statiques ou dynamiques. Ils sont souvent très grands, dans le sens où leur représentation discrète est difficile pour la capacité de traitement et de stockage d'ordinateurs haut de gamme.

Dans ce travail, notre principal intérêt est l'utilisation de géomodèles à partir d'une perspective infographique. Dans ce sens, les géomodèles peuvent aussi être vus comme des "objets graphiques" [GCDV96], un concept qui comprend tous les objets manipulés dans un système infographique sous une définition mathématique consistante, en termes de forme (topologie et géométrie), attributs (par ex. propriétés physiques), descriptions et représentations.

Le Projet Gocad proposa une implémentation pionnière d'un géomodèle tridimensionnel intégré, basé sur des objets graphiques avec topologie, géométrie et propriétés discrètes et un ensemble de contraintes pour représenter les relations d'interdépendance entre différents objets et leurs propriétés (ainsi qu'une méthode d'interpolation spécifique qui tient compte de ces relations [Mal89, Mal92]). La formulation de contraintes géologiques a été l'objet d'une recherche active dans le groupe [CSM03]. Comme nous en parlerons plus tard, ces interdépendances apportent des difficultés considérables au problème de contrôle de concurrence dans la modélisation collaborative, puisqu'elles compliquent l'annulation et la reprise des opérations.

Dans notre travail, un modèle Gocad est utilisé comme le géomodèle partagé, autour duquel une collaboration synchrone prend place. Pour une interopérabilité avec d'autres applications, l'environnement cadre OpenSpirit Component Framework [OS], conçu et développé par l' OpenSpirit Alliance, peut être utilisé. OpenSpirit est une plateforme de logiciel indépendante d'application visant à valider l'intégration "plug-and-play" des applications E&P, qui agit comme un intergiciel (*middleware*) permettant l'accès à des stocks de données de tiers. Elle permet aussi un certain niveau d'interopération synchrone parmi différentes applications à travers l'utilisation de quelques événements prédéfinis (suivi du curseur, point d'intérêt, sélection d'objet, notification de changement de propriétés et de données).

Le terme "Shared Earth Model" (SEM) est aussi utilisé dans un contexte plus général dans l'industrie du pétrole et du gaz pour décrire un modèle unifié pour toutes les activités de géosciences dans le flux de travaux des activités d'exploitation et de production.

1.3 Flux des travaux en amont

Le schéma suivant (Figure 1.1) montre une vue simplifiée des principaux stades impliqués dans la création d'un modèle pour l'exploitation et la production de pétrole et de gaz, à partir de l'acquisition de données sismiques jusqu'à l'exploitation d'un réservoir.

ACQUISITION ET TRAITEMENT SISMIQUE
↓
INTERPRETATION
↓
MODÉLISATION GEOLOGIQUE
↓
MODÉLISATION DE RESERVOIR
↓
CARACTÉRISATION DU RESÉRVOIR
↓
SIMULATION D'ÉCOULEMENT
↓
CONCEPTION DE PUITS
↓
PLANIFICATION D'INSTALLATIONS
↓
PRODUCTION

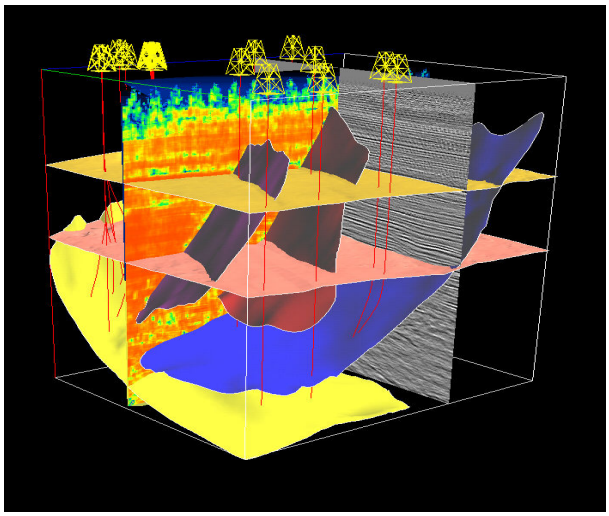


FIG.1.1 – Flux de travaux simplifié de géomodélisation.

En fait, le flux de travaux n'est pas linéaire - des retours en arrière peuvent arriver à partir de n'importe quelle étape, puisque des révisions du modèle sont exigées pour intégrer de nouvelles informations. Les premiers stades consistent en l'acquisition et le traitement de données sismiques par les géophysiciens. D'un point de vue informatique, les problèmes principaux, à cette phase, sont les tailles gigantesques des ensembles de données (qui peuvent atteindre des Terabytes) et le temps de traitement demandé. La plus grande partie de cette activité est fondée sur un traitement des données par de super ordinateurs et grappes (actuellement avec jusqu'à quelques milliers de noeuds).

Ensuite, les données sismiques sont "interprétées" (segmentées) par des géophysiciens et des géologues pour l'identification des structures internes de la subsurface. Les régions d'intérêt modelées varient selon la situation, étant plus grandes pour des études régionales et plus restreintes pour des études de réservoir. L'ensemble des données sismiques manipulé à cette phase peut facilement occuper toute la mémoire principale des ordinateurs de bureau haut de gamme (quelques Gigabytes), spécialement lorsque de multiples attributs sismiques sont saisis pour le même volume (actuellement une situation habituelle). Des ordinateurs centraux avec une grande mémoire principale (jusqu'à cent Gigabytes) dédiés à la visualisation haute performance sont utilisés pour permettre un travail interactif avec la plus grande quantité possible de données.

Au stade de modélisation géologique, les géophysiciens et les géologues représentent d'abord la configuration de couches de roches et blocs en modelant les surfaces qui les délimitent. Ces surfaces consistent principalement en deux types géologiques: *horizons* et *failles* (Figure 1.2, paragraphe 1.4.1). Un horizon représente l'interface entre deux différentes couches de roches, composée de sédiments superposés le long du temps géologique. Une faille correspond à la fracture de quelques unes de ces couches conduisant au déplacement relatif de blocs en raison des efforts de rupture, compression ou extension. Sous l'action de forces géologiques, les couches peuvent avoir été déformées et peuvent avoir assumé des configurations géométriques complexes. Les surfaces d'horizon et de faille sont modelées à travers l'interpolation de types hétérogènes de données (points, lignes, parties de surface) obtenues au stade d'interprétation et à partir de l'analyse des puits présents dans la zone.

La modélisation précise des surfaces, et spécialement des intersections entre elles, demande l'utilisation de méthodes de modélisation de pointe. Une question fondamentale à ce stade est le fait que toutes les informations disponibles ont besoin d'être honorées, avec différents niveaux de fiabilité. Une intense collaboration parmi les géologues et les géophysiciens est cruciale pour l'accomplissement d'un modèle qui représente le mieux la connaissance existante concernant le champ.

Ensuite, au stade de modélisation et de caractérisation de réservoir, une grille volumétrique est définie pour un bloc (région fermée) du modèle identifié comme un réservoir (Figure 1.3, paragraphe 1.4.1). Cette grille est utilisée par les ingénieurs de réservoir comme un support pour la saisie des propriétés pétrophysiques et ensuite pour

les études de simulation de réservoir, tenant compte de la localisation et la géométrie des puits utilisés pour exploiter le réservoir.

La conception et le placement du puits constituent des décisions très importantes dans le flux de travaux E&P, étant donné que le coût d'un puits est très élevé, et sa localisation a un impact économique fondamental sur la production et sur la définition des installations utilisées dans un champ.

La conception d'un puits est réalisée à travers une collaboration directe entre les géophysiciens, les géologues et les ingénieurs de réservoir et de forage, et est une activité critique à supporter par un système coopératif. Pendant l'opération de forage, le progrès de la trajectoire doit être surveillé, pour que, selon l'accord ou non des informations obtenues du bout du puits avec ce qui était attendu du modèle, il soit guidé en temps réel (voir Figure 1.4, paragraphe 1.4.2).

Un problème crucial dans ce processus de travail est le besoin de l'intégration des informations venant des différents domaines et de la redéfinition coopérative du modèle, dès que de nouvelles informations sont obtenues. Ceci a conduit des fournisseurs de logiciel à la fusion d'applications indépendantes traditionnelles en outils intégrés. Cependant, comme conséquence, un seul système fournissant toutes les fonctionnalités combinées nécessaires dans toutes les disciplines souvent finit par devenir très complexe. La solution adoptée dans certains systèmes, comme Gocad, est l'adaptation de l'application à travers l'utilisation de plugins.

1.4 Activités coopératives

Lors du flux de travaux décrit ci-dessus, certaines activités plus que d'autres impliquent une étroite coopération entre des spécialistes à différents endroits. Nous en considérons deux en particulier, la modélisation géologique et la planification de puits, qui sont les plus représentatives des conditions, bénéfices et possibilités apportés par une collaboration à distance

Ci-dessous nous analysons davantage ces activités. Ensuite, au paragraphe 1.6, elles sont représentées dans des cas d'usage simplifiés utilisés pour l'identification de conditions de collaboration, et aussi comme cas de test.

1.4.1 Modélisation géologique

La construction d'un modèle géologique est une tâche difficile. Les données utilisées comme entrée pour le processus de modélisation sont éparses et comportent un degré considérable d'incertitude, menant à différentes interprétations possibles. La construction d'un modèle demande une connaissance géologique sur les prospections, maîtrisée par peu de spécialistes. Elle dépend des informations générées lors des activités précédentes dans le flux de travaux, et est cruciale pour les suivantes. Une collaboration entre les géophysiciens et les géologues est essentielle dans cette phase,

par exemple dans des situations telles la définition de l'extension de failles, ou leur configuration d'intersection dans des réseaux de failles complexes (Figure 1.2).

Actuellement, chaque fois que des discussions et de la collaboration sont nécessaires, quelques professionnels se réunissent autour de l'ordinateur de bureau ou dans des salles de visualisation équipées de grands écrans, s'ils travaillent au même endroit; sinon ils discutent par téléphone ou doivent voyager souvent pour une coopération face à face. Des outils de partage d'application sont quelquefois utilisés, mais avec des limitations (comme analysé plus loin au paragraphe 2.3.1).

Les outils de modélisation tridimensionnels sont relativement nouveaux dans l'industrie, pas simples à utiliser et en constante évolution. Ainsi les non-spécialistes ont besoin continuellement de cours et de formation. Des experts disponibles sont souvent obligés de donner des conseils à distance en utilisant des moyens de communication insatisfaisants, comme les conversations par téléphone et des échanges de fichiers. Des outils plus efficaces pour la consultation à distance avec des experts en modélisation sont fortement souhaitables.

Dans les cas d'usage décrits aux paragraphes 1.6.1 et 1.6.2 nous décrivons des sessions de modélisation synchrone distribuées avec communication intégrée pour consultation ou formation à distance.

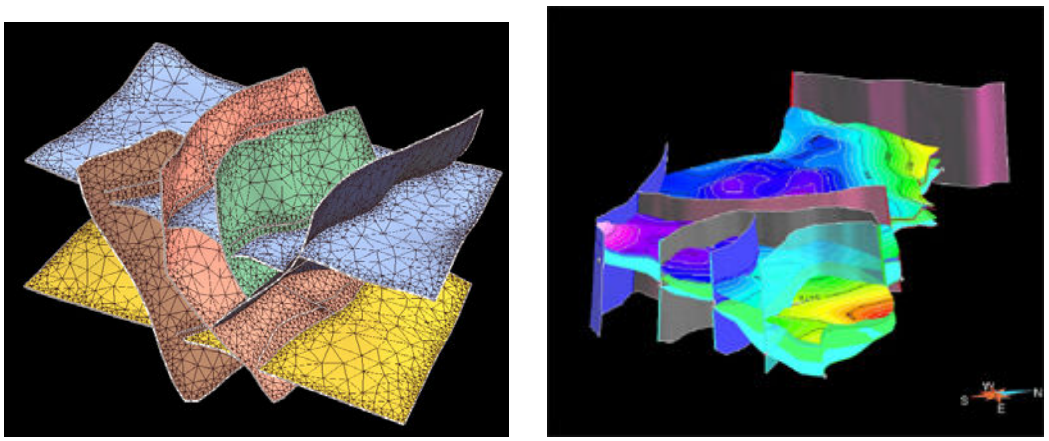


FIG. 1.2 – Modèles structuraux. Des modèles complexes peuvent avoir des douzaines (ou même des centaines) de failles et d'horizons, avec des configurations compliquées.

Modélisation de réservoir

Lorsque le modèle structural d'un réservoir étudié est simple, contenant basiquement des horizons plats et des failles verticales, la construction des grilles utilisées pour la caractérisation et la simulation du réservoir est relativement simple. Cependant, dans des configurations structurales plus complexes, ceci devient une étape difficile dans le flux de travaux de modélisation.

La visualisation est critique pour l'inspection de la qualité de la grille et des résultats d'interpolation des propriétés et des simulations (Figure 1.3). Les ingénieurs de réservoir veulent souvent discuter avec des pairs ou avec le géologue responsable de la construction du modèle structural. Comme lors de la phase de modélisation géologique, les spécialistes se réunissent autour de l'ordinateur de bureau ou d'un grand écran pour résoudre un problème.

Les dynamiques et les conditions de cette collaboration sont similaires à celles de l'étape de modélisation structurale, ainsi que les besoins de consultation et de formation avec des experts à distance. Par conséquent, les cas d'usage 1.6.1 et 1.6.2 sont également applicables à cette activité.

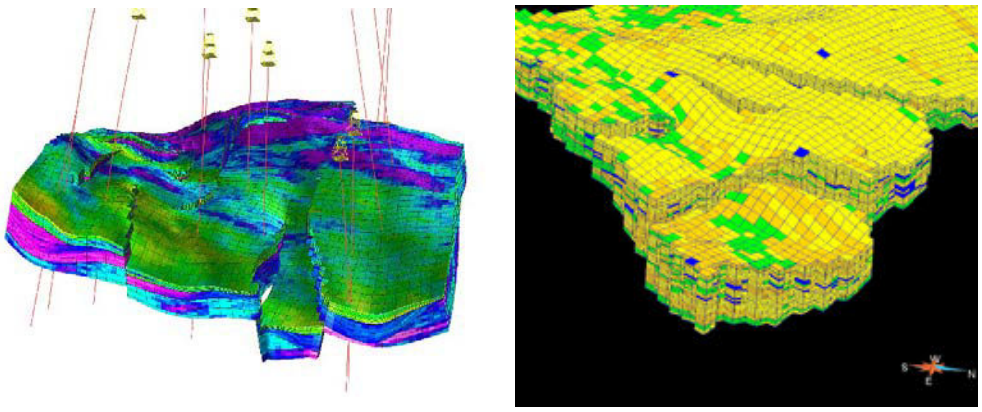


FIG 1.3 – Modèles de réservoir. Des grilles volumétriques conformément aux modèles structuraux contiennent des propriétés pétrophysiques calculées selon les données disponibles, et les résultats des simulations de réservoir.

1.4.2 Planification et guidage des puits

La conception et le guidage du puits en temps réel sont les applications-clés du présent travail. Ces activités impliquent la visualisation intégrée et l'analyse de tous les types d'informations (données sismiques, surfaces géologiques, grilles de réservoir, trajectoire du puits et diagraphie des propriétés) par une équipe multidisciplinaire de spécialistes, travaillant souvent à différents endroits.

Le coût élevé d'un puits (jusqu'à quelques douzaines de millions de dollars dans un endroit profond de la mer) rend toute amélioration en précision et productivité dans ces activités très rentable, considérant que des douzaines de puits peuvent être utilisés dans un seul champ. Pour minimiser le nombre de puits et optimiser la production de certains types de réservoirs, les compagnies emploient souvent des puits "directionnels" spéciaux. Dans ce cas, des tronçons de puits non-verticaux peuvent avoir de plusieurs centaines à quelques milliers de mètres, et ont besoin d'être placés précisément le long des couches de réservoir (qui parfois n'ont que quelques mètres d'épaisseur), ce qui demande une collaboration étroite entre les spécialistes pour la planification et le géoguidage du puits.

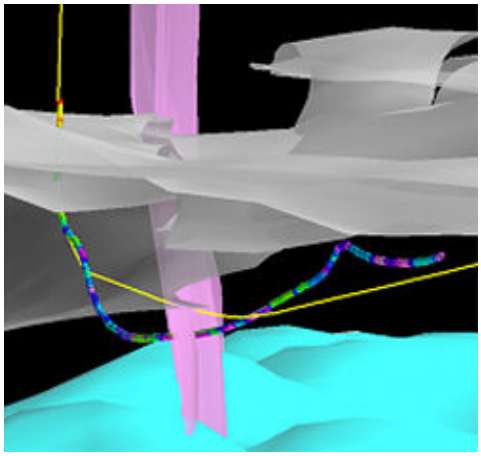


FIG. 1.4 – Puits planifié (jaune) et réel pendant (coloré) le forage (image d'une opération réelle de géoguidage dans laquelle le système a été appliqué).

Pendant l'opération de forage, les diagraphies du puits (mesures des propriétés pétrophysiques des formations de roches perforées) sont enregistrées et envoyées à la surface en temps réel. Ces mesures permettent aux géoscientistes d'identifier les types de roches perforées, et de vérifier si elles coïncident avec le modèle géologique actuel.

En raison des incertitudes inhérentes impliquées dans la définition du modèle, ce n'est souvent pas le cas (Figure 1.4). Habituellement, deux types principaux de problèmes peuvent surgir: le réservoir peut ne pas être trouvé à la position attendue, ou une fois à l'intérieur, le foret peut subitement lâcher, traversant l'horizon supérieur ou l'inférieur.

Si un problème arrive et un désaccord est détecté entre le modèle et la configuration de subsurface réelle, des décisions doivent être prises rapidement concernant la manière d'ajuster la direction de forage pour pouvoir maintenir la trajectoire du puits dans la meilleure position à l'intérieur du réservoir - une décision délicate en temps, étant donné que le forage ne peut pas être stoppé. Simultanément, le modèle doit être corrigé pour coïncider avec la réalité observée.

Il y a encore peu de temps, la communication entre la plateforme pétrolière et le bureau était basiquement faite par téléphone avec des échanges de fichiers décrivant l'évolution de la trajectoire du puits. De nos jours, quelques systèmes commencent à offrir des fonctionnalités pour une visualisation synchrone du modèle 3D par deux partenaires à distance (paragraphe 1.5). Cependant, beaucoup de caractéristiques sont encore nécessaires, comme analysé au paragraphe 1.7.

La qualité des connexions du réseau aux sites de forage varie beaucoup de site en site. Alors que les tours de forage ou les plateformes pétrolières proches de la côte peuvent avoir des connexions à haut débit avec le bureau, les plateformes distantes n'ont souvent que des connexions par satellite avec faible débit et temps de latence élevé.

1.4.3 Réalité Virtuelle

Les limitations de l'interaction basée sur la souris pour réaliser des opérations de modélisation tridimensionnelles complexes ont mené à un intérêt pour l'utilisation de dispositifs avec de plus hauts degrés de liberté. Cela a été longtemps proposé dans les domaines d'infographie et réalité virtuelle (RV; en anglais VR, pour *virtual reality*), et a été récemment apporté aux situations opérationnelles dans les géosciences.

Une possibilité est l'utilisation des applications de réalité virtuelle dans des environnements immersifs, avec des dispositifs d'interaction à six degrés de liberté, visualisation stéréo et suivi de tête (*head tracking*). Une autre alternative prometteuse est l'utilisation de systèmes haptiques. Cependant, ces interfaces sont conçues pour des utilisateurs individuels, et chacune d'elles convient mieux à des types spécifiques de tâches. Un modèle de travail intéressant et plutôt non exploité est l'utilisation en sessions de collaboration hétérogènes dans lesquelles les participants peuvent utiliser différents types d'interfaces d'utilisateur selon leurs tâches spécifiques.

Par exemple, un participant utilisant un système haptique pourrait être responsable de tâches d'édition localisées, associé à un autre participant utilisant une interface d'ordinateur de bureau, responsable de tâches demandant une vue globale du modèle et plus de manipulations d'interfaces basées sur le menu, et encore un troisième dans un environnement immersif avec une vue à échelle humaine du modèle résultant. Nous

attendons qu'à l'avenir ce type de collaboration puisse mener à une meilleure efficacité générale que lorsque juste un type d'interface d'utilisateur est utilisé par tous les participants dans une session.

Planification de puits

L'utilisation des interfaces de réalité virtuelle montre un potentiel considérable pour des améliorations de performance en conception de puits, facilitant la sélection de cibles et la compréhension de relations spatiales complexes dans la région du puits [Gru04, Lei05]. Cela permet à l'utilisateur de pénétrer "à l'intérieur" du modèle, à une échelle appropriée, en utilisant une visualisation stéréo et suivi de la tête pour analyser les endroits spécifiques en bougeant intuitivement autour d'eux.

Récemment une application RV commerciale, Inside Reality [IR], a été opérationnellement utilisée dans certaines compagnies pour la planification de puits; dans le cadre de ce travail elle a été utilisée dans des séances d'évaluation de la fonctionnalité RV avec des professionnels en géosciences.

Une autre application RV, Immersive Drilling Planner (IDP), a été développée initialement comme un plugin Gocad [IDPa], mais actuellement n'est pas disponible commercialement. Une partie de sa fonctionnalité a été ensuite transférée vers un plugin régulier de planification de puits pour Gocad [IDPb].

A travers les mécanismes analysés dans ce travail (chapitre 3), Go2VR peut être utilisé de façon collaborative, en combinaison avec d'autres plugins Gocad, y compris IDP, pour l'exécution de sessions de collaboration hétérogènes comprenant la participation d'utilisateurs de RV.

Modélisation géologique

Tandis que pour des modifications locales du modèle dans des tâches comme la conception de puits, comme analysé ci-dessus, les interfaces de réalité virtuelle peuvent être très efficaces, pour d'autres types d'opérations de modélisation comme l'édition globale de modèles géologiques elles présentent certains points faibles, en raison des fonctionnalités restreintes normalement fournis par l'interface d'immersion, de la précision de pointage plus faible et de la fatigue causée par l'utilisation de dispositifs d'interaction 3D pendant de longues sessions de travail. Cependant, l'interface d'immersion peut améliorer la compréhension de modèles géologiques complexes en permettant aux utilisateurs de naviguer librement, comme s'ils visitaient l'endroit réel.

Haptique

Les dispositifs d'affichage haptique ont été récemment apportés aux géosciences pour permettre la performance de quelques tâches de modélisation d'une manière très intuitive, avec l'aide du "retour d'effort" (*force feedback*) [Har04].

Nous avons testé deux applications. Reachin GeoEditor est une application haptique de modélisation de surface pour le système d'affichage Reachin [RGE]. Les utilisateurs peuvent réaliser plusieurs opérations de modélisation, avec une sensation très réaliste de toucher les surfaces (horizons ou failles), ce qui facilite énormément certaines tâches

difficiles comme les corrections locales de mailles détaillées. Comme décrit au paragraphe 3.5.4, l'application est lancée à travers un plugin Gocad, permettant l'échange d'objets entre l'instance Gocad conventionnelle et l'interface haptique, en utilisant la version préliminaire du protocole NetGocad.

Une autre application pour le système Reachin est le Well Path Planner développé par Norsk Hydro en partenariat avec Reachin Technologies, qui permet la conception de la trajectoire d'un puits et la manipulation de cibles en 3D, sujet à l'application de force proportionnelle aux propriétés mécaniques du puits.

Pour des raisons d'utilisabilité, les deux applications offrent une gamme tout à fait limitée et spécialisée d'opérations de modélisation à travers leurs interfaces haptiques. Nous croyons que leur intégration directe dans des sessions collaboratives hétérogènes permettrait un travail beaucoup plus efficace.

1.5 Systèmes de géosciences associés

Parmi les systèmes commerciaux de modélisation en géosciences disponibles aujourd'hui, très peu (à notre connaissance) offrent un certain niveau de capacité de collaboration synchrone tridimensionnelle à distance. Deux exemples sont EarthVision et Inside Reality. (Au paragraphe 2.6 nous analysons l'utilisation des systèmes coopératifs dans des domaines d'application associés.)

EarthVision [DGI] est un logiciel de modélisation et visualisation 3D développé par Dynamic Graphics Inc. La société propose un visionneur à distance qui permet la visualisation synchronisée de modèles 3D à de sites éloignés. Une instance du visionneur peut être connecté à une instance de l'application ordinaire, chacune avec une réplique locale du modèle, et les événements de changement de point de vue peuvent ensuite être échangés pour la synchronisation des vues. Le logiciel permet des connexions, une par une, entre le bureau et un site de forage de puits pour la prise de décision pendant les opérations. Cependant, il ne permet qu'une visualisation synchronisée, et ne supporte pas une coopération plus sophistiquée ou la communication intégrée.

Inside Reality [IR], développée par Schlumberger (et discuté au paragraphe précédent), est actuellement la seule application commerciale de réalité virtuelle pour les géosciences. Un utilisateur actif utilise le suivi de la tête et interagit avec le système avec une baguette 3D. Le systéme permet une collaboration à entre des utilisateurs RV à distance, représentés comme des avatars géométriques (mais cette fonctionnalité n'a pas été utilisée dans nos évaluations, en raison de la non disponibilité des ressources nécessaires).

1.6 Cas d'usage pour la collaboration à distance

Les cas d'usage suivants résument les situations opérationnelles cibles que nous considérons dans ce travail, qui définissent les conditions pour le modèle de collaboration proposé (paragraphe 1.7). Ils ont été établis suite à l'étude des activités décrites aux paragraphes précédents, et après discussions avec des spécialistes sur les fonctionnalités attendues des systèmes coopératifs. Plus tard ils seront utilisés pour la définition d'un ensemble initial de politiques de collaboration pour le système développé, et pour les tests (chapitre 4).

1.6.1 Consultation et modélisation collaborative

L'utilisateur *A* construit un modèle et se trouve face à un problème concernant la réalisation d'une certaine tâche (par ex., l'édition d'une surface géologique). *A* décide alors de demander une aide à *B*. Il crée une session de collaboration (une conférence), invite *B* à s'y joindre à partir d'un site à distance, transfère le modèle et explique le problème, tournant autour du modèle, zoomant la zone du problème, pointant les choses et faisant des annotations que *B* peut voir d'une façon synchronisée, comme si les deux regardaient le même écran. *B* peut alors manipuler la caméra et faire des annotations directement sur le modèle tridimensionnel, qui est vu par *A*. Pour les deux participants seulement, une conversation peut être établie par téléphone. Des canaux intégrés de visioconférence, automatiquement établis parmi les participants d'une conférence, simplifient énormément et améliorent la communication. Une fois qu'une session collaborative est créée, l'audio et la vidéo peuvent être utilisées à tout moment, sans le besoin de connexions externes (ce qui serait le cas si une application de visioconférence indépendante était utilisée).

En principe, des télépointeurs individuels peuvent être utilisés tout le temps, ainsi que des annotations graphiques. La couleur du curseur choisie par chaque participant est indiquée dans l'interface. Les mouvements de la caméra peuvent être négociés ou non. Pour une collaboration un à un, un contrôle non négocié permet plus d'interaction et de communication informelles. Des mouvements en conflit sont immédiatement détectés, et une négociation directe pour le contrôle peut être faite pendant la discussion par le canal audio, sans le besoin d'un passage de contrôle explicite. Cependant, si plus de deux participants sont impliqués, la négociation du contrôle à travers un protocole verbal devient compliquée, interférant avec la discussion principale, alors un mécanisme de tour de rôle devient nécessaire. Dans ce cas, l'interface devrait montrer clairement qui a le contrôle (le *floor*) à un certain moment, et permettre un passage de contrôle intuitif.

Pour une édition simultanée, une stratégie de contrôle de concurrence doit être utilisée, sinon le modèle peut devenir inconsistant entre les sites à distance en raison des opérations conflictuelles.

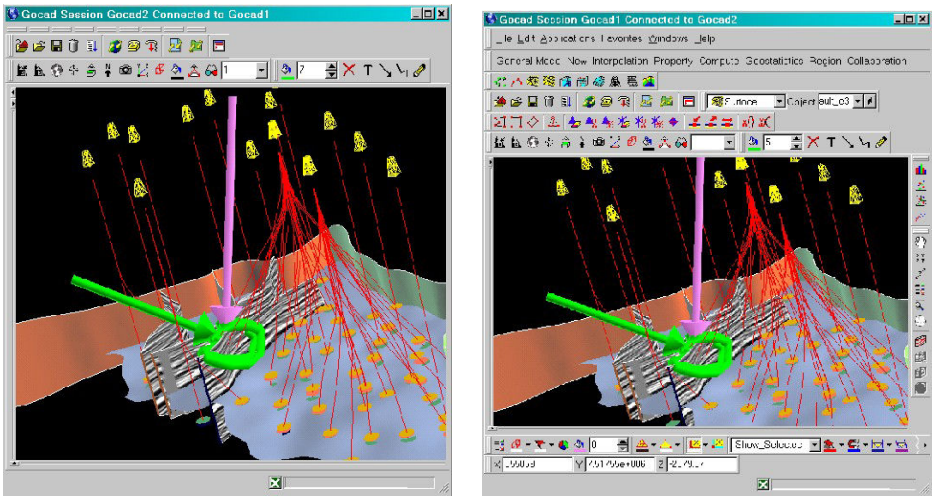


FIG 1.5 – Session de collaboration avec le système développé, dans une situation équivalente à celle décrite dans le cas d'usage 1.6.1.

1.6.2 Formation

Pour une formation à distance, plusieurs participants doivent être impliqués dans une session de groupe. Dans ce cas, l'instructeur a besoin d'agir comme un coordinateur de la session, avec le droit de passer le contrôle des ressources de collaboration (télépointeurs, contrôle de caméra, droits d'édition, etc.) parmi les participants, et de le reprendre à tout moment.

Un aspect fondamental de la formation est que l'instructeur doit être capable de montrer aux autres participants comment accomplir une certaine tâche. Pour cela, des manipulations d'interface doivent être visibles à distance, et non seulement le résultat des opérations. La vidéo peut être utilisée non seulement pour la communication, mais aussi comme un moyen d'affichage à distance de l'opération d'interface. Avec des outils réguliers de visioconférence, ceci pourrait être fait simplement en pointant la caméra vers l'écran, mais avec une qualité faible; alternativement, un outil de partage d'écran pourrait être utilisé.

Des flux multiples audio et vidéo doivent être permis entre les participants. Cependant, comme la moyenne des ordinateurs de bureau ne peut pas supporter l'utilisation simultanée de beaucoup de flux, et aussi à cause d'éventuelles restrictions de largeur de bande du réseau, l'outil de visioconférence ou de partage d'écran devrait permettre le contrôle sur les connexions individuelles, avec un mécanisme adapté géré par le contrôleur de conférence (l'instructeur).

Habituellement, pour des sessions de formation, initialement uniquement des flux audio bidirectionnels et vidéo unidirectionnels devraient être créées entre l'instructeur et chaque participant. Si quelqu'un a un doute, l'instructeur peut permettre la création d'un flux vidéo dans sa direction, pour qu'il/elle puisse voir l'interface de l'apprenti. Des flux peuvent aussi être créés temporairement pour montrer le problème aux autres apprentis.

L'utilisation d'un programme de visioconférence d'usage courant ne permettrait pas ce type de contrôle. L'intégration d'un outil de visioconférence avec l'application coopérative est importante pour permettre le contrôle sur des flux individuels. Elle permet aussi l'intégration de services de gestion de session et une identification consistante des participants.

1.6.3 Planification et guidage de puits

Comme analysé précédemment, de très importants cas d'usage pour une collaboration à distance sont la conception et le guidage en temps réel de puits (géoguidage). Les dynamiques de travail et les conditions de la phase de planification de puits sont équivalentes à celles de la modélisation collaborative, sauf que des professionnels de spécialités différentes sont impliqués: des géophysiciens, des géologues et des ingénieurs de puits. Ils peuvent être dispersés entre un bureau régional et le siège de la société. Généralement, un professionnel de chaque spécialité est impliqué dans un projet, et ils travaillent ensemble ou par deux. Des spécialistes supplémentaires peuvent se joindre au groupe pour une discussion, ainsi que le chef d'équipe. Ceci souligne le besoin d'ajustement de l'interface de l'utilisateur avec des configurations hétérogènes aux différents sites, puisque les participants utilisent les fonctionnalités spécifiques à leurs disciplines.

Pendant la phase de forage de puits, les données acquises au bout du puits dans la subsurface sont reçues au bureau à partir de la localisation du forage à distance et sont affichées sur la caméra 3D, pour une comparaison avec le puits planifié et le modèle géologique, partagées par les géoscientistes locaux et éloignés.

Cette activité implique au moins deux sites : le bureau local où un géoscientiste ou plus, responsables du plan du puits, suivent l'opération (chacun utilisant une configuration d'interface différente convenant à sa spécialité), et le site de forage, où l'opérateur utilise un ordinateur moins puissant et une version moins lourde de l'application pour visualiser le modèle partagé, mais avec des droits d'édition restreints. Les professionnels du bureau central peuvent aussi se joindre à la discussion.

Fréquemment une largeur de bande de réseau très limitée est disponible sur le site de forage, demandant un contrôle strict de l'utilisation de la communication audio et vidéo.

1.6.4 Géoguidage avec réalité virtuelle

Dans ce cas d'usage une interface RV est utilisée comme partie d'une session de collaboration hétérogène. Un participant à la session utilise le système RV dans un

environnement virtuel immersif, collaborant avec d'autres utilisateurs d'ordinateurs de bureau. Parmi d'autres questions d'utilisabilité, des événements de caméra ne peuvent pas être échangés entre les utilisateurs d'ordinateurs de bureau et de RV. Une telle collaboration implique :

- L'opérateur, sur le site de forage, suivant la progression de l'opération en utilisant une version simplifiée de l'application (un visionneur 3D).
- Un géoscientiste travaillant dans un environnement d'immersion, utilisant l'interface de réalité virtuelle, avec la responsabilité de la conception de la trajectoire du puits. L'immersion, la visualisation stéréo et la proximité du modèle, rendu à l'échelle humaine, permettent une meilleure perception de relations spatiales et par conséquent une analyse plus efficace de configurations géométriques complexes pour le puits et les objets environnants (surfaces géologiques, données du puits, grilles sismiques et du réservoir).
- Un géoscientiste non immergé, utilisant une version d'ordinateur de bureau de l'application, responsable de tâches de modélisation globales et d'autres manipulations d'interfaces basées sur le menu. La coopération avec l'utilisateur immergé suit un protocole de collaboration: tous les participants peuvent pointer les problèmes avec des télépointeurs et rajouter des annotations 3D, mais l'utilisateur RV ne doit pas recevoir d'événements de manipulation de caméra des participants d'ordinateur de bureau, étant donné qu'il a un point de vue avec suivi automatique (sinon il serait complètement désorienté et aurait une perspective faussée de l'environnement immersif). Dans l'autre direction, les utilisateurs d'ordinateur de bureau peuvent recevoir des événements de caméra de l'utilisateur RV, pour qu'ils puissent observer la scène du même point de vue. La position de la caméra de l'utilisateur RV pourrait aussi être diffusée comme un avatar pour être rendue dans les vues non synchronisées des participants à distance. En fait toutes les positions suivies des dispositifs d'interaction devraient aussi être diffusées et utilisées dans le rendu d'avatars. La communication devrait être faite d'une manière intégrée (pour que, par exemple, l'audio puisse être capturée et reproduit par des microphones et des haut-parleurs placés convenablement).

1.7 Besoins et conséquences

Au long de plusieurs mois nous avons observé, discuté et pris part aux séances de travail opérationnel concernant les activités décrites. Les groupes que nous avons suivi ont utilisé des applications de fournisseurs multiples, en déroulements d'opérations similaires. Aucun d'entre eux n'a utilisé de systèmes de collaboration à distance, sauf pour certaines utilisations expérimentales d'outils de partage d'application (voir paragraphe 2.3.1.1). Des applications de réalité virtuelle ont été utilisées expérimentalement. À partir de ces observations nous avons dessiné quelques besoins spécifiques pour l'architecture proposée, tant du point de vue des utilisateurs que des développeurs du système, qui ont guidé nos décisions de conception. Dans ce

paragraphe nous commentons brièvement quelques conséquences de ces conditions, qui seront analysées plus loin aux chapitres 2 et 3.

1.7.1 Petits groupes

Toutes les activités considérées impliquent un petit nombre de participants (pour une consultation normalement deux, parfois trois; de deux à quatre pour une planification de puits; un peu plus pour une formation). Par conséquent, le passage à l'échelle (*scalability*) n'est pas une question dans ce domaine. C'est une des caractéristiques des systèmes coopératifs scientifiques et d'ingénierie, à l'opposé d'autres types d'applications avec un plus grand nombre d'utilisateurs (jusqu'à des centaines ou même des milliers), comme les jeux à joueurs multiples, des simulations militaires ou d'autres types d'environnements virtuels de réseaux à grande échelle, dans lesquels le passage à l'échelle est une question centrale, avec des implications architecturales importantes [SZ99, Gre99]. Dans notre solution, ceci se reflète dans le choix que nous avons fait d'une architecture hybride, avec la réplication de l'application et des données, mais une diffusion et un schéma de coordination centralisés.

1.7.2 Haute performance graphique

La modélisation et la visualisation tridimensionnelles demandent une interaction très réactive. Pour un travail efficace, les fréquences de trame devraient être d'au moins 10 trames/seconde, mais des fréquences plus rapides sont fortement souhaitables pour une précision et un confort accrus [Nie94]. La réactivité a besoin d'être maintenue avec des réseaux d'enterprise standards locaux et globaux (même avec un trafic concurrent). Par conséquent, une architecture répliquée a besoin d'être utilisée, par opposition à des schémas centralisés, pour que le rendu puisse être réalisé localement, et la synchronisation d'état puisse être faite à travers la diffusion des commandes et les événements d'interaction, en utilisant un réseau à faible débit (paragraphe 2.4.1). D'ailleurs, des méthodes de diffusion et des protocoles réseaux appropriés doivent être considérés, spécialement pour des événements d'interaction graphique.

1.7.3 Communication multimédia intégrée

Les communications audio et vidéo sont des composants fondamentaux des systèmes de modélisation collaborative. L'audio est un canal essentiel pour supporter le travail synchrone [SWC76], et la vidéo est importante pour fournir un sens de co-présence et pour faciliter les tâches de négociation [OO95, IT94].

Dans certains systèmes, ceci est fourni à travers des outils externes, demandant une gestion et des contrôles de sessions séparés. Nous considérons que l'audio et la vidéo doivent être fournis comme canaux de communication intégrés, pour permettre une initiation de session plus simple et plus efficace, et aussi l'utilisation de canaux multiples directement contrôlés à travers l'application principale. La vidéo ne devrait

pas servir seulement à la communication directe mais aussi à l'affichage à distance de l'interface de l'utilisateur et à l'observation du travail des pairs.

Comme l'utilisation de flux multiples audio et vidéo pose de fortes conditions sur la consommation de bande passante et traitement, les participants à la conférence doivent être capables de contrôler les flux individuels, avec des droits soumis aux politiques basés sur les rôles pour le type de conférence mis en place.

1.7.4 Réutilisation de fonctionnalité mono-utilisateur

Le déploiement de l'application est l'un des aspects les plus difficiles du développement collecticiel [Ehr99]. Nos applications cibles demandent l'intégration d'un grand ensemble d'outils et de fonctionnalités de modélisation multidisciplinaires, habituellement développés depuis de nombreuses années. Par conséquent, une exigence principale pour l'architecture proposée est de permettre non seulement la réutilisation du code des applications de base, mais l'extension directe d'une application mono-utilisateur existante dans un système coopératif, à travers l'incorporation de composants chargés dynamiquement, sans recompilation et redistribution.

1.7.5 Contrôle de concurrence

Comme analysé au paragraphe 1.2, la modélisation géologique implique la construction de divers types d'objets mathématiques (surfaces, mailles, modèles solides) avec une géométrie et une topologie complexes, des relations d'interdépendance, et un très grand nombre d'éléments (une seule version de certains objets peut approcher aisément la taille de mémoire disponible). Actuellement, il n'est pas possible d'assumer que les opérations de modélisation peuvent toujours être défaites efficacement. Dans ce contexte, une gestion de concurrence efficace avec une réactivité élevée est encore un défi (comme analysé au paragraphe 2.5.3).

Par conséquent, l'exigence de base adoptée à présent pour un maintien de consistance dans la modélisation est la fourniture de mécanismes d'évitement de conflit, c'est-à-dire, une exclusion mutuelle à travers le contrôle de tour d'utilisation des commandes de modélisation *(floor control)*. Comme les sessions ont un petit nombre de participants, les *floors* doivent être optionnels, puisque, selon la situation, la simple dépendance des protocoles sociaux peut être une approche satisfaisante.

Pour des canaux multimédia et d'interaction graphique, le mécanisme de contrôle devrait permettre des activités simultanées, soit pour tous les participants à la fois ou selon des politiques basées sur les rôles (par ex. lors d'une session de formation contrôlée par un instructeur).

Les mécanismes de contrôle devraient être aisément extensibles pour faciliter leur évolution et l'introduction des méthodes plus sophistiquée de contrôle d'utilisation et de concurrence, à l'avenir, comme analysé au chapitre 2.

1.7.6 Conscience de groupe

Permettre aux participants, dans une session, d'être conscients de la présence et des actions des autres est essentiel pour la coordination et l'efficacité d'un travail coopératif. La conscience de groupe (*awareness*) doit être fournie à travers différents moyens. L'information doit être aisément accessible à l'interface de l'utilisateur, concernant les participants dans une session, leurs rôles et droits de contrôle, selon le type de collaboration mis en place. Des télépointeurs et des annotations, associés aux participants à travers des signaux visuels tels que des marques ou des couleurs, sont très importants pour permettre l'indication de caractéristiques sur le modèle partagé. Comme analysé au paragraphe 1.7.3, la communication audio et vidéo sert aussi de mécanismes fondamentaux de conscience de groupe.

1.7.7 Interfaces hétérogènes

Les activités scientifiques considérées demandent l'utilisation d'interfaces d'utilisateurs hétérogènes. L'application a besoin d'être adaptable aux utilisateurs de différentes disciplines, étant donné que la fourniture de toute la fonctionnalité dans un système statique la rendrait trop complexe. Ceci empêche l'utilisation seulement de solutions de collaboration qui demandent des configurations d'interface identiques dans tous les sites (par ex. partage d'application, paragraphe 2.3.1).

Les participants devraient pouvoir collaborer de façon synchronisée en utilisant des paradigmes d'interaction différents (par ex. interfaces de réalité virtuelle et d'utilisateur d'ordinateur de bureau). Des mécanismes doivent être fournis de sorte que ceci puisse être pris en compte dans la définition des rôles des participants et sur le comportement des canaux de coopération et communication.

1.7.8 Conditions d'opération hétérogènes

Le système doit être capable de travailler sous des conditions hétérogènes. Ceci interdit la dépendance de solutions qui supposent la disponibilité d'une haute largeur de bande de réseau. Pour la communication multimédia, des compétences de multidiffusion ne devraient pas être présumées disponibles. Le contrôle sur des flux individuels audio et vidéo devrait être possible, de sorte que des ajustements peuvent être faits pour les conditions de chaque site, tenant compte du type de conférence et des rôles des participants.

Toutes les solutions employées doivent être multiplateformes, de sorte que différents systèmes d'exploitation puissent être utilisés dans chaque site.

Chapitre 2

Collaboration à Distance

2.1 Introduction

Le champ du TCAO (Travail Coopératif Assisté par ordinateur; en anglais, CSCW, *Computer-Supported Cooperative Work*) concerne l'étude de l'utilisation de la technologie informatique pour appuyer et améliorer le travail de groupe. Il a été établi comme domaine de recherche depuis le milieu des années 80 [Gru94a, Gru94b], avec de solides contributions des sciences sociales (psychologie, sociologie, anthropologie, ethnographie, etc.) et plusieurs sous-champs de la science informatique (systèmes distribués, communications et réseaux, interaction homme-machine, ingénierie de logiciels, infographie et multimédia). Pour des analyses socialement orientées du travail coopératif, voyez, par exemple, [Sch02, Ack01, Sha94].

Les applications utilisées pour appuyer la coopération de groupe sont généralement appelées *collecticiels* (en anglais, *groupware*). Comme défini dans [Gre91]: "Le collecticiel est un logiciel qui supporte le travail de groupe. C'est un label techniquement orientée pour différencier les produits explicitement conçus pour assister les groupes de gens travaillant ensemble, des produits qui aident les gens à poursuivre uniquement leurs tâches isolées".

Ainsi, la distinction la plus souvent acceptée entre les termes *collecticiel* et *TCAO* (*groupware* et *CSCW*) est que le premier se réfère primairement aux systèmes réels qui appuient le travail de groupe, tandis que le second indique l'étude des techniques et outils informatiques ainsi que leurs aspects sociaux et organisationnels. Néanmoins, beaucoup d'auteurs utilisent parfois *groupware* et *CSCW* comme des synonymes. D'autres termes souvent employés pour collecticiel sont *applications coopératives* (ou *collaboratives*) et *systèmes coopératifs* (ou *collaboratifs*).

Il existe une large gamme de systèmes collecticiels, comme téléconférence, édition collaborative, apprentissage à distance, systèmes de gestion de flux de travaux (*workflow*), systèmes d'aide à la décision et jeux à multi-utilisateurs. Dans ce travail, nous sommes particulièrement intéressés par les systèmes de modélisation

collaborative synchrone, un type de co-conception. Il est important de noter que le type de contenu édité (documents textuels, graphiques 2D, objets 3D, etc.) a des conséquences considérables dans beaucoup d'aspects du système. Par exemple, en raison de l'importance et de la relative simplicité des documents textuels, de nombreux systèmes d'édition collaborative basés sur des textes ont été développés. D'un autre côté, la modélisation collaborative d'objets tridimensionnels complexes de géosciences pose quelques problèmes particulièrement difficiles, avec des conséquences sur l'architecture du système et sur les techniques exigées pour garantir la réactivité et la consistance.

Les systèmes coopératifs ne devraient pas seulement permettre à des utilisateurs multiples d'interagir avec des objets partagés mais aussi de communiquer et de coordonner leurs actions. C'est-à-dire, ils doivent fournir trois types de services: *production*, *communication* et *coordination :*

- la coopération concerne la production d'artéfacts communs à travers les opérations disponibles au groupe ;
- la communication concerne la communication entre les participants pendant le travail;
- la coordination permet la gestion de relations entre les participants et de dépendances entre les activités.

Celles-ci sont identifiées comme les trois composants essentiels de haut niveau des modèles conceptuels de collaboration [EGR91], parfois avec des variations dans la terminologie. Par exemple, dans le Modèle Clover [Sal95, LN02], le terme *production* est utilisé au lieu de coopération. Ce modèle à son tour est basé sur le modèle conceptuel proposé par Ellis et Weiner [EW94], qui décompose le collecticiel en trois sous-modèles fonctionnels complémentaires :

- le modèle ontologique, qui décrit les objets manipulés et les opérations possibles (le résultat du travail avec le système de base, étendu à une fonctionnalité de collaboration);
- le modèle de coordination, qui identifie les acteurs (participants) et leurs rôles, ainsi que les tâches, activités et leurs relations;
- le modèle d'interface, qui décrit l'interface des utilisateurs avec le système et avec les autres utilisateurs.

Les modèles ontologique et de coordination correspondent, respectivement, aux services de coopération et coordination décrits précédemment. Le modèle d'interface d'Ellis, cependant, comprend à la fois les systèmes d'interaction et de communication entre les utilisateurs. Salber [Sal95] et d'autres auteurs considèrent que l'interface homme-machine devrait être traitée séparément des autres services fonctionnels. Dans ce sens, le composant communication se réfère uniquement à la communication entre les utilisateurs, n'incluant pas l'aspect d'interface utilisateur-ordinateur, qui est

considéré indépendamment du noyau fonctionnel, étant donné que l'interface de l'utilisateur doit refléter les trois services: coopération, coordination et communication.

L'importance relative de chacun de ces composants dépend des objectifs poursuivis par des types spécifiques de systèmes coopératifs. Par exemple, les systèmes synchrones d'édition collaborative marquent la fonctionnalité de coopération, souvent supportant la coordination dans un sens restreint (coordination d'activités synchrones) et fournissant la communication comme un service indépendant, non intégré. Des *mediaspaces* [Mac99], de l'autre côté, marquent une communication informelle pour améliorer la conscience de groupe de l'équipe, ne supportant généralement aucune fonctionnalité de coopération et une coordination restreinte (contrôle de l'utilisation simultanée de canaux de communication).

La même chose arrive avec des architectures et des outils assistant le développement de systèmes coopératifs. Par exemple, l'outil GroupKit [RG96a] privilégie la coordination et la gestion de session, mais réserve relativement peu d'attention aux aspects de la production. Des plateformes comme COCA [LM98] et DCWPL [CM96] fournissent des mécanismes de coordination flexibles, mais pas de support direct pour la production et la communication. Beaucoup d'architectures traitent la communication comme un aspect indépendant et ne permettent pas une réelle intégration avec les autres services. En fait, l'intégration efficace de la communication avec la coopération et la coordination a été identifiée comme un but important à poursuivre par les applications coopératives [Pra99].

Dans notre travail, ces trois aspects du modèle de collaboration conceptuel sont traités de la manière suivante :

- les mécanismes de coopération et les services de gestion de sessions sont dynamiquement introduits dans l'application mono-utilisateur de base en chargeant un plugin basé sur CORBA;
- les services de communication sont fournis à travers un outil de visioconférence, intégré avec la coopération et avec les composants de coordination: les canaux audio et vidéo peuvent être directement activés par les participants d'une session à partir de l'interface du système et sont constamment contrôlés selon les politiques basées sur les rôles, valables pour tous les "canaux de collaboration" (paragraphe 3.3.2);
- les mécanismes de coordination sont extraits dans un module indépendant, qui est implémenté dans un langage de script interprété et fonctionne en connexion avec le composant coopération, contrôlant l'attribution des rôles aux participants et l'utilisation des canaux de collaboration.

Dans le reste du chapitre, nous analysons les principales questions impliquées dans le développement de systèmes coopératifs, en commentant leurs relations avec l'architecture proposée, qui sera décrite au chapitre 3.

2.2 Classification des systèmes coopératifs

Une classification traditionnelle de systèmes coopératifs est basée sur les types de situations dans lesquelles le travail prend place en termes de distribution en temps versus distribution en espace, comme représenté dans le tableau suivant [EGR91]:

	Même temps	**Temps différents**
Même endroit	interaction face à face	interaction asynchrone
Endroits différents	interaction synchrone distribuée	interaction asynchrone distribuée

TAB 2.1 – Classification des systèmes collecticiels

Cette classification représente la distinction basique entre les systèmes synchrones et asynchrones. Elle peut être étendue, par exemple, avec l'addition d'autres dimensions comme la taille du groupe, ou la considération de prévisibilité de temps et endroit, comme dans [Gru94a], ou encore avec la considération d'interaction multi-synchrone, dans laquelle il y a une alternance entre les cycles de divergence (asynchrone) et convergence (synchrone) [MMB01]. D'autres taxonomies pour les systèmes coopératifs tiennent compte du type de fonctionnalité fournie et du degré de partage des tâches ou environnement [EGR91], en différenciant, par exemple, les éditeurs partagés des environnements virtuels collaboratifs.

Un exemple de collecticiel conçu pour appuyer une interaction face à face sont les systèmes de salles de réunion. Une collaboration sur place arrive aussi souvent avec des personnes utilisant des applications standard mono-utilisateur, à tour de rôle pour le contrôle de la souris et du clavier. Un concept intéressant mais encore largement sous-exploité destiné à améliorer ce type de collaboration est celui des systèmes "Single Display Groupware" [SBD99 , BIS+99, BF91], dans lesquels des utilisateurs multiples peuvent interagir avec une seule instance d'une application, sur le même affichage et en même temps, en utilisant des dispositifs d'interaction individuels (comme dans les jeux à multi-utilisateurs). Dans ce cas, le système doit être capable d'associer l'entrée venant de différents dispositifs à différents utilisateurs, en utilisant des widgets et interfaces multi-utilisateurs spécifiquement conçus (mais plusieurs questions restent à traiter).

Dans une interaction asynchrone, les actions d'un seul utilisateur n'affectent pas directement les autres en même temps qu'elles arrivent, et la collaboration se produit pendant des périodes de temps indéfinies. Dans ce cas, d'un point de vue de médiation informatique, être au même lieu ou à distance n'est pas important. Des exemples habituels de systèmes collecticiels asynchrones sont les systèmes de gestion de flux de travaux cooperatifs (*cooperative workflow systems*).

Les systèmes collecticiels synchrones permettent à un groupe d'utilisateurs de travailler simultanément sur un artéfact commun, de sorte que chacun soit capable de noter les modifications réalisées par les autres dans ce qui est perçu comme la réalité à un temps donné. Ces systèmes peuvent être encore classés comme à "collaboration implicite" (*collaboration unaware*) ou à "collaboration explicite" (*collaboration aware*) [RSVW94]. Dans ce contexte, "*awareness*" (conscience de groupe) se réfère à la fourniture, ou non, par l'application de fonctionnalités spécifiques pour appuyer la collaboration. Les systèmes à collaboration implicite permettent le partage à distance, à travers les mécanismes externes, d'applications mono-utilisateur qui ne supportent pas explicitement la collaboration. Comme nous l'analysons ci-dessous, ils sont utiles dans certains contextes, mais présentent plusieurs limitations. Les applications à collaboration explicite sont spécifiquement développées ou adaptées pour supporter la collaboration, fournissant les services nécessaires d'une manière intégrée, comme proposée dans ce travail.

2.3 Collaboration synchrone à distance

2.3.1 Systèmes à collaboration implicite

Cette approche permet l'utilisation simultanée par différents utilisateurs d'une instance d'une application mono-utilisateur standard sous le contrôle d'un progiciel approprié. Dans certains cas, des composants matériels peuvent aussi être employés. Un nombre de solutions est disponible, en utilisant des mécanismes qui agissent à différentes couches (niveaux sémantiques) de l'application (ex. écran, fenêtre, GUI).

L'approche la plus populaire est le partage d'application au niveau du système de fenêtrage, comme implémenté dans beaucoup de systèmes basés sur X-Windows [LL02], Microsoft's Windows NetMeeting [MNM, Sum98] et par d'autres vendeurs. D'autres solutions sont le partage d'écran à travers des flux vidéo (à un niveau sémantique plus bas) et le partage d'événements de widgets GUI (à un niveau plus élevé, mais demandant la réplication de l'application).

Le principal avantage commun de tous les schémas de collaboration implicite est qu'aucune version de l'application spécifiquement développée pour la collaboration n'est nécessaire, des applications standard peuvent être utilisées de façon collaborative.

Les principaux désavantages sont :

- Strict WYSIWIS (*What You See Is What I See;* en français, Ce Que Vous Voyez Est Ce Que Je Vois) est appliqué. Tous les utilisateurs doivent partager exactement la même configuration d'interface, étant donné qu'une seule instance de l'application est utilisée.
- Des activités concurrentes ne sont pas possibles, pour la même raison. Un mécanisme peut être utilisé pour le tour de rôle des utilisateurs pour le contrôle de toute l'application.
- Comme l'application et le modèle résident uniquement dans le site du serveur, l'alternance entre des sessions de travail privées et collaboratives ou entre une collaboration synchrone et asynchrone ne sont pas possibles.
- Une largeur de bande considérable est exigée, étant donnée que toute la communication parmi les endroits à distance se passe à un niveau sémantique bas. Ainsi la dissémination de ce type d'outil peut mener à une surcharge considérable dans le réseau d'entreprise.
- Il n'y a pas de support approprié de conscience du groupe, étant donné que l'application partagée travaille comme si opérée par un seul utilisateur, ne fournissant aucune information relative au groupe.

Ainsi, en relation avec nos conditions (paragraphe 1.7), les outils à collaboration implicite présentent plusieurs limitations au support efficace de la géomodélisation collaborative. Néanmoins, comme ils constituent habituellement la seule alternative pour une collaboration à distance avec des applications mono-utilisateur, ils sont effectivement utiles dans certains cas d'usage. Dans le contexte opérationnel dans lequel ce travail est inséré, deux types de solutions sont parfois employés: partage d'application et flux vidéo.

2.3.1.1 Partage d'application

Le partage d'application permet à une instance de l'application fonctionnant sur une machine, le serveur de l'application, d'être utilisé de façon collaborative. Tous les utilisateurs tournent sur leurs machines un progiciel permettant le partage; au serveur choisi il prend le contrôle de l'application et le diffuse à tous les clients. Le programme réunit aussi les entrées d'utilisateurs, les met en série et les envoie au serveur, qui fonctionne comme sous le contrôle d'un seul utilisateur, celui avec la main. La performance graphique est limitée, compromettant l'interactivité élevée demandée par les applications de visualisation 3D dans notre contexte d'application.

Cette technologie est utile lorsque c'est la seule alternative pour la collaboration à distance, et pour le partage de produits auxiliaires d'usage courant (éditeur de texte, tableurs, logiciel de présentation, etc.) utilisés complémentairement à une application principale de collaboration explicite dans une session. A côté de Microsoft NetMeeting (limité à l'environnement Windows), d'autres outils commerciaux implémentant cette approche sont SunForum, SGIMeeting et Lotus Sametime, tous suivant les standards ITU pour l'interopérabilité avec des solutions de tiers.

2.3.1.2 Flux vidéo

Le partage d'applications à distance peut aussi être atteint à un niveau plus bas que dans le cas précedent, à travers l'échange de flux vidéo entre des endroits éloignés. Ceci peut être fait à travers la diffusion de trames compressées sur des réseaux optiques ou IP, soit avec un matériel, un logiciel ou des solutions hybrides.

Malgré les exigences d'une largeur de bande de réseau plus élevée et parfois un équipement spécial, cette approche peut être très intéressante lorsqu'une bande passante élevée est disponible et le problème décisif est le partage des ressources infographiques à haute performance centralisées, avec une interactivité et qualité graphique élevée. Ceci est important dans certaines situations, par exemple: lorsque le travail avec un très grand modèle sur un serveur de visualisation à haute performance a besoin d'être partagé avec des endroits éloignés, avec une capacité graphique et de traitement limitée; lorsque la distribution d'un grand ensemble de données n'est pas possible en raison du temps de transmission demandé, des restrictions de stockage ou pour des raisons confidentielles; ou lorsqu'une licence de l'application n'est pas disponible au site à distance.

Un exemple d'une solution basée sur un logiciel est SGI Vizserver [SGI], dans lequel le serveur d'application gère les ressources graphiques (*pipes*) d'une machine SGI, en envoyant des trames rendues aux clients à distance. Chaque trame capturée est compressée en utilisant soit des algorithmes à compression avec pertes ou sans pertes. Le client est une application légère qui décompresse et affiche le flux reçu, et dirige aussi toute l'interaction vers le serveur, de sorte que le côté du client se comporte comme si les utilisateurs étaient en train d'interagir localement avec l'ordinateur de bureau à haute performance. De multiples clients à distance peuvent participer à une session collaborative, avec le transfert de contrôle d'interaction parmi les participants.

Une autre possibilité est l'utilisation d'un matériel dédié au traitement vidéo. Des solutions spécifiques ont été développées, basées sur des équipements vidéo combinés avec des réseaux standard Ethernet [FT00]. Des produits encore plus récemment disponibles [TER] sont composés d'un matériel spécial (unités de transmetteur et récepteur) utilisé du côté du serveur et du client pour s'occuper de la compression et décompression du flux, supportant une transmission stéréoscopique à pleine résolution sur des réseaux standard IP. Encore une autre possibilité est la transmission de vidéo non compressée à haute résolution de pleine qualité sur des réseaux optiques, mais avec un coût d'infrastructure beaucoup plus élevé, puisque des fibres dédiées (*dark fibers*) sont nécessaires, avec un débit de quelques Gigaoctets par seconde. Dans ces cas une collaboration à distance impliquant les sites connectés est aussi possible, et est obtenue avec des mécanismes pour le passage du contrôle de toute l'application, à tour de rôle.

Une observation est qu'avec l'augmentation potentielle en disponibilité de réseaux à largeur de bande élevée à des coûts bas, les approches de partage d'application deviennent plus intéressantes. Cependant, le problème de manque d'une vraie collaboration concurrente persiste – mais pourrait être résolu avec l'utilisation

d'applications conçues pour être multi-utilisateurs (comme les systèmes de collecticiel à affichage unique cités au paragraphe 2.2).

2.3.2 Systèmes à collaboration explicite

Dans cette approche, employée dans notre solution et analysée dans le reste de ce chapitre et dans les suivants, la fonctionnalité de collaboration est une responsabilité de l'application elle-même. Des répliques de l'application (ou certains de ses composants) tournent à chaque site impliqué dans la collaboration et maintiennent un état consistant par l'échange d'informations, soit directement, de poste à poste, ou en utilisant un serveur central.

Certains avantages basiques de cette approche sont :

- Une largeur de bande de communication très basse est demandée, si comparée avec les méthodes de collaboration implicite, étant donné que les instances distribuées de l'application échangent uniquement des informations sur les changements d'état, à un niveau sémantique élevé, au lieu de primitives graphiques à bas niveau.
- Il n'y a pas de besoin de synchronisation stricte parmi les interfaces utilisateur des participants (WYSIWIS): des instances locales de l'application peuvent avoir différentes couches d'interface et des fonctionnalités spécifiques; différentes versions de l'application, adaptées aux tâches spécifiques ou à l'équipement, peuvent être utilisées à chaque endroit; et les utilisateurs peuvent avoir des points de vue indépendants.
- Une réplique du modèle et de l'application est disponible à chaque site, ainsi un travail indépendant est possible en-dehors des sessions de collaboration.
- Une interaction concurrente entre différents sites est possible: des télépointeurs et des annotations peuvent être utilisés simultanément, et une édition concurrente peut arriver (mais une gestion de consistance convenable doit être implémentée).

Certains problèmes sont :

- Très peu ont actuellement intégré le support pour la collaboration (avec des degrés de fonctionnalité variés).
- Tous les participants doivent disposer de suffisamment de ressources informatiques et graphiques pour interagir avec les modèles; sinon, si les ressources aux différents sites impliqués dans une session sont trop déséquilibrés, soit tous les participants seront limités par la vitesse du plus lent, soit des grandes divergences peuvent se développer pendant les manipulations de visualisation.
- Habituellement l'opération de l'interface d'utilisateurs (manipulations de menus, etc.) n'est pas perçue aux sites à distance (voir prochain paragraphe).

Comme nous allons l'analyser dans le reste de ce chapitre, le développement de systèmes à collaboration explicite complets implique beaucoup de solutions en dehors des fonctionnalités fournies par l'application de base de domaine spécifique.

2.3.2.1 Affichage d'interface à distance

Normalement les messages échangés entre les instances répliquées des applications à collaboration explicite correspondent aux opérations sur des objets sémantiques (des événements internes au composant graphique de l'interface de l'utilisateur ne sont pas diffusés). De cette manière, l'opération des menus de l'interface utilisateur à un endroit n'est pas reflétée aux autres, seul le résultat final des commandes envoyées.

Néanmoins, pour la consultation, la formation, et pour le transfert de connaissances concernant l'utilisation de l'application en général, l'affichage à distance de l'opération de l'interface est un aspect essentiel – la manière la plus efficace d'apprendre comment faire quelque chose est à travers l'observation du travail des autres. Pour permettre à cela d'arriver, nous avons considéré deux solutions : la diffusion d'événements GUI, ou l'utilisation de flux vidéo pour le partage d'écran (la solution adoptée).

Diffusion d'événements GUI

"*GUI event multiplexing*" [Tie01] est une solution de collaboration dans laquelle chaque utilisateur emploie une instance de l'application adaptée avec une couche spéciale entre le GUI et le récepteur des événements. Cette couche diffuse tous les événements GUI aux sites connectés, et interprète tous les événements reçus comme si l'utilisateur local les avait générés.

Une caractéristique de cette technique est que cela permet non seulement l'exécution à distance d'opérations mais aussi l'affichage à distance de manipulations d'interface. Cependant, cela demande l'utilisation, aux différents sites, de versions identiques de l'application, avec des configurations d'interface identiques. Un autre problème est que cela demande la modification des modules d'interface de l'utilisateur pour le marquage de tous les éléments widget, un effort de programmation considérable qui va contre notre demande de réutilisation de l'application mono-utilisateur.

Partage d'écran

Un outil de flux vidéo peut être employé non seulement pour une communication personnelle mais aussi pour l'affichage à distance du fonctionnement d'un système. Dans [SBMK01], un système type "collaboratoire" (*collaboratory*), deux caméras sont utilisées, l'une pointant vers l'utilisateur et l'autre vers le dispositif d'opération (un nanomanipulateur haptique), et ensuite les utilisateurs peuvent commuter entre les deux pendant la collaboration. Pour l'affichage à distance de l'opération de l'interface utilisateur (GUI), cependant, ce type de réglage devient encombrant, et ne permet pas une bonne lisibilité de l'interface.

Nous avons adapté notre outil de visioconférence sur mesure de sorte qu'il puisse aussi être utilisé pour un affichage à distance de l'opération d'interface, à travers un partage d'écran. Pour cela nous permettons à l'utilisateur de commuter à tout moment l'image transmise, qui peut être soit celle acquise par la caméra soit l'écran capturé (une fenêtre d'affichage spécifique ou l'écran entier compressé vers la résolution vidéo actuelle ; voir paragraphe 3.4.5). Si l'image transmise a une résolution plus haute que la fenêtre de la visioconférence, il y a une dégradation de qualité, dépendant des réglages de compression et de l'échelle utilisée. Mais en général la définition d´image est suffisante pour permettre l'observation à distance des manipulations d'interface.

La situation idéale est quand les participants ont suffisamment d'espace d'écran (par ex. des moniteurs doubles ou grands écrans) et une largeur de bande de réseau suffisante pour utiliser des images à haute résolution. Cette solution a l'avantage de devenir très générale, étant donné qu'aucune homogénéité des configurations d'interface ou des environnements informatiques n'est demandée entre les sites à distance. En comparaison avec l'utilisation des outils d'usage courant, elle permet aussi le contrôle de flux soumis à des politiques basées sur les rôles imposées par l'application d'hébergement.

2.4 Caractérisation architecturale

L'architecture d'un logiciel définit ses composants, leur fonction et l'interconnexion entre eux [SG96]. Commençant par un modèle conceptuel, l'architecture doit définir: la décomposition du système dans les composants fonctionnels majeurs; le modèle de communication entre eux; les fonctions spécifiques réalisées par chaque composant; et les tâches concrètes qui motivent cette sélection [KBAW94].

Dans le cas de systèmes coopératifs, les aspects principaux qui caractérisent une architecture [Dew99] sont: la définition du modèle conceptuel de collaboration, selon la sémantique de l'application de base; la décomposition du système en modules, couches, répliques et processus; et la définition de l'interaction parmi ces composants, spécialement la conscience des modules et couches sur la collaboration. Quelques importantes propriétés attendues sont, par exemple, la performance, la facilité de modification et la réutilisation de code mono-utilisateur.

Dans notre cas, étant donné nos exigences spécifiques (paragraphe 1.7), deux solutions sont particulièrement importantes pour la définition de la conception employée: la réutilisation de la fonctionnalité mono-utilisateur, la réactivité et la performance graphique (étant donné que nous négocions avec des applications de modélisation synchrone avec de très hautes demandes en termes d'interactivité).

Nous basons notre analyse sur le modèle architectural conceptuel de Dewan [Dew99], qui sert d'environnement-cadre pour l'analyse des questions et décisions impliquées dans la conception des systèmes coopératifs. Le modèle Dewan est une généralisation du "*zipper model*" (modèle fermeture éclair) de Patterson [Pat94] pour le collecticiel synchrone, et décrit une application coopérative comme une pile de

couches partagées et répliquées qui communiquent les unes avec les autres à travers l'échange d'événements.

Cela commence avec la définition d'un modèle général pour l'édition collaborative. L'application est vue comme un éditeur d'objets sémantiques (dans notre cas, des modèles géosciences 3D), et l'utilisateur interagit avec elle par l'édition d'un rendu de ces objets utilisant des commandes (Figure 2.1).

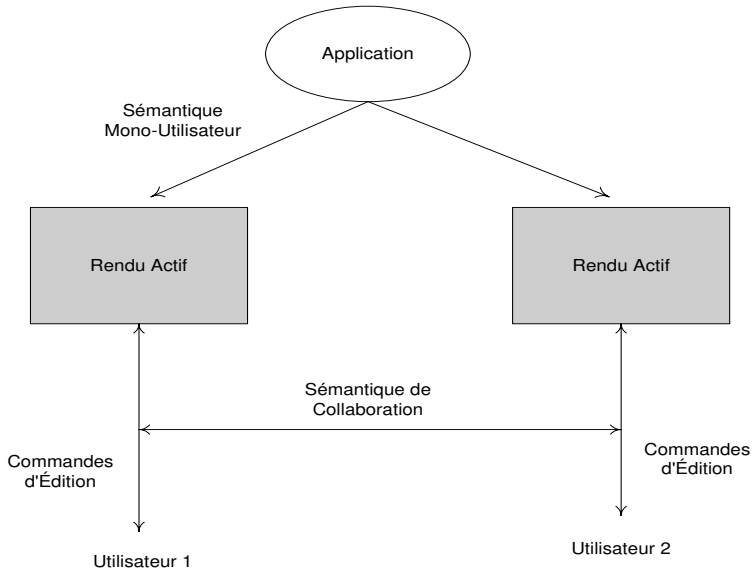


FIG 2.1 – Modèle de collaboration basée sur le paradigme d'édition [Dew99].

Les messages échangés parmi les modules de l'application peuvent être classifiés comme événements d'*interaction* ou de *collaboration*, basés sur le fait qu'ils supportent la sémantique individuelle ou de collaboration de l'application, respectivement. Les événements de collaboration sont traités dans notre architecture comme "canaux de collaboration", une généralisation des différents types d'informations échangées par les applications répliquées (commandes d'édition, événements d'interaction graphiques et communication multimédia), comme analysé au chapitre 3.

Cette architecture générique est ensuite raffinée pour décrire de possibles implémentations, considérant que :

- L'interaction des utilisateurs dans des applications est traitée par une hiérarchie de couches (c'est-à-dire, de haut en bas: modèle, vue, widget, fenêtre, écran), les couches du bas plus près du matériel et les couches du haut plus près du

modèle. La couche du haut se désigne aussi comme la couche *sémantique*, contenant les objets sémantiques de l'application de base.

- Certains niveaux peuvent être *partagés* (une instance unique des modules à cette couche traite l'interaction de multiples utilisateurs) tandis que d'autres sont répliqués ou versionnés (des instances privées des modules à cette couche sont responsables du traitement des événements d'interaction et de collaboration à chaque site des utilisateurs). Le *niveau de réplication* du système est défini comme le niveau de la couche répliquée la plus haute.
- Des messages de collaboration peuvent être échangés à tout niveau, ou même entre différents niveaux. Le *niveau de conscience de collaboration* (*collaboration-awareness level*) est aussi défini comme le niveau de la couche la plus haute qui est consciente de la collaboration.

Tous les modules du système ne suivent pas forcément ce protocole de couches : certains modules peuvent être *externes* (paragraphe 2.4.3).

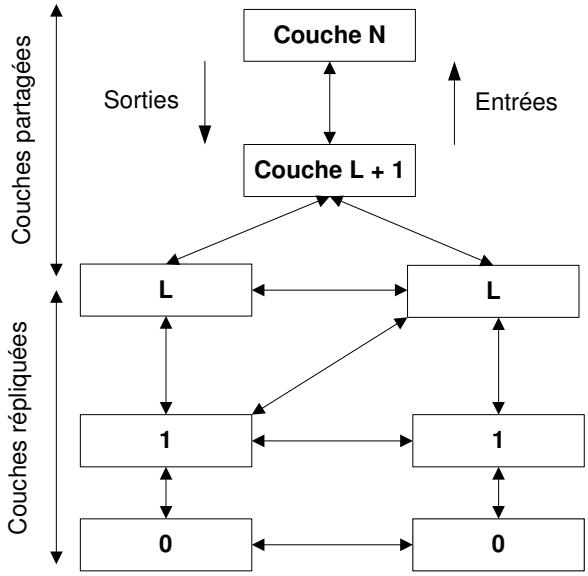


FIG. 2.2 – Architecture générique de Dewan [Dew99].

La structure générale d'un système est composée d'une tige et de branches (Figure 2.2). La tige est composée de couches partagées (L+1 to N), et une branche est composée de couches répliquées (0 to L). Les utilisateurs de l'application interagissent

avec les couches répliquées et les objets qu'elles contiennent, qui sont maintenues consistantes à travers l'échange d'événements de collaboration.

En principe différentes configurations possibles de cette architecture générique peuvent être implémentées, selon la manière dont certaines questions de définition ("dimensions" de l'espace de conception) sont considérées: architecture mono-utilisateur, concurrence, distribution, réplication, conscience de groupe. Aux prochains paragraphes, nous analysons les stratégies de réplication et de conscience de groupe.

2.4.1 Réplication

Les architectures des systèmes coopératifs sont habituellement classifiées comme centralisées ou répliquées, selon la localisation du modèle partagé. Dans un schéma centralisé, uniquement une version du modèle existe, à un site central (en fait, selon le modèle de Dewan, ceci correspond à une couche partagée du haut et à un degré de réplication sous L). Dans des architectures répliquées, des versions du modèle (et de l'application) existent à chaque site (un degré de réplication L, ou réplication intégrale).

Avec des architectures centralisées, le contrôle de concurrence est géré à travers la sérialisation d'événements, et le problème du maintien de consistance devient plus simple. Par conséquent, des solutions centralisées sont utilisées dans beaucoup de systèmes, spécialement dans le cas de collaboration asynchrone. Cependant, un problème fondamental de la centralisation est qu'elle freine la réactivité, puisque tous les événements, y compris les requêtes de lecture, doivent passer à travers le réseau, avec des retards associés au rassemblage et au transport de données.

Pour une modélisation synchrone collaborative basée sur l'infographie, la centralisation n'est pas l'idéal. Le temps demandé pour actualiser les vues, même pour des actions locales, n'atteindra pas la haute interactivité demandée. Ceci tient aux réseaux locaux, et est particulièrement critique pour des réseaux étendus, où le temps d'attente est plus élevé. Par conséquent, les systèmes graphiques coopératifs utilisent souvent des architectures intégralement répliquées, qui permettent une très haute réactivité, étant donné que l'accès au modèle est local. La communication exigée pour le maintien de consistance peut être très efficace, puisque les répliques ont besoin de diffuser uniquement des opérations de changement d'état.

Pour la synchronisation des vues 3D à distance aux taux élevés, comme attendu pour l'interaction graphique, les sites à distance peuvent percevoir un certain retard (plus ou moins sérieux selon le temps d'attente du réseau), mais normalement ceci ne devrait pas compromettre la coopération, étant donné que des positions de visualisation identiques peuvent être rapidement rétablies lorsque l'interaction s'arrête. Pour cela, des points de vue absolus doivent être transmis au lieu de changements de position (de sorte que les messages perdus ne sont pas importants). Une attention spéciale doit être donnée pour éviter les goulots d'étranglement et les débordements: le taux de diffusion des événements d'interaction graphique devraient être contrôlés (les événements devraient être filtrés).

La réplication permet plus de concurrence d'activités et aussi de divergence: la synchronisation de vues répliquées peut être suspendue pour l'utilisation de vues hétérogènes (points de vue indépendants, affichage de différents objets ou les mêmes objets d'une manière différente). Pour l'édition du modèle, cependant, le contrôle de concurrence devient un problème, comme analysé au paragraphe 2.5.

2.4.2 Degré de conscience de collaboration

Une décision doit être prise concernant le choix des niveaux d'une architecture que doivent être faites conscientes de la collaboration (*collaboration aware*), et comment. Dans notre cas, nous avons décidé de ne garder que la couche du haut (sémantique) consciente de la collaboration. Selon les termes de Dewan, ceci est appelé degré L de conscience (ou simplement "*application awareness*").

Maintenir la conscience de collaboration au plus haut niveau de l'application fournit plus de degrés de liberté de divergence aux couches inférieures (par ex. un modèle partagé peut être représenté par des vues indépendantes), et rend la conception plus simple. Mais l'option de rajouter la conscience de collaboration dans plus d'un niveau peut avoir certains bénéfices. Nous avons, par exemple, considéré la possibilité de faire le niveau widget aussi consciente de la collaboration, dans le but de permettre l'affichage à distance de l'opération de l'interface, comme analysé auparavant. Cependant, ceci demanderait une coordination parmi les différentes couches, et dupliquerait la fonctionnalité et rendrait l'évolution du système plus complexe.

Comme analysé dans [Dew99], en principe la communication de couches répliquées peut être faite à travers deux mécanismes différents: l'inclusion de pseudo-couches entre deux couches existantes de l'application mono-utilisateur, ou l'addition de la fonctionnalité de collaboration aux couches existantes.

L'utilisation de pseudo-couches ne demande pas le changement des modules originaux, mais leur inclusion dans le système demande une recompilation (ou re-liaison) et redistribution, quelque chose que nous voulons éviter. Aussi, dans ce cas, la communication est passée à travers la couche extra, avec une pénalité de performance possible.

L'approche de rajouter la fonctionnalité de collaboration directement aux couches originales est plus efficace, mais implique les inconvénients suivants: elle limite la réutilisation de l'application existante, étant donné que cela demande des changements aux couches que sont conscientes de la collaboration, et cela n'est pas viable si le code de la source de la couche à changer n'est pas disponible (souvent le cas).

Une caractéristique fondamentale proposée dans notre solution est que nous rajoutons la fonctionnalité de collaboration à l'application originale sans modifier son code original, exclusivement avec l'utilisation des mécanismes d'extension introduits à travers des plugins, qui redéfinissent dynamiquement certaines classes de l'application pour la création de mécanismes d'interception et de diffusion pour des commandes et pour des événements d'interaction graphique, qui sont transformés en événements de collaboration.

2.4.3 Modules externes

Une partie de la fonctionnalité de collaboration demandée devrait peut-être mieux être maintenue en-dehors de la structure en couches. Dans notre cas nous avons deux types de modules externes (comme analysé en détail au chapitre 3): un gestionnaire général pour le système, responsable de la gestion de session (service de répertoire, création de conférences, inclusion et exclusion de participants) et une conférence centrale responsable des mécanismes de diffusion et des politiques de coordination (gérée par un composant indépendant qui charge les spécifications au temps d'exécution).

2.5 Questions de systèmes coopératifs

À côté des aspects architecturaux analysés aux paragraphes précédents, d'autres questions basiques doivent être prises en compte dans la conception de systèmes à collaboration explicite. Dans ce paragraphe nous analysons les concepts en rapport avec certaines de ces questions, dans le contexte de notre application : maintien de consistance, coordination, gestion de session, conscience de groupe, protocoles de mise en réseau et sécurité.

Un problème central pour les systèmes coopératifs est l'utilisation concurrente des ressources partagées. Cette question a été intensivement étudiée dans CSCW sous divers aspects. Selon la situation, deux approches différentes peuvent être employées : une prévention de conflit (à travers un contrôle d'utilisation) ou un contrôle de concurrence.

2.5.1 Contrôle d'utilisation

Les *floors* sont des accès temporaires et des permissions de manipulation accordées aux participants dans une session de collaboration dans le but d'imposer une exclusion mutuelle sur l'utilisation de ressources partagées, en établissant un tour de rôle organisé. Le contrôle de d'utilisation (*floor control*) est habituellement utilisé pour l'évitement de conflit dans l'utilisation de médias en continu (audio et vidéo), dispositifs à distance, ou même des applications entières [DG97, GCL89]. Un exemple est le tour de rôle pour l'utilisation du canal audio pour permettre des discussions organisées (l'expression prendre le "*floor*", ou prendre la main, vient de l'acte d'utilisation de la scène dans une conférence pour la personne qui parle). Dans l'édition en groupe, les *floors* sont aussi utilisés pour une exclusion mutuelle sur l'utilisation d'objets et d'opérations, si permettre la manipulation concurrente d'un artéfact n'est pas souhaité ou possible.

Conformément au type de ressource contrôlée, "avoir le floor" peut indiquer différentes significations : "transmettre" pour des canaux audio ou vidéo, "modifier"

pour des objets partagés, “changer de position” pour un télépointeur, “contrôler” pour des instruments à distance, etc.

L’opération de *floors* est basée sur des rôles assumés par les participants en ce qui concerne le contrôle d’utilisation : *détenteur de floor* (ou *contrôleur*), *propriétaire de floor*, *demandeur de floor,* etc. Ces rôles peuvent avoir des associations avec les rôles sociaux assumés par les participants dans chaque type de collaboration. Par exemple, comme nous le verrons plus tard, dans notre système pour chaque type de conférence un rôle social (par ex. professeur dans une salle de classe) est toujours désigné comme le *propriétaire de la session,* impliquant que le seul preneur de ce rôle deviendra automatiquement le président de la conférence, et le propriétaire de toutes les *floors*. Le propriétaire du *floor* a des droits préférentiels, et peut assigner le contrôle à n'importe quel participant. Seul le *détenteur du floor* (l’utilisateur contrôlant le *floor* à un moment donné) est autorisé à utiliser la ressource sous contrôle.

Dans les schémas généraux de contrôle d’utilisation [DG97], les *floors* peuvent être caractérisés par des attributs tels l’état (détenue, demandée, libre), la politique d’attribution (explicite par le coordinateur, implicite par les limites de temps, signalée par des dispositifs d’entrée, basée sur un événement, en attente, pas en attente), granularité (global, ressource spécifique), permissions (exécuter, modifier, transmettre, bouger, noter, lire, écrire, etc.). La politique par défaut pour les ressources partagées est habituellement “libre pour tous”, et les conflits sont résolus par la sérialisation des demandes de contrôle sur une base de premier-arrivé-premier-servi.

Si un *floor* est global, tout le système est contrôlé par le détenteur actuel. Ceci est une solution couramment utilisée dans des systèmes à collaboration implicite, qui permettent le partage d'une instance unique d'une application mono-utilisateur. Dans ce cas la libération et la demande de contrôle sont faites habituellement à travers des touches du clavier. Dans des systèmes à collaboration explicite, des *floors* indépendants peuvent être établis pour des ressources différentes. Dans ce cas, l’information de contrôle de *floor* doit être clairement indiquée dans l’interface de l’utilisateur.

Dans notre implémentation, nous traitons toutes les ressources partagées, soit médias continus (audio, vidéo), événements d’interaction (mouvements de caméra, télépointeurs, etc.), ou commandes, comme “canaux de collaboration”. Ensuite nous employons un mécanisme uniforme pour permettre l’établissement de floors optionnels, indépendants pour chacun de ces canaux. La coordination est gérée par un module de contrôle annexé à la conférence centrale.

Comme analysé au paragraphe suivant, un contrôle de concurrence efficace dans le contexte de modélisation géologique tridimensionnelle est encore un défi, ainsi nous comptons sur le contrôle d’utilisation pour éviter des conflits dans l’édition collaborative. Le schéma est simple mais, comme il est extensible, des politiques plus sophistiquées peuvent être implémentées plus tard indépendamment d’autres modules du système. Eviter des conflits de modélisation à travers l’utilisation du contrôle d’utilisation a été satisfaisant pour les dynamiques des activités considérées. Nous avons remarqué que dans de petits groupes, les participants (habituellement seulement deux ou trois) sont en communication constante et spontanée avec les autres concernant

ce qu'ils sont en train de faire à travers les canaux audio et vidéo, et préfèrent souvent compter sur un protocole social pour coordonner leurs actions, au lieu de l'utilisation de mécanismes de contrôle formels. Ce comportement a aussi été observé ailleurs [MD96, GM94, LJDB99].

2.5.2 Contrôle de concurrence

Le contrôle de concurrence permet la synchronisation d'une activité simultanée et la résolution de conflits parmi les participants dans une session collaborative. Dans une architecture répliquée, si des objets partagés peuvent être modifiés simultanément, les techniques de contrôle de concurrence sont demandées pour assurer leur maintien en état consistant. Par exemple, considérant deux sites 1 et 2 avec des répliques d'un modèle, si les deux utilisateurs essaient de l'éditer simultanément, avec des opérations A et B, d'abord exécutées localement et ensuite diffusées, les opérations seraient appliquées dans des ordres différents aux deux sites, menant à une possible inconsistance (Figure 2.3).

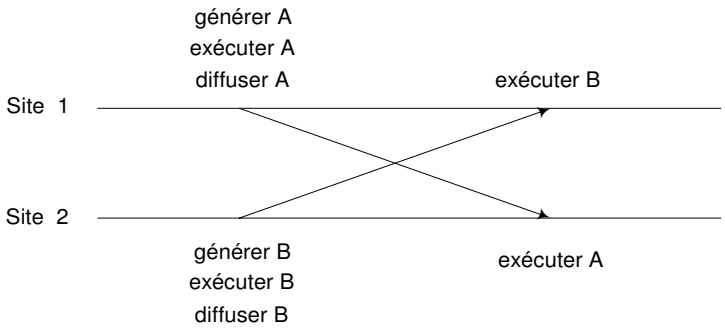


FIG 2.3 – Des opérations exécutées sans contrôle de concurrence à deux sites peuvent mener à des états inconsistants [Pra99], étant donné que l'ordre d'exécution est différent à chaque site.

Le contrôle de concurrence constitue un domaine de recherche propre. Cependant, des techniques traditionnelles utilisées dans des systèmes distribués de base de données ne sont pas directement applicables à un collecticiel, à cause des différents buts de ces systèmes [MD96].

Des conflits de base de données sont principalement liés à la lecture et écrite des entrées, et sont résolus avec l'utilisation de transactions qui sont effectuées quels que soient les conflits, mais gardées uniquement en cas de succès. Sinon elles sont annulées, et les opérations déjà réalisées sont invalidées. Les systèmes de base de données s'efforcent de cacher la présence d'utilisateurs concurrents des autres, les

protégeant de la vue des états intermédiaires des transactions des autres. Les systèmes coopératifs synchrones, au contraire, doivent laisser les utilisateurs conscients des activités des autres, et garantir la réactivité, une condition particulièrement importante dans la modélisation basée sur l'infographie.

Les besoins de consistance pour les systèmes d'édition collaborative ont mené les chercheurs au développement de diverses techniques spécifiques. Des analyses préliminaires du domaine peuvent être trouvées en [EG89, GM94]. Ici nous exposons brièvement les approches de contrôle de concurrence et les limitations en systèmes graphiques d'édition collaborative, en suivant [Pra99] et [SC02]. Dans le cas particulier de modélisation collaborative tridimensionnelle dans les géosciences, les exigences deviennent spécialement difficiles du fait que les objets modelés sont de grande taille, complexes et interdépendants.

Une première approche possible de gestion de concurrence est l'utilisation de diffusions ordonnées, pour garantir que toutes les opérations sont reçues dans le même ordre à tous les sites. Les opérations réalisées à n'importe quel site doivent d'abord être sérialisées par un processus central, et ensuite appliquées dans le même ordre partout. Une question avec cette approche est que toutes les opérations doivent passer à travers le réseau avant l'exécution locale, résultant en problèmes similaires à l'utilisation d'une architecture centralisée. Un autre problème est que les opérations émises à un site éloigné peuvent finir par être exécutées avant celles émises localement, menant à des résultats non voulus.

En fait, la préservation d'intention est l'une des propriétés de base identifiées comme conditions pour le *maintien de consistance* dans l'édition collaborative [SC02, SJZ+98]:

- convergence: les états du document final ou du modèle doivent être les mêmes à tous les sites;

- préservation de causalité : l'ordre d'exécution doit suivre l'ordre de cause-effet ;

- préservation d'intention : l'effet réel d'une commande exécutée doit être égal à l'effet voulu.

Plusieurs techniques ont été conçues visant à remplir ces conditions dans le contexte de systèmes d'édition collaborative. Elles peuvent être classées soit comme pessimistes ou optimistes. Les techniques pessimistes utilisent des verrous pour prévenir les conflits pouvant arriver. L'idée dans les techniques optimistes n'est pas de prévenir les inconsistances, mais d'essayer de les réparer après leur apparition.

Contrôle de concurrence pessimiste

Les méthodes pessimistes demandent l'acquisition de verrous sur le réseau avant l'exécution d'opérations d'édition. Un premier problème avec cette approche est que acquérir et libérer des verrous peut freiner la réactivité du système, comme dans le cas de la sérialisation.

Un autre problème avec le verrouillage est de décider à quel niveau de granularité verrouiller les objets. Le problème est sérieusement aggravé lorsque les objets peuvent être interdépendants (comme dans notre cas), étant donné que le schéma de verrouillage doit prendre en compte les relations entre eux.

À l'extrême, on peut avoir besoin de verrouiller le modèle entier, ce qui est équivalent à utiliser le contrôle d'utilisation, mais plus complexe. Comme indiqué dans [SJZ+98]: "sauf si la granularité de verrouillage est tout le document, aucun des trois problèmes d'inconsistance ne peut être résolu en verrouillant, étant donné que l'occurrence de ces problèmes est indépendante de si oui ou non les opérations d'édition se réfèrent au même objet".

Néanmoins, sous des conditions spécifiques, beaucoup de systèmes coopératifs d'édition graphique ont adopté une prévention de conflit basée sur le verrouillage [Pra99, PS94, GM94].

Contrôle de concurrence optimiste

Des méthodes optimistes sont utilisées dans des systèmes répliqués d'édition collaborative pour garantir une haute réactivité. Une idée de base dans ces méthodes est d'estampiller les opérations, exécuter celles qui sont locales immédiatement et ensuite diffuser. Ensuite si on détecte que des opérations ont été exécutées dans le désordre, elles sont défaites et refaites dans l'ordre correct (Figure 2.4).

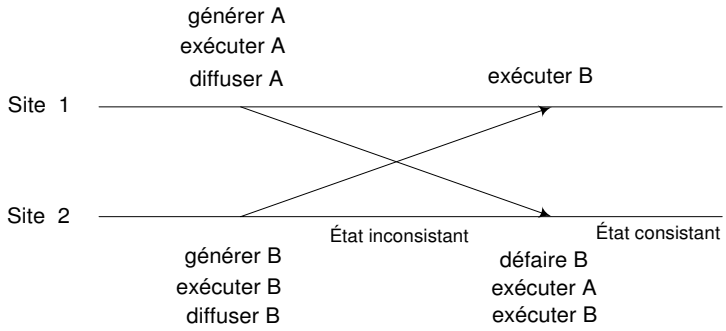


FIG 2.4 – Stratégie de défaire/refaire pour un contrôle de concurrence optimiste [Pra99].

Une limitation avec cette approche, dans notre contexte, est que l'implémentation de mécanismes de sauvegarde d'état et d'annulation (*undo*) à multi-niveaux peuvent être informatiquement chers et très lents, étant donné que les objets sont complexes, très

grands et interdépendants. Dans Gocad, par exemple, des mécanismes d'annulation sont disponibles, mais pas complets (défaire uniquement des opérations relativement simples est faisable; pour des opérations affectées par des contraintes ou impliquant de grands objets, défaire n'est pas disponible).

Un problème difficile avec des méthodes optimistes est comment garantir la consistance avec des intentions d'utilisateurs [Pra99]. Quelques approches utilisées pour la préservation d'intention sont des transformations d'opérations et du "rien faire" (assumer que les utilisateurs peuvent corriger les problèmes, ce qui peut être acceptable dans certaines applications simples comme les tableaux blancs collaboratifs utilisés pour esquisser).

La transformation d'opérations, une technique utilisée en éditeurs de texte de groupe, travaille en transformant les événements arrivant en désordre via un ensemble de règles de manière que l'effet est le même que s'ils étaient arrivés en ordre (voir [SE98] pour une révision complète). Cependant, ces algorithmes ne sont actuellement applicables qu'aux objets ayant une relation d'ordre linéaire, comme un texte.

Une technique à versions multiples a été récemment proposée pour atteindre la préservation d'intention et la convergence dans le domaine de l'édition graphique [SC02]. Elle résout les conflits identifiés en logeant les effets de toutes les opérations dans des versions multiples du même objet, en utilisant un ensemble d'algorithmes et de règles avec des propriétés formellement prouvées qui minimisent le nombre de versions d'objets créés.

Certaines limitations connues de cette méthode sont que les opérations doivent cibler uniquement un objet, les attributs d'objet doivent être indépendants et les objets ont aussi besoin d'être indépendants les uns des autres - conditions qui ne sont pas satisfaites dans notre contexte. Des interfaces d'utilisateur appropriées pour l'affichage et la gestion des versions d'objets sont reconnues comme l'objet de recherche future par les auteurs (même pour l'édition graphique 2D), ainsi que la fusion de versions, les techniques d'annulation à multi-utilisateurs, la consistance sémantique, et les extensions aux systèmes CAO collaboratifs.

En résumé, dans le cas de géomodélisation collaborative, de fortes limitations pour l'application de méthodes de contrôle de concurrence optimiste sont les interdépendances parmi les objets et leur taille et complexité topologique, qui rend une compensation (*rollback*) efficace irréalisable.

2.5.3 Coordination

La coordination peut être généralement définie comme la gestion de dépendances entre les activités [MC94, MC90]. Dans le contexte de systèmes coopératifs, la coordination peut être considérée à deux niveaux: le niveau activité et le niveau objet [EW94].

Au niveau activité, la coordination est en rapport avec la séquence d'activités qui composent une procédure. Cette vue large de coordination est aussi appelée *travail d'articulation* [SB92]: la gestion d'activités distribuée au-delà du temps et de l'espace,

consistant en identification des objectifs du groupe, élaboration de ces objectifs en tâches, sélection des participants, distribution de tâches entre eux et contrôle de l'exécution des tâches. Ceux-ci sont des problèmes fondamentaux pour l'accomplissement d'un grand ensemble d'activités demandant des mécanismes de coordination sophistiqués, où les tâches dépendent l'une de l'autre pour démarrer, pour être réalisées et pour finir, par exemple des activités supportées par des systèmes de flux de travaux, d'apprentissage et de jeux multi-utilisateurs [RF02].

La modélisation collaborative comprend une telle série d'activités, de l'acquisition de données à la prise de décision, qui peuvent bénéficier du support des systèmes de flux de travaux (des mécanismes de flux de travaux sont ancrés dans Gocad et d'autres applications de modélisation). Dans ce travail, cependant, nous sommes principalement concernés par le support de la collaboration pendant la modélisation synchrone, et nous assumons que ces aspects plus larges d'organisation de travail asynchrone ont été écartés.

Par conséquent, nous sommes uniquement intéressés par les méthodes pour le support de la coordination dans son sens étroit: coordination au niveau objet. À ce niveau, le modèle de coordination décrit comment le système négocie avec l'accès simultané de participants multiples aux mêmes ressources (objets, outils, canaux de communication).

Nous accompagnons l'approche proposée dans [LM98, CM96] d'établir une séparation claire entre la coordination et les autres fonctionnalités dans le système (paragraphe 2.6.2). Ceci accorde de la souplesse à l'évolution des politiques de contrôle, qui ne sont pas incorporées au code de l'application. Dans notre schéma actuel, des droits d'accès simples aux canaux de collaboration sont définis en termes de rôles que les participants assument dans chaque type de collaboration. Dans la littérature TCAO, de nombreux systèmes ont utilisé le concept *rôle* comme une base pour un contrôle d'accès plus sophistiqué aux ressources partagées (par ex. [Edw96, EGR91, LM98, CM96]). Certains chercheurs considèrent aussi le concept de rôles dans un contexte plus général de la représentation des aspects sociaux du travail coopératif (par ex., [FBM02, FTK95]).

2.5.4 Gestion de session

Une question importante pour l'utilisation d'une application coopérative est combien elle facilite l'initiation et la gestion de sessions de collaboration. La gestion de sessions [Edw94] se réfère aux services qui ont besoin d'être fournis par le système pour: créer des conférences, effacer des conférences, trouver des conférences existantes, trouver des gens, se joindre à des conférences et les quitter, trouver qui est dans une conférence, leurs rôles et droits, et accepter les retardataires.

Dans notre architecture, les services de gestion de session sont fournis par un module externe centralisé qui fonctionne comme un bureau d'enregistrement (*registrar*). Après l'inscription, les utilisateurs peuvent prendre part aux conférences et

trouver les autres sans la nécessité de connaître l'information d'endroit à un bas niveau (adresses IP et ports TCP/IP).

Nous utilisons une métaphore de "porte ouverte" [RG96a] pour créer et joindre des conférences, dans laquelle les gens pensent en termes de noms de conférence et "appellent" pour entrer en contact. Le propriétaire de la conférence (initiateur) écoute une sonnerie et permet ou non l'entrée du nouveau participant. Les retardataires peuvent recevoir toutes les commandes diffusées depuis le début de la conférence. Un dialogue d'interface d'utilisateurs pour la conférence (Figure A3.1, Appendice 3) énumère les participants et leurs droits sur les canaux de collaboration. N'importe quel membre peut démarrer la visioconférence, qui est automatiquement établie entre les participants.

D'autres politiques possibles de gestion de session utilisées par des systèmes coopératifs sont les points de rendez-vous, où les gens vont dans des "salles" ou "endroits" et ensuite sont automatiquement connectés à d'autres à cet endroit (comme dans TeamRooms [RG96b] et dans beaucoup de Multi-User Dungeons – MUDs); et des schémas document-centriques dans lesquels une session de collaboration est automatiquement établie avec d'autres gens partageant un document lors de l'accès.

2.5.5 Conscience de groupe

La conscience de groupe (awareness) permet aux participants d'une session de percevoir la présence d'autres personnes et de savoir ce qu'elles sont en train de faire [DB92, RSVW94], ainsi les gens sont capables de bâtir un contexte de travail et de coordonner leurs actions. C'est naturellement une question essentielle pour les systèmes coopératifs, profondément en rapport avec d'autres aspects principaux comme la communication, coordination, coopération, gestion de session et interface d'utilisateur.

En collaboration face à face, trois mécanismes principaux aident les gens à maintenir la conscience de groupe [GPS04]: la communication explicite (en parlant des activités en cours); la communication conséquentielle (en regardant le travail d'un autre); et le "feedthrough" (observation des changements des objets partagés comme un résultat du travail).

Avec la médiation informatique dans la collaboration à distance, la conscience de groupe devient difficile à maintenir et le système a besoin de fournir des mécanismes alternatifs pour ces fonctions. La communication explicite, supportée par la visioconférence, joue un rôle essentiel. Dans le cas de petits groupes, des liens simultanés audio et vidéo peuvent être maintenus entre tous les participants, permettant aux gens de garder un échange constant d'informations sur leurs intentions et actions. Bien que différent d'une communication face à face, ceci aide à une coopération harmonieuse, avec moins de besoin d'utilisation explicite de contrôle d'utilisation pour éviter les conflits. La visioconférence peut aussi permettre l'observation du travail d'un autre, jouant un rôle essentiel pour la communication conséquentielle.

Des télépointeurs et des annotations, associés aux participants à travers des signaux visuels tels que des marques ou des couleurs, permettent l'indication de caractéristiques sur le modèle partagé. Ils jouent le rôle de gestes, fréquemment utilisés en communication face à face d'une façon déictique (pour pointer des objets ou endroits, et pour illustrer ce qui est dit verbalement).

En plus des annotations et de simples télépointeurs, d'autres mécanismes peuvent être fournis pour supporter la communication implicite, comme des incarnations visibles de participants et des représentations visuelles d'actions. Ceci est communément fait dans des interfaces de réalité virtuelle, où les avatars peuvent exprimer les positions et gestes des participants en 3D.

Les services de gestion de session ont aussi une fonction de conscience de groupe en permettant l'identification des participants dans une conférence à travers une interface d'utilisateur commode, comme analysé ci-dessus, et aussi l'initiation automatique de la communication multimédia entre les participants, selon leurs rôles.

2.5.6 Mise en réseau

L'implémentation de systèmes efficaces de collaboration synchrone implique la considération d'une gamme de questions, exigences et protocoles de mise en réseau, pour l'intégration de caractéristiques comme l'édition collaborative (qui demande une distribution de message fiable), la visualisation interactive (qui demande un temps faible, rapide, pas nécessairement une communication fiable), et la communication multimédia (qui demande des flux médias en temps réel, continus).

Dans ce travail, nous avons adopté CORBA comme une plateforme d'intergiciel pour l'informatique distribuée, et nous évitons l'implémentation directe d'une fonctionnalité de mise en réseau à bas niveau. Pour la plupart des canaux de collaboration, sauf pour audio et vidéo, nous employons actuellement le Service d'Evénements de CORBA [HV99] pour la distribution des messages. Pour la communication audio et vidéo, notre outil de visioconférence utilise CORBA pour l'échange de commandes, mais repose sur JMF (Java Multimedia Framework, qui utilise RTP, Real Time Protocol) pour un flux média en temps réel, comme analysé au paragraphe 3.4.

Dans les contextes dans lesquels nous avons appliqué le système, les réseaux d'entreprise de zone locale (LAN) et les réseaux de zone étendue (WAN), cette solution a mené à une performance satisfaisante. L'utilisation de Service de Notification de CORBA, une évolution de Service d'Evénements, devrait permettre la définition indépendante des paramètres de qualité de service (QoS) pour chaque type de canal (une caractéristique importante, étant donné que différents canaux ont des exigences de performance différentes), et supporter aussi le filtrage d'événements. La disponibilité de ce service est attendue, mais pas encore fournie actuellement pour l'implémentation libre CORBA que nous employons [PR00]. D'autres implémentations commerciales CORBA fournissent des implémentations de Service de Notification avec divers niveaux de sophistication (par exemple, certains fournissent des multidiffusions UDP

comme un protocole sous-jacent). Une implémentation CORBA de pointe consacrée aux applications en temps réel est présentée dans [GSGP04].

Comme QoS et multidiffusion ne sont pas toujours disponibles à travers les réseaux, une question active de recherche sont les outils “conscientes de réseau” (*network aware*) qui peuvent utiliser automatiquement ces services lorsqu’ils sont disponibles.

2.5.7 Sécurité

Si la collaboration ne se restreint pas à prendre place dans un domaine protégé (un groupe d’objets protégé par un pare-feu commun), les questions de sécurité surviennent. La sécurité de CORBA est une question ample, adressée par la spécification complexe du Service de Sécurité [OMG, Bla99]. Les principaux problèmes rencontrés sont: premièrement, il faut traverser les pare-feus existants; deuxièmement, ceci doit être fait sans compromettre la politique de sécurité en place et en garantissant la sécurité à l'application (c'est-à-dire, s'assurer que les utilisateurs non autorisés ne pourront pas interférer avec la confidentialité et l'intégrité de la communication).

Dans le cas des applications CORBA, il y a certaines difficultés particulièrement ardues pour faire du trafic IIOP (protocole de communication de CORBA) traverser des pare feus (un ou plusieurs – entre un client protégé et des serveurs externes, ou entre des clients externes et des serveurs protégés [OS]).

Un problème est que les références de l’objet CORBA contiennent des informations concernant l’adresse d’hébergement et le numéro de port des objets cibles, mais ceux-ci ne seront pas directement accessibles s’ils se trouvent derrière un pare-feu, qui laissera normalement le trafic passer juste à travers quelques ports contrôlés, à une machine spécifique.

Dans des configurations statiques, les serveurs peuvent être traversés à des ports fixés, et la barrière de sécurité peut être configurée pour laisser le trafic passer à travers. Ceci, cependant, peut créer des trous de sécurité ou peut ne pas être tout à fait possible, selon la politique de sécurité en place. Si les serveurs sont démarrés dynamiquement, cette solution devient encore plus difficile.

Une possibilité, proposée par quelques vendeurs, est l’utilisation d’un “tunnel HTTP”: le trafic IIOP est encapsulé et passé à travers une connexion HTTP (qui utilise un port fixé). Cette solution, cependant, n’est pas considérée sûre, et est inefficace.

Une solution plus souple et plus sûre est d’utiliser des serveurs mandataires (*proxy servers*), qui tournent sur des ports fixés et agissent comme pare-feux eux-mêmes, en utilisant les règles pour accorder l’accès aux objets à l’intérieur du domaine protégé (l'OMG propose l'utilisation de différents types de mandataires à différents niveaux de la couche de communication).

2.6 Systèmes de collaboration associés

2.6.1 Boîtes à outils et plateformes

Le développement de systèmes coopératifs à caractéristiques intégrales à partir de zéro est une tâche énorme, étant donné les nombreuses questions qui doivent être considérées et les problèmes comme le processus difficile pour tester des applications distribuées impliquant des utilisateurs multiples. Des chercheurs ont créé des boîtes à outils de construction de collecticiels pour des classes de systèmes spécifiques, avec le but de réduire l'effort de développement en soulageant le développeur d'un nombre de questions techniques. Les boîtes à outils destinées aux systèmes synchrones fournissent habituellement quelques blocs de construction basiques comme les abstractions de programmation pour la synchronisation de vues et de modèles de données spécifiques, la communication et la gestion de processus, la gestion de session, et support pour le développement d'interfaces d'utilisateurs multiples [GR99]. Des exemples sont Rendezvous [HBRP94, PHRM90] et GroupKit [RG96a].

Rendezvous est un système centralisé basé sur l'architecture Model-View-Controller (MVC). La boîte à outils emploie une version orientée objets de LISP, basée sur des contraintes pour maintenir la consistance entre de multiples vues et leurs données sous-jacentes. Les classes comprennent le support pour télépointeurs, contrôle d'utilisation, texte et graphiques multi-utilisateurs et gestion de session.

Groupkit est une boîte à outils basée Tcl/Tk qui utilise une architecture répliquée avec diffusion d'événements pour envoyer des opérations de changements aux utilisateurs à distance. Un gestionnaire de session configurable est fourni, ainsi que plusieurs widgets de collecticiel y compris des fenêtres d'annotations transparentes, des barres d'outils multi-utilisateurs et des widgets de textes, conscience sociale et support pour télépointeurs.

Des boîtes à outils sont aussi disponibles pour supporter le développement de la téléimmersion collaborative. Un exemple est CAVERNsoft [PKS+00, TEL], qui fournit des modules pour accélérer la construction d'applications de réalité virtuelle collaborative (principalement pour les ordinateurs SGI), et beaucoup d'outils de pointe de mise en réseau pour supporter une collaboration d'immersion à longue distance et haute capacité.

Au-delà des boîtes à outils, quelques plateformes de collaboration à niveau plus élevé fournissent un environnement collaboratif de temps d'exécution associé à un environnement-cadre de développement pour l'implémentation d'applications en conformité. Par exemple, le système Groove [GRO] est une plateforme de collaboration pair à pair orientée au travail de bureau, qui laisse les utilisateurs créer des espaces de travail dans lesquels ils peuvent inviter d'autres utilisateurs avec lesquels ils veulent collaborer. Groove ne fournit aucun modèle partagé de données commun pour les divers outils utilisés dans l'espace de travail partagé; la synchronisation et le partage de données sont gérés à travers l'échange d'objets de "commande" créé par les outils qui encapsulent les opérations de changement, qui sont

interceptées par la plateforme et peuvent être sérialisées et diffusées à des répliques d'outils aux sites connectés. Les outils ont besoin d'être développés spécifiquement en utilisant le Groove SDK (Software Development Kit).

Pour nos pratiques, le problème avec tous ces outils est qu'ils demandent le développement spécifique d'applications selon les environnements-cadres fournis. Ceci devient un sérieux empêchement lorsque l'application mono-utilisateur qui sert de base à la collaboration est complexe et déjà établie opérationnellement.

2.6.2 Composants de coordination

Un paradigme pour le développement d'applications coopératives est l'utilisation de langages et composants de coordination. Différemment des boîtes à outils et environnements-cadres, qui fournissent une fonctionnalité et des services intégrés à haut niveau, une idée fondamentale imposée par des outils comme DCWPL [CM96] et COCA [LM98] est la séparation claire des aspects de coordination et de calcul du système, de sorte que l'implémentation de coordination et de contrôle n'est pas durement codée et incorporée à l'application de base. Au lieu de cela, les modules responsables de la fonctionnalité du domaine spécifique à un seul utilisateur communiquent avec les composants de coordination à travers une interface claire.

La spécification des politiques de contrôle est faite avec des langages interprétés, et par conséquent peut évoluer sans affecter la fonctionnalité informatique la plus stable et sans la nécessité de recompilation du système et redéploiement à tous les sites, ce qui n'est pas toujours faisable dans des situations pratiques. Ceci est particulièrement important, étant donné que les politiques de coordination sont généralement associées à des pratiques de travail qui sont difficiles à saisir d'abord.

DCWPL (Describing Collaborative Work Programming Langage) est un langage de script déclaratif qui fournit des facilités pour la modélisation et le contrôle d'accès aux artéfacts partagés, à travers la définition d'un ensemble d'attributs. L'exécution d'applications est contrôlée par une "machine de contrôle" externe, qui peut interpréter des programmes DCWPL et imposer des politiques de collaboration.

La solution qui a motivé notre travail a été l'architecture et le langage de spécification COCA (Collaborative Objects Coordination Architecture). Chez COCA, les politiques de coordination sont spécifiées avec un langage exécutable basé sur la logique (similaire à Prolog), interprété en temps de passage par une machine virtuelle qui tourne à chaque site. Les participants de la collaboration prennent des rôles, qui sont associés à un ensemble de règles utilisées pour la définition des politiques de coordination, qui représentent tous les mécanismes associés à la collaboration pour le contrôle d'accès, le contrôle de concurrence, le contrôle de session, la distribution de données, etc.

Des descriptions complètes de l'architecture peuvent être trouvées dans [LM98, LM99, LWM99]. Nous montrons ici un exemple simple adapté de [LM99] pour donner une idée du langage, étant donné qu'il sert de référence à la conception de notre propre syntaxe, analysée au chapitre 3. COCA utilise une architecture à double bus. Le *bus de*

collaboration connecte tous les participants impliqués dans une collaboration à différents sites; le *bus de conférence* connecte une instance locale de la machine virtuelle COCA avec différents outils travaillant ensemble à un site. Pour chaque rôle, des règles sont définies, qui sont lancées lorsque les *règles actives* sont unifiées. Le principal type de règle active est :

on-arrival(gate(<*gate name*>), message) :- action

Une porte (*gate*) indique un point où la machine virtuelle COCA interagit avec son environnement, c'est-à-dire, une connexion avec un bus. Cette règle agit lorsqu'un message arrive à une porte causant un traitement, par exemple faisant suivre le même paquet ou de nouveaux paquets à d'autres portes, ou juste en le bloquant.

L'opérateur: **gname ! (<message>)** réalise une diffusion du message vers la porte *gname*. A travers ces mécanismes, des messages de coordination sont transmis à partir des outils d'un participant à ceux des autres.

Dans la spécification très basique montrée dans le Listage 2.1, une application de tableau blanc (wb) a diffusé toutes les opérations à chaque participant, sans contrôle d'utilisation. Un seul rôle est défini, le *drawer* (dessinateur). Les machines virtuelles COCA à chaque site communiquent avec une autre à travers le canal "wb-remote" du bus de collaboration, et échangent des messages avec le *wb* local via les canaux "wb-in" et "wb-out" du bus de conférence. Il y a seulement deux règles actives. L'une est définie pour la porte "wb-remote": lorsqu'une opération Op arrive de quelque wb à distance, on la fait suivre au wb local via la porte "wb-out". Une autre règle est définie pour porte la "wb-in": lorsque l'interface de wb local envoie une opération à la machine virtuelle, elle est renvoyée à l'instance local pour exécution, via la porte "wb-out", et distribuée aux autres wb à travers la porte "wb-remote".

```
collaboration meeting
{
    collaboration-bus
    {
        channel(wb-remote).
    }

    role drawer
    {
        conference-bus
        {
            channel(wb-in).
             channel(wb-out).
        }

       on-arrival(gate(wb-remote). Op) :-
            wb-out !Op.
```

LISTAGE 2.1 – Une spécification de collaboration COCA basique.

Dans cet exemple, chacun peut envoyer des opérations à tout moment, et chaque opération reçue d'autres participants est immédiatement affichée. Comme ceci n'est pas acceptable en collaboration réelle, qui demanderait une interaction plus contrôlée, une politique d'utilisation (*floor policy*) doit être implémentée, rendant la spécification beaucoup plus complexe (voir [LM99]). En fait, COCA est une méta architecture dans le sens où elle ne fournit ou n'impose aucune politique de coordination. Les utilisateurs ont la liberté et l'obligation de spécifier ces politiques à l'aide du langage.

Le langage basé sur la logique est supposé être simple. Cependant, des exemples plus réalistes peuvent rapidement devenir complexes et difficiles à négocier avec pour les utilisateurs non experts en programmation logique (une situation courante). En comparaison, nous adoptons un schéma à niveau plus élevé, dans lequel la majeure partie de la fonctionnalité de collaboration et des services (diffusion, gestion de contrôle d'utilisation et de session) est fournie par défaut, et peut être étendue ou modifiée avec un langage simple procédural et interprété (paragraphe 3.3.5).

Un problème fondamental avec COCA, analysé par les auteurs [LM98], est la performance, étant donné que tous les messages vers et de chaque participant passent à travers la machine virtuelle avant d'être exécutés localement – ce qui n'est pas acceptable pour des applications graphiques interactives. Pour cette raison, nous optons pour un schéma dans lequel le contrôle d'utilisation agit directement sur les instances répliquées de l'application pour bloquer les messages avant qu'ils ne soient émis sur des canaux non autorisés. Tous les événements d'interaction sont exécutés localement, indépendamment de la diffusion, si les changements locaux sont permis.

Les outils tels COCA et DCWPL sont puissants, mais d'un autre côté exigent des compétences de programmation avancées de leurs utilisateurs et demandent le développement d'applications selon les mécanismes de coordination proposés.

2.6.3 Visualisation scientifique collaborative

Les applications de visualisation scientifique sont généralement basées sur un flux de travaux impliquant le filtrage de données crues, la sélection de régions d'intérêt et la transformation de valeurs sous une forme graphique. Les paramètres de visualisation peuvent être ajustés et le processus réexécuté interactivement. Leur architecture diffère fondamentalement des applications de modélisation géométrique scientifique, qui en plus de fournir une fonctionnalité de visualisation similaire supporte la mise en forme d'objets avec une géométrie et une topologie complexes.

Comme aucune édition directe de modèles géométriques ne prend place, la consistance est plus simple à maintenir que dans le cas de modélisation collaborative. Les applications de visualisation collaborative peuvent être classées selon le niveau de contrôle partagé qu'elles permettent sur les paramètres et produits du processus de visualisation [JE98], et selon la façon dont leurs architectures ont évolué.

Certaines ont été développées comme extensions à des systèmes mono-utilisateur existants, tels Collaborative AVS [CAVS], Cspray [PW97] et TeleInVivo [CGS+96]. D'autres systèmes, tels Shastra [AB94], ont été construits à partir de zéro avec le but de supporter la collaboration. Shastra est en fait un environnement collaboratif multimédia à caractéristiques intégrales, qui comprend les composants pour la gestion d'applications distribuées, l'initialisation et le maintien de sessions collaboratives et le support de téléconférences audio et vidéo. De nouveaux outils doivent être développés selon les lignes architecturales spécifiées par l'environnement-cadre.

A part les applications, des bibliothèques graphiques distribuées à niveau plus bas, basées sur des graphes de scène telles Repo-3D [MF98] et Distributed Open Inventor (DIV) [HSFP99] peuvent énormément faciliter le développement de systèmes de visualisation collaborative. En plus des systèmes standard, il y a toujours eu un solide effort pour utiliser une technologie de réalité virtuelle dans la visualisation scientifique, étant donné que RV a le potentiel d'être un outil puissant pour améliorer la compréhension de grands ensembles de données multidimensionnels [DFL+00, JL01]. Un point intéressant est que malgré le fait que la plupart des systèmes de réalité virtuelle immersifs sont conçus pour un unique utilisateur suivi par un capteur de position (les systèmes multi-utilisateurs sont l'objet d'une active recherche [BMC04] mais restent expérimentaux), une collaboration à distance permet la coopération parmi des utilisateurs RV multiples, chacun utilisant un environnement et un point de vue privé, et ouvre de nouvelles possibilités pour améliorer la compréhension de données scientifiques multidimensionnelles [PKL00].

2.6.4 CAO collaborative

Bien qu'en principe similaire à la modélisation géoscientifique, la Conception Assistée par Ordinateur (CAO), orientée par des ingénieurs, mène à des solutions architecturales particulières associées à la nature des modèles de données utilisés [BFD+03]. Spécialement, la plupart des systèmes emploient des bases de données centralisées associées à un intergiciel spécifique, qui favorise l'utilisation d'architectures centralisées ou hybrides. Ceci simplifie le maintien de consistance, mais par contre limite l'interactivité en collaboration.

Plusieurs systèmes, tels [NW98, ZCV00], proposent la visualisation collaborative de modèles partagés, mais non leur édition. OneSpace [ONE] est un système commercial qui autorise l'édition collaborative. Certains systèmes favorisent les interfaces web, comme CyberCad [TR03], une application basée sur Java avec une communication intégrée multimédia.

Quelques systèmes CAO offrent des interfaces d'immersion en réalité virtuelle qui peuvent être utilisées pour la révision de conception [DHJ+00, Bro99]. En général un graphe de scène indépendant est utilisé, et les opérations de modélisation ne sont pas directement reflétées dans le modèle original CAO.

Comme observé dans [ARH02], "la collaboration pour la conception de produits demande des compétences considérablement plus larges que la plupart d'autres applications coopératives simples disponibles aujourd'hui" (dans cet article les auteurs discutent d'autres systèmes qui permettent une modélisation collaborative, dont la plupart employant des bases de données centralisées).

2.6.5 Communication vidéo

L'utilisation de la communication vidéo pour supporter la collaboration a beaucoup de possibilités au-delà de la visioconférence traditionnelle. Avec la disponibilité croissante de la mise en réseau à large bande, particulièrement à l'intérieur de réseaux d'entreprises, les approches maintenant restreintes à des domaines académiques tendent à gagner une nouvelle impulsion.

Une ligne de recherche est l'utilisation de la vidéo avec une emphase sur la communication informelle et extensible dans les arrangements sociaux. Un exemple intéressant est Mediaspaces [Mac99], dans lequel de multiples canaux vidéo et des mécanismes de contrôle sont utilisés pour essayer de fournir des types similaires d'interactions personnelles comme quand les gens sont en colloque.

Une importante perspective pour une coopération à distance dans des activités de modélisation est l'intégration de la vidéo dans l'interface de l'utilisateur à travers des arrangements d'affichage innovateurs, qui permettent le maintien de la direction du regard (*eye gaze*) pour une conscience spatiale améliorée [Ishi99]. Plusieurs groupes enquêtent actuellement sur l'utilisation de la vidéo stéréoscopique à haute résolution pour la téléprésence, dans le sens de donner aux utilisateurs à un site la sensation que

les gens à d'autres sites sont physiquement présents et interagissent avec des modèles partagés (par ex., l'Office of the Future Project [RWC+98, OOTF])

Dans tous les cas, une question fondamentale est la recherche d'une intégration améliorée des nouvelles formes de communication supportées par ces technologies avancées basées sur la vidéo avec les systèmes coopératifs de domaine spécifique.

Chapitre 3

Architecture de collaboration

3.1 Introduction

Dans ce chapitre, nous présentons l'architecture du système proposé, conçue pour supporter la collaboration distribuée synchrone dans le contexte décrit auparavant. L'architecture est basée sur l'intégration de modules d'extension (*runtime plugins*) avec l'application de modélisation de base, permettant sa transformation dynamique dans un système coopératif distribué.

Un plugin de collaboration, NetGocad, fournit la principale fonctionnalité de coopération à travers la redéfinition de certaines classes de l'application et l'introduction de mécanismes de distribution basés sur CORBA, permettant la diffusion d'information entre les participants d'une session de collaboration à travers des "canaux de collaboration" indépendants. La diffusion est soumise à des politiques basées sur les rôles définies avec un composant configurable. La solution fournit un mécanisme de contrôle simple et efficace qui n'affecte pas la performance graphique du système, une exigence basique dans le domaine d'application considéré.

Des canaux de communication multimédias sont fournis à travers un composant intégré, CSVTool. Des flux individuels peuvent être contrôlés par les participants, soumis à la politique établie pour la conférence en cours.

Un plugin de réalité virtuelle, Go2VR, compatible avec les mécanismes de collaboration proposés, permet la création de sessions de collaboration hétérogènes avec les participants RV. Dans ce cas, le schéma de coordination peut être utilisé pour le contrôle des canaux de collaboration selon le type d'interface utilisée par un participant.

Dans les prochains paragraphes, nous décrivons chacun de ces aspects du système, en commençant par les caractéristiques architecturales de support de l'application de base.

3.2 Architecture Gocad

Gocad est un logiciel de géomodélisation pionnier, développé à l'origine par un groupe de recherche supporté par un consortium international [GOC], qui permet la construction de modèles tridimensionnels pour les applications de géophysique, géologie et ingénierie de réservoir. Son architecture est basée sur l'utilisation systématique de Motifs de Conception (*Design Patterns*) [GHJV95] tels : Abstract Factory, Builder, Chain of Responsibility, Command, Composite, Factory Method, Interpreter, Iterator, Observer, Proxy, et Singleton. Il fournit aussi un environnement de développement souple pour permettre la création de plugins qui peuvent être chargés dynamiquement au temps d'exécution à l'intérieur de l'application [GDG].

Comme nous allons expliquer, l'intégration dynamique des mécanismes de collaboration proposés dans l'application principale est basée, en particulier, sur l'utilisation des motifs Command, Abstract Factory et Observer. La même approche peut être employée dans le cas d'autres applications qui les utilisent.

3.2.1 Motifs de Conception

Commande (*Command*)

"Encapsuler une demande comme un objet, vous laissant ainsi paramétriser les clients avec différentes requêtes, les mettre en file d'attente ou dans un journal, et supporter l'annulation d'opérations."

Chez Gocad, toute opération interactive est stockée dans une Commande, qui a une représentation comme une chaîne de caractères souple, traitée par un Interprète de Langage de Commande (CLI, *Command Language Interpreter*). L'interface de l'utilisateur distribue des demandes aux objets sans rien savoir concernant l'opération demandée ou le récepteur de la demande. Le motif Command transforme la demande elle-même en un objet, qui peut être stocké et passé comme d'autres objets.

L'utilisation du motif Command pour isoler le traitement d'opérations à partir de leur invocation à l'interface de l'utilisateur facilite amplement la création d'un mécanisme de diffusion. Un nouvel exécuteur pour les commandes peut être créé, remplaçant l'original à travers la redéfinition d'un Factory correspondant (voir ci-dessous). Le nouvel exécuteur se comportera ensuite comme suit: à part traiter la commande localement, il l'enverra à la conférence connectée (si les contrôles correspondants sont validés), qui est ensuite responsable de la rediffusion à tous les participants.

Fabrique Abstraite (*Abstract Factory*)

"Fournir une interface pour créer des familles d'objets associés ou dépendants sans spécifier leurs classes concrètes."

L'utilisation d'Abstract Factory permet la redéfinition de classes de l'application principale par le plugin. Avec de nouvelles fabriques installées, quand les objets

correspondants sont créés, ils seront des instances de la nouvelle classe redéfinie. Ce mécanisme est utilisé pour la création d'objets responsables de la diffusion de commandes d'événements graphiques. Le plugin de collaboration remplace les fabriques originales par des nouvelles, responsables de la création de :

- exécuteurs de commande; le nouvel exécuteur enverra automatiquement les commandes aux mécanismes de diffusion, avant leur exécution, comme décrit ci-dessus ;
- constructeurs de caméras; le nouveau constructeur construira des caméras équipées de fonctionnalité de collaboration (nouveaux observateurs attachés, responsables de la diffusion d'événements graphiques, comme décrit ci-dessous; des primitifs graphiques et des outils de manipulation pour les annotations et des curseurs 3D; des barres d'outils d'annotations; etc.)

Observateur (*Observer*)

"Définit une dépendance de type un à plusieurs entre des objets, de sorte que lorsqu'un objet change d'état, tous ses dépendants sont notifiés et actualisés automatiquement."

L'utilisation de ce modèle pour la gestion d'événements permet l'annexation par le plugin de nouveaux observateurs pour les caméras redéfinies, qui seront ensuite en charge de la diffusion d'événements graphiques (par ex., des points de vue de caméra et des mouvements de télépointeur) à des conférences à distance connectées. Chaque fois que la méthode d'actualisation des nouveaux observateurs est exécutée, elle vérifiera aussi l'ensemble de contrôles définis des par la logique de coordination pour le canal correspondant, avant d'envoyer des événements pour le traitement local ou au partenaire connecté (voir 3.3.3).

3.2.2 Modules d'extension

L'environnement de développement Gocad fournit des outils pour la création de modules d'extension (*plugins*) dérivés de l'application originale (Figure 3.1). Un nombre de plugins peut être indépendamment chargé au temps d'exécution avec l'application principale, dans le but d'y ajouter des caractéristiques spécifiques (objets neufs ou dérivés, algorithmes, commandes, graphiques, éléments de l'interface utilisateur, etc.). Le développement est fait indépendamment de la distribution originale [GDG].

Une nouvelle version de l'arbre de développement (un "arbre parallèle") est créée avec les sous-directoires nécessaires, contenant tous les binaires, bibliothèques, fichiers *include*, source, *makefiles* et ressources supplémentaires pour la création du plugin. Sur cette structure de nouvelles fonctionnalités peuvent être définies, et un nouvel exécutable (*main*) généré.

Par exemple, comme nous l'avons analysé au chapitre 1, différents plugins peuvent être utilisés d'une façon intégrée dans le cadre de ce travail pour le géoguidage collaborative (Figure 3.2):

- NetGocad, responsable de la fonctionnalité principale de collaboration (paragraphe 3.3);
- Go2VR [GO2], responsable d'ajouter une interface de réalité virtuelle, adaptée à la compatibilité avec NetGocad (paragraphe 3.5);
- gWlog, responsable de l'acquisition de données en temps réel d'un site de forage à distance (paragraphe 3.6);

En outre, d'autres plugins existants peuvent être directement utilisés, comme IDP [IDPa], pour ajouter une fonctionnalité avancée de planification de puits.

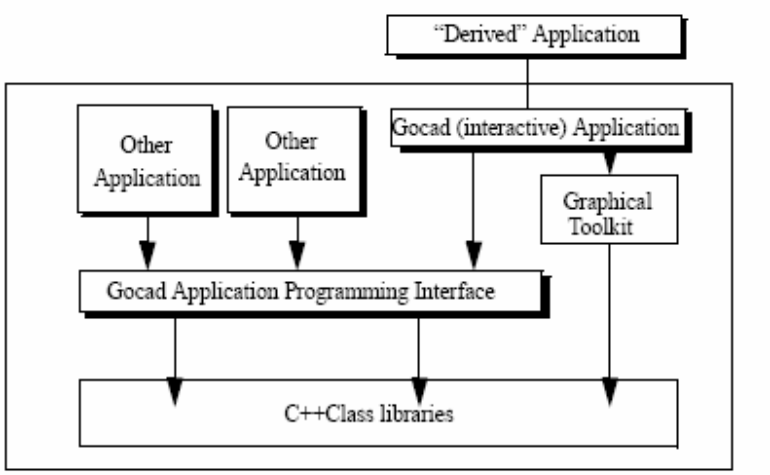


FIG. 3.1 – Plugin Gocad: "Derived" Application (Application "Dérivée"). [GDG]

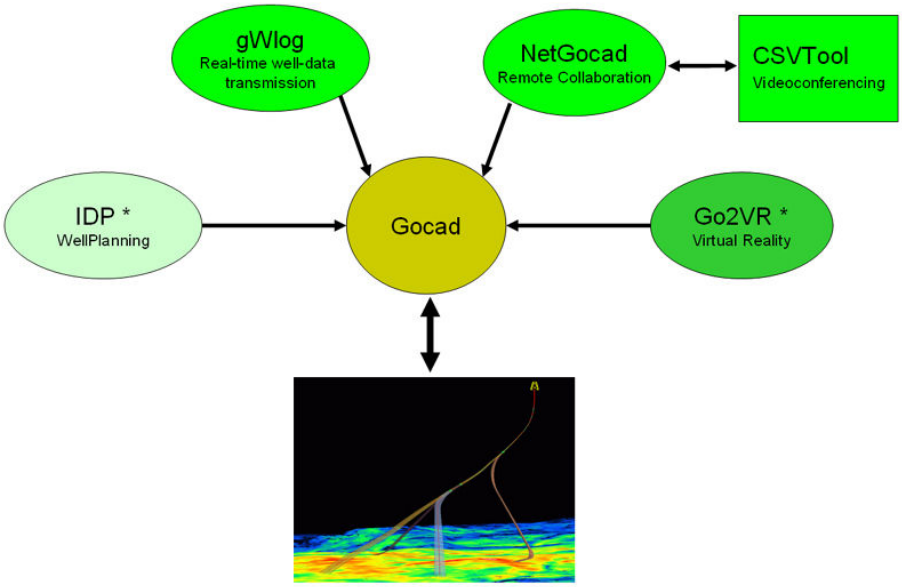


FIG. 3.2 – Exemple de composition de plugins pour le géoguidage collaborative (configuration à un site).

3.3 Architecture NetGocad

NetGocad est le plugin responsable de l'introduction dans l'application de base des mécanismes analysés au paragraphe 3.2.1, ainsi que des éléments d'interface utilisateur (voir Appendice 1), et accès aux outils intégrés comme CSVTool et Go2VR.

La première version de NetGocad, développée en collaboration avec Fabien Bosquet et commercialisé pour Earth Decision Sciences, permet des connexions face à face entre des utilisateurs à distance. La seconde version présentée ici rajoute à celle-ci: des compétences de conférence multi-participants, une coordination basée sur des rôles, une communication intégrée multimédia, et une intégration avec le plugin de réalité virtuelle.

La Figure 3.3 montre un diagramme simplifié avec les classes principales de NetGocad. La fonctionnalité de collaboration principale est fournie par une bibliothèque d'objets CORBA (GMeeting). Le standard CORBA (Common Object Request Broker Architecture [HV99, PR00]) a été choisi comme technologie de distribution d'objets pour les raisons suivantes :

- transparence de système d'opération: CORBA est un standard multiplateformes, une exigence essentielle dans notre contexte;
- transparence d'architecture: les particularités d'architectures CPU sont cachées des développeurs ;
- services flexibles: des services de haut niveau sont disponibles, comme Service de Nommage (*Naming Service*), Service d'Evénements (*Event Service*), Service de Notification (*Notification Servic*e), Service de Sécurité (*Security Service*);
- intégration avec des solutions d'interopérabilité utilisées dans l'industrie, en particulier avec Open Spirit [OS], utilisé comme plateforme d'intégration de données (paragraphe 1.2);
- transparence de langage: les objets peuvent être implémentés dans n'importe quel langage, beaucoup d'attachements établis sont disponibles pour C++ et Java, les langages d'implémentation que nous utilisons.

La classe centrale dans cette architecture est le Participant, qui agit comme principal serveur pour les demandes de clients à distance, invoquant des opérations dans l'instance locale de l'application.

Une Conference, créée par le Manager, garde la trace d'un groupe de Participants. Elle fournit la même interface pour les événements de canal que les Participants, mais elle les diffuse à ses membres. La classe abstraite Partner (partenaire) agit comme une superclasse pour des Participants et pour la Conference.

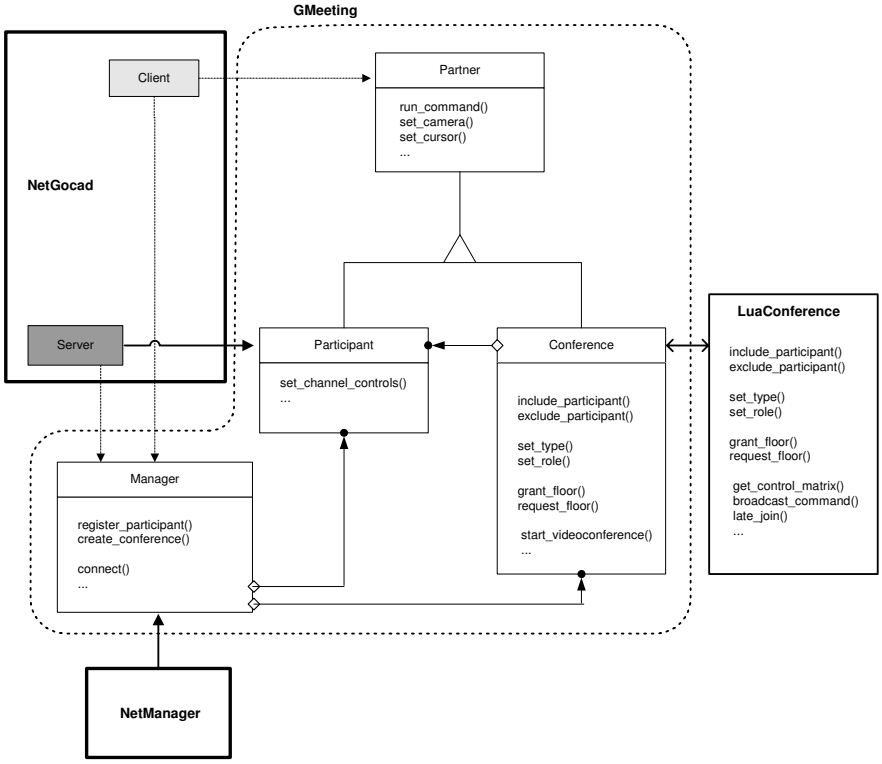


FIG. 3.3 – Classes principales NetGocad (simplifié).

Les classes principales CORBA de la bibliothèque GMeeting sont décrites ci-dessous. Un résumé de l'interface pour les classes montrées (définitions IDL) est présenté dans l'Appendice 2.

Partner (Partenaire): Déclare l'interface pour les méthodes qui traitent des événements du canal de collaboration réellement implémentés par les Participants et les Conférences. Les clients gardent les références d'un Partner, et par conséquent ils n'ont pas besoin de savoir le type de Partner avec lequel ils sont réellement connectés (qui pourrait être une conférence ou un participant individuel).

Participant (Participant): Implémente des méthodes pour traiter les événements des canaux de collaboration en invoquant les opérations appropriées à l'application hôte (exécution de commandes, réglage de la position de la caméra, etc.).

Conference (Conférence): Gère un groupe de Participants et lui diffuse les événements de collaboration. Garde aussi la trace des *floors*: si un canal est contrôlé, le participant contrôlant le *floor* est le seul autorisé à envoyer des événements à travers ce canal, comme décrit aux paragraphes suivants.

Manager (Manager): Gère tout le système de conférence. Pour un domaine donné (une société, par exemple), il est responsable de l'inscription et du listage des participants, de la création et du listage des conférences, et permet aux clients de se connecter à des participants et à des conférences à distance. C'est l'unique objet publié, instancié par un daemon, et sert de point d'entrée pour la création de sessions de collaboration.

LuaConference, Client et Server sont aussi des classes CORBA.

LuaConference (LuaConférence): est un module implémenté dans le langage d'extension Lua ([IFC96, Ier03], voir aussi Appendice 3), créé par chaque Conference au temps d'exécution et responsable des politiques de coordination. Il charge les définitions des types de collaboration et des rôles des participants, et interagit avec la Conference pour fournir des services configurables (gestion de session, contrôle d'utilisation, définition des types de conférences et des rôles des participants). L'implémentation de ce module est analysé plus loin au paragraphe 3.3.5.

Client (Client): classe Singleton [GHJV95] implémentée dans C++. Instancié par l'application, il fournit des méthodes pour se connecter avec le Manager et avec des Partners, gardant les références avec eux. Lorsqu'un utilisateur réalise une opération localement (exécute une commande, un mouvement de caméra, etc.), un observateur correspondant (redéfini par le plugin pour l'application) utilisera l'instance Client pour invoquer la même opération sur le Partner auquel il est connecté.

Server (Serveur): classe Singleton implémentée dans C++ utilisée par l'application pour démarrer l'ORB, instancier un Participant et l'inscrire avec le Manager

3.3.1 Fonctionnement

Pour qu'une session de collaboration prenne place, un processus daemon (NetManager) instanciant un Manager doit être en marche (Figure 3.4). Lorsque NetGocad est lancé, il instancie un Server et un Client, et est connecté au Manager lorsque l'application est "enregistrée" par l'utilisateur (rendue disponible pour la collaboration).

Lorsqu'un groupe de gens veut collaborer, d'abord quelqu'un crée une conférence (à travers le Manager), en choisissant un type de collaboration parmi ceux disponibles (chargés par LuaConference), et il devient automatiquement le propriétaire de la conférence. Ensuite les autres participants appellent la conférence (aussi à travers le Manager), en choisissant l'un des rôles disponibles pour le type de conférence actuel (paragraphe 3.3.4). Si le propriétaire accepte l'appel, l'appeleur est invité à se joindre et reçoit les droits sur les canaux de collaboration selon le rôle choisi (voir Figure 3.5). Après, chaque fois qu'un participant envoie un événement à la conférence sur un canal de collaboration, il est diffusé à tous les membres capables de le recevoir. Différents mécanismes de diffusion sont utilisés, selon le type de canal, comme analysé aux paragraphes suivants. Si quelqu'un démarre une communication audio ou vidéo, les

flux appropriés sont créés, selon la politique en place, à travers l'outil de visioconférence (paragraphe 3.4).

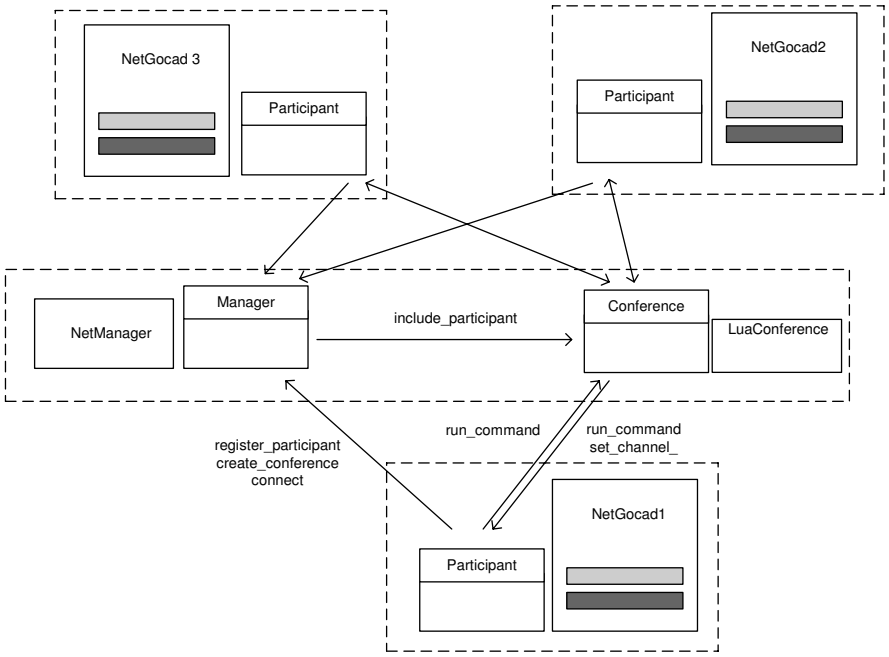


FIG. 3.4 – Opération NetGocad, avec des exemples de communication entre des objets distribués (les principales méthodes sont énumérées à l'Appendice 2). Des rectangles pointillés représentent les ordinateurs distincts (le Manager et la Conference peuvent également être en marche sur l'un des sites des Participants).

Pour la distribution initiale du modèle répliqué, deux stratégies sont possibles: une transmission préalable de projets par des moyens externes, et ensuite un chargement local par chaque participant, ou une transmission directe à travers NetGocad (soit des projets entiers ou d'objets individuels). Pour de grands projets, une transmission externe est plus efficace, et pourrait être faite soit à travers FTP ou, de préférence, à travers l'outil 'rsync' (qui facilite la synchronisation des fichiers à distance). La transmission à travers NetGocad est faite à travers CORBA, et par conséquent a un coût supplémentaire de rassemblage de données. Elle devrait être utilisée plutôt pendant les séances de travail pour la retransmission d'objets en cas de besoin.

3.3.2 Canaux de collaboration

Nous définissons “canal de collaboration” comme une voie abstraite pour véhiculer les informations parmi les instances de l'application. Actuellement les canaux gérés sont: commandes, positions de caméra, télépointeurs, annotations, avatars, audio et vidéo. Les contrôles d’utilisation (*floor controls*) peuvent être établis pour n’importe quel canal d’une manière homogène (paragraphe 3.3.4). Cependant, les exigences particulières de différents types d’informations échangées nous mènent à la définition de méthodes de diffusion et de contrôle particulières pour différentes classes de canaux.

La plupart des canaux correspond à une méthode spécifique, déclarée dans la classe Partner et implémentée par les Participants, responsable du traitement d’événements et du transport de leurs effets dans l'application hôte (exécution d’une commande, déplacement d’une caméra ou d’un télépointeur, etc.). Des exceptions sont l’audio et la vidéo, qui sont actuellement traités uniquement par un module séparé (bien que conceptuellement l’audio et la vidéo pourraient aussi être gérés par les objets du Participant, par exemple pour le rendu des flux vidéo à l’intérieur d’une caméra 3D).

Normalement certains canaux devraient utiliser le contrôle d’utilisation alors que d’autres non: chacun dans une session peut être amené à utiliser des annotations en même temps mais pour éviter les inconsistances, les commandes de modélisation devraient être contrôlées par défaut. Dépendant du type de conférence, les *floors* peuvent aussi être souhaitées pour une communication audio et vidéo.

Le traitement du contrôle d’utilisation est indépendant du type de canal. La définition de la logique de contrôle est faite dans le module LuaConference, où tous les canaux sont traités comme des éléments d’une série homogène. Par défaut si un canal est sous contrôle, le *floor* peut être passé par le contrôleur actuel ou par le propriétaire de conférence à tout moment (cependant la stratégie de contrôle peut être aisément redéfinie, étant donné qu’elle est implémentée dans Lua et interprétée en temps d’exécution). Des commutateurs de contrôle et des matrices sont utilisés pour la définition des droits basés sur les rôles, sur chaque canal, comme analysé au paragraphe 3.3.3. Pour l’implémentation de mécanismes de diffusion et de contrôles, nous sous-divisons les canaux en trois classes: *canaux de commande, canaux d’interaction graphique* et *canaux de flux*.

Il vaut la peine de remarquer que les canaux de flux ont la particularité que les flux individuels et les contrôles de l’interface-utilisateur sont fournis par la communication entre chaque paire de participants, dans un schéma “nxn” (à l’interface de visioconférence il y a une fenêtre pour chaque partenaire à distance, avec des contrôles individuels audio et vidéo, comme montré Fig 3.13, paragraphe 3.4.4.3). Pour les canaux restants, un schéma “1xn” est utilisé. Ceci est renvoyé dans l’interface de l’utilisateur: pour chaque canal, le Tableau de Contrôle de Conférence (Figure 3.5) permet seulement de définir si un participant est en train d’envoyer une information aux autres ou non.

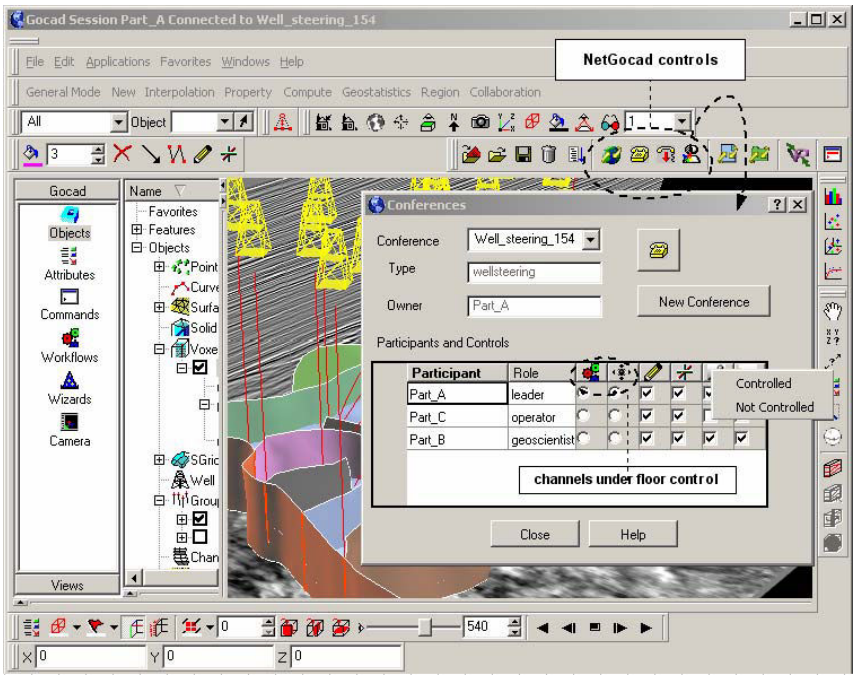


FIG. 3.5 – Tableau de contrôle de conférence. Les canaux pour chaque participant sont affichés et peuvent être manipulés. Les canaux sous contrôle d'utilisation (*floors*) sont représentés par des boutons de radio, les canaux libres avec des boîtes de contrôle (*check boxes*). Le propriétaire de conférence a accès aux contrôles de tous les participants (comme montré sur l'image), et peut ainsi prendre des actions préférentielles. Le propriétaire peut aussi établir ou enlever des *floors* pour chaque canal en cliquant sur l'icône du canal (comme montré pour le canal audio). Pour les canaux sous contrôle d'utilisation le contrôleur actuel (ou propriétaire) peut passer le *floor* à n'importe qui. D'autres participants ne peuvent manipuler que leurs propres contrôles permis selon l'état de coordination; les autres boutons sont immobilisés.

Bien qu'actuellement seuls audio et vidéo soient traités comme canaux nxn, en principe tout canal pourrait être géré de cette manière, pour permettre, par exemple, des sessions de travail parallèles en sous-groupes (*coteries*). Dans ce cas une interface d'utilisateur convenable devrait être fournie et l'utilisation de ce schéma devrait être étudiée.

3.3.2.1 Canaux de commandes

Les canaux de commandes portent les commandes de l'application, exécutées localement et envoyées à la Conference pour diffusion. Un nouvel *exécuteur de commandes* (pragraphe 3.2.1) installé par le plugin de collaboration gère les commandes générées par l'utilisateur local de la manière suivante: si le contrôle *local* correspondant à la commande est validé, la commande est exécutée localement. Si le contrôle *out* est validé, elle est envoyée à la Conférence pour diffusion. Les commandes reçues sont exécutées si le contrôle *in* est validé. Ce schéma de contrôle est analysé plus loin au paragraphe 3.3.3.

Des balises avec le nom de l'émetteur sont toujours ajoutées aux commandes diffusées, de sorte que les commandes arrivant de partenaires éloignés puissent être identifiées par l'exécuteur et gérées en conséquence (seul le contrôle *in* est vérifié dans ce cas, et elles ne sont pas rediffusées).

La Conference envoie toujours les commandes diffusées au module de coordination (LuaConference) où elles sont enregistrées pour permettre une association tardive (*late join*): toutes les commandes enregistrées peuvent être envoyées par la Conference aux nouveaux participants rejoignant la session, si demandé à travers une opération définie dans le menu de collaboration (Appendice 1).

Les commandes sont classées dans un fichier de configuration associant des listes de commandes à des étiquettes. Une classe mandataire est "donot_broadcast", définissant toutes les commandes (en format de chaîne CLI) qui ne devraient pas être diffusées (enregistrement, connexion, sortie, transfert de modèle, création d'outils d'interaction, etc.), mais uniquement exécutées localement.

Ce mécanisme peut aussi être utilisé pour la classification de commandes en sous-canaux logiques séparés, de sorte que chacun d'eux puisse être soumis à un contrôle d'utilisation indépendant. Ceci est fait actuellement avec des annotations tridimensionnelles qui, bien que similaires en effet à d'autres canaux graphiques d'interaction, sont en fait générées par l'application comme commandes uniquement quand la création interactive d'une primitive est complète (par conséquent le processus de dessin des primitives n'est pas renvoyé aux autres sites, uniquement leur effet final). De cette manière, les annotations sont traitées comme un canal indépendant qui peut être utilisé librement, même si le contrôle d'utilisation est établi pour les commandes restantes.

De nouvelles classes de commandes peuvent aussi être créées, par exemple avec des commandes associées à la manipulation de paramètres de visualisation, de sorte qu'elles peuvent aussi être utilisées librement par les participants alors que la modélisation reste sous contrôle d'utilisation. L'analyse syntaxique de la commande pourrait aussi être utilisée pour la spécification de *floors* sur des objets spécifiques et outils basés sur l'identification des commandes de modélisation qui les manipulent (similairement à ce qui est fait dans DCWPL avec l'utilisations de tables de possession, "*own tables*").

En termes de communication sous-jacente, les canaux de commandes demandent une transmission fiable, et devraient être transmis à travers le protocole TCP.

Les annotations tridimensionnelles méritent une certaine discussion, étant donné qu'elles sont essentielles pour faciliter une coopération efficace, permettant l'indication de caractéristiques directement sur le modèle tridimensionnel, utilisé comme support pour dessiner et pour le placement d'annotations primitives comme les flèches et le texte. Des outils similaires à ceux trouvés dans les interfaces de peinture habituelles sont disponibles. Une attention spéciale est donnée pour garantir la visibilité et le bon placement de chaque type de primitive. Par exemple, les lignes sont créées réellement comme des tubes 3D en grisé. Le dessin peut être fait juste en déplaçant la souris, comme avec un outil "main libre", et est très rapide (parce que la valeur z à chaque pixel pertinente est obtenue directement du tampon-z). Les annotations n'affectent pas le modèle, et peuvent immédiatement être effacées et recréées.

3.3.2.2 Canaux d'interaction graphique

Les canaux d'interaction portent les événements qui sont générés par une manipulation directe à l'intérieur de la caméra 3D (mouvements de caméra, télépointeurs, avatars), et qui ont besoin d'être traités à des rythmes élevés pour l'interactivité graphique lisse. Des curseurs tridimensionnels (télépointeurs) identifiant leurs utilisateurs par une couleur ou une étiquette permettent aux participants de pointer les objets partagés. Des avatars peuvent être utilisés pour afficher la position de la caméra et le *frustum* (cône tronqué de visualisation) d'un participant avec un point de vue non synchronisé (Figure 3.6).

Ces événements ne sont pas passés par la LuaConference. Les événements sont envoyés à la Conference pour diffusion par les *observateurs* installés par le plugin de collaboration (paragraphe 3.2.1), après avoir vérifié l'ensemble des contrôles pour les canaux (comme décrits ci-après pour les commandes, et comme analysé au paragraphe 3.3.3).

Une observation est que, comme des caméras multiples de types différents peuvent être créées à chaque site (3D, 2D, Sections), actuellement nous adoptons la stratégie suivante: les événements d'interaction ne sont échangés qu'entre des caméras ayant la même étiquette. Ceci permet aux participants d'ouvrir des caméras secondaires qui ne sont pas synchronisées avec la principale, mais uniquement avec les caméras correspondantes ouvertes par d'autres participants, qui reçoivent automatiquement la même étiquette selon l'ordre dans lequel elles sont créées. Un schéma consistant d'appellation est utilisé par NetGocad pour la création de nouvelles caméras, de sorte que, quand une caméra est fermée, son identification est réutilisée par la caméra suivante du même type ouverte. Une alternative possible pour ce schéma serait la création de multiples canaux de caméras indépendants, mais nous pensons que ceci compliquerait inutilement l'interface de l'utilisateur.

Le taux de diffusion des événements graphiques devrait être limité par des minuteurs (actuellement nous utilisons une limite fixée à 20 diffusions par seconde, mais ce paramètre pourrait être ajustable selon les conditions du réseau).

Comme la transmission doit être rapide mais pas nécessairement fiable (des positions abolues sont utilisées, ainsi les messages perdus ne sont pas importants), UDP est le protocole préféré (voir paragraphe 2.5.6).

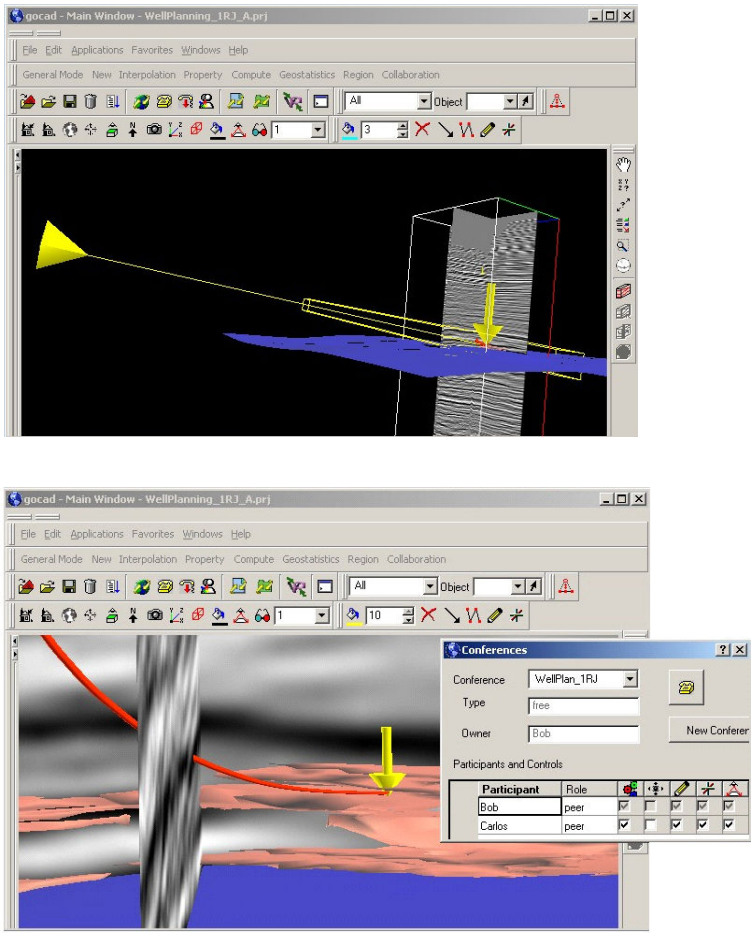


FIG. 3.6 – Télépointeur et avatar. Le participant du dessus voit le télépointeur jaune (flèche verticale) manipulé par celui du dessous, et aussi un avatar indiquant son point de vue (un simple cône à gauche représentant le point de vue, plus le frustum actuel). L'utilisateur du dessous ne peut pas voir un avatar parce que son frustum est contenu dans celui de l'utilisateur du dessus.

3.3.2.3 Canaux de flux

Les flux audio et vidéo sont en fait gérés par un composant intégré (l'outil CSV, comme décrit en détails au paragraphe 3.4) avec lequel NetGocad communique à travers un protocole CORBA. Comme les autres canaux, les canaux de flux sont soumis aux politiques de contrôle basés sur les rôles (paragraphe 3.3.4) et peuvent être mis sous contrôle d'utilisation à travers le Tableau de Conférence (Figure 3.5). En

raison des exigences spécifiques de mise en réseau qu'ils demandent, les flux audio et vidéo sont traités de poste à poste en utilisant le protocole RTP.

3.3.3 Contrôle de canaux

Une hypothèse basique pour la modélisation et l'implémentation des politiques de contrôle de collaboration est que les aspects de coordination du système devraient être séparés des aspects informatiques, comme préconisé dans d'autres schémas de collaboration comme DCWPL [CM96] et COCA [LM98]. Ceci accorde de la souplesse pour l'ajustement du système, permettant la modification aisée des politiques de coordination sans affecter les autres modules.

Dans notre architecture, la logique de coordination est implémentée avec le langage d'extension Lua dans le module LuaConference, créé par la conférence au temps d'exécution. Le modèle de contrôle adopté est inspiré de COCA [LM98] (voir paragraphe 2.5.3), dans le sens où nous utilisons aussi les notions de canaux et portes ("commutateurs" de contrôle), mais avec quelques différences essentielles:

- les services de session et de contrôle d'utilisation sont fournis par défaut (mais peuvent être redéfinis);

- au lieu d'utiliser des règles basées sur la logique (comme dans COCA) ou des programmes déclaratifs (comme dans DCWPL) pour définir les politiques de contrôle – qui sont puissantes et souples mais peuvent devenir difficiles à écrire et adapter par des programmeurs non compétents – nous avons créé une syntaxe simple orientée objet pour la spécification des droits de contrôle basés sur les rôles sur les canaux ;

- comme la performance graphique est un problème principal, pour les canaux d'interaction le mécanisme de contrôle d'utilisation replace le schéma défini dans COCA (basé sur le traitement de tous les événements à travers une machine virtuelle externe à l'application) avec le contrôle de commutateurs d'entrée et de sortie réglé par le composant de coordination pour chaque canal directement à l'application. Ces contrôles sont ensuite vérifiés par les observateurs installés par le plugin avant que les événements ne soient traités ou diffusés. Passer tous les messages traversant les canaux d'interaction à travers une machine virtuelle externe aurait un impact sur la réactivité graphique du système.

Comme analysé au paragraphe précédent, un canal de collaboration peut être de deux types basiques : "1xn" (un à plusieurs) ou "nxn" (plusieurs à plusieurs). Nous utilisons deux types de mécanismes pour leur contrôle (Figure 3.7):

- des commutateurs booléens de contrôle (*out*, *in* et *local*) pour chaque canal, stockés à l'objet du participant;
- matrices de contrôle nxn (*K_out, K_in, I_out, I_in*) pour chaque canal nxn (actuellement uniquement audio et vidéo).

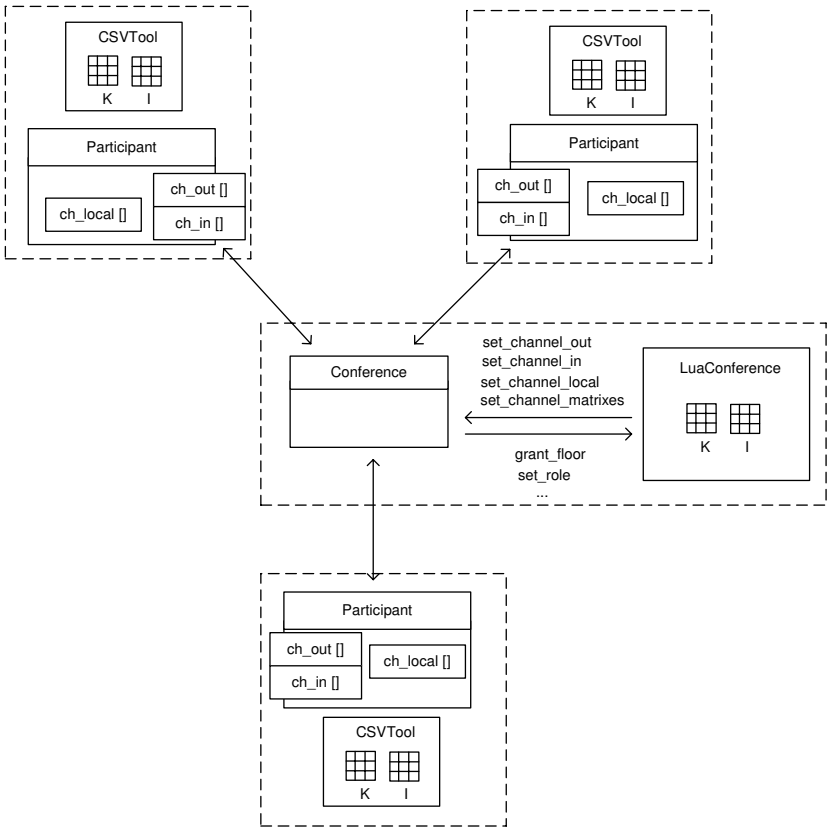


FIG. 3.7 – Mécanismes de contrôle de canaux: commutateurs pour les canaux 1xn et matrices pour les canaux nxn (voir aussi Listages 3.2 et 3.3 pour une description complète des interfaces entre la Conference et la LuaConference)

Pour les canaux multimédia, des flux de poste à poste sont créés parmi les participants, donc toute la matrice de connectivité doit être spécifiée. Pour les autres canaux, les événements sont soit envoyés à tous les autres participants ou à aucun, et ils sont également reçus soit de tous les autres participants ou d'aucun.

Cependant, pour la spécification des droits de contrôle basés sur les rôles pour chaque type de conférence, nous utilisons un schéma uniforme pour tous les canaux, basé sur la définition des entrées en matrices de contrôle Key et Intention (paragraphe 3.3.4).

Matrices de contrôle

Dans une conférence avec n participants, pour chaque canal nxn, trois types de matrices booléennes sont utilisées pour spécifier l'état de connectivité : *Key* (Clé), *Intention* et *Connection* (Connexion). Pour les canaux audio et vidéo, ces matrices contrôlent l'état de l'outil de visioconférence (see paragraphe 3.4).

Les matrices Key sont utilisées pour contrôler les droits permanents de chaque participant pour envoyer (*K_out*) et recevoir (*K_in*) l'information d'un autre. Logiquement, invalider une entrée de Key est équivalent à éteindre le dispositif nécessaire pour envoyer ou recevoir les événements associés. Les matrices Intention sont utilisées pour contrôler les intentions actuelles des participants d'envoyer (*I_out*) et recevoir (*I_in*) les informations (à condition qu'ils aient ce droit). Les matrices Connection ne sont pas directement spécifiées par les utilisateurs, mais saisies selon les valeurs Key et Intention. Une connexion est établie lorsque les Keys et les Intentions d'envoi et de réception des participants en corrélation sont vraies. C'est-à-dire, pour chaque position dans la matrice de connexion,

$$C_{ij} = (K\_out_{ij} \wedge I\_out_{ij}) \wedge (K\_in_{ji} \wedge I\_in_{ji})$$

La spécification des droits de chaque *rôle* pour envoyer/recevoir des événements à/de chaque autre *rôle*, pour chaque type de collaboration, est faite comme décrite au paragraphe suivant. Ensuite les matrices *n*x*n* pour les participants sont assemblées par la LuaConference lorsqu'ils entrent et quittent la session, selon leurs rôles, et communiquées à l'application par la Conference. Les intentions peuvent être directement altérées par les participants à travers les contrôles de l'interface utilisateur (si validés, selon les valeurs des clés).

Commutateurs de contrôle

A chaque objet Participant, les commutateurs *out*, *in* et *local* sont spécifiés pour chaque canal 1xn. Les commutateurs sont réglés initialement selon les rôles des participants et la spécification de collaboration et peuvent être changés par la LuaConference pour imposer des politiques de contrôle d'utilisation, ou par les utilisateurs (si les contrôles sont validés).

Le commutateur *local* est vérifié pour un canal avant que les événements ne soient traités localement; le commutateur *out* avant que les événements ne soient diffusés à un partenaire; et le commutateur *in* avant qu'un événement reçu ne soit passé au traitement local.

Pour l'établissement d'un *floor* sur un canal, le module de coordination allume les commutateurs *out* et *local* du détenteur du *floor*, éteint les commutateurs *out* et *local* de n'importe qui d'autre, et allume les commutateurs *in*. Habituellement, si un commutateur *out* est allumé, le commutateur *local* le sera aussi (sinon les actions exécutées par l'utilisateur seront répliquées à distance, mais non localement). Allumer le commutateur *local* et éteindre le commutateur *out* permet un travail privé (mais pour la modélisation, le contrôle de version aurait ensuite besoin d'être géré).

Les réglages définis par la LuaConference passent toujours à travers l'objet Conference, étant donné qu'elle ne communique pas directement avec les Participants. Pour les canaux libres, les utilisateurs peuvent spécifier directement leurs intentions à travers la boîte de vérification à l'interface de l'utilisateur de conférence, si les contrôles sont validés. Les commutateurs *in* ne sont actuellement pas manipulables à travers l'interface de l'utilisateur, mais peuvent être spécifiés à la configuration de la conférence (par exemple comme dans le listage 3.1, où les caméras des utilisateurs RV ne sont pas validées pour recevoir des événements de position de caméra d'autres utilisateurs).

Le schéma de spécification est basé sur les réglages d'entrées dans les matrices Key et Intention basées sur les rôles (voir paragraphe suivant). Comme les canaux 1xn sont contrôlés à travers des commutateurs au lieu de matrices, uniquement les spécifications du type "role:all" (rôle:tous) sont valables, les autres sont ignorées. Les réglages pour une entrée de K_out fausse pour un canal éteint ce commutateur définitivement pour le rôle spécifié, et invalide le contrôle d'interface correspondant. Une entrée dans la matrice Intention règle une valeur initiale, mais laisse le contrôle actif.

3.3.4 Spécification de collaborations

En utilisant les mécanismes du langage Lua, spécialement les tables (voir Appendice 3), nous avons défini une syntaxe orientée objet simple pour la spécification des types de conférences, les rôles et les droits des participants sur d'autres canaux, basé sur la définition des matrices Key et Intention basées sur les rôles, pour chaque type de canal.

La spécification des politiques de contrôle est faite pour chaque type de *collaboration* (le listage 3.1 définit une collaboration de guidage de puits simplifiée). Les collaborations contiennent des *roles* et des *channels*, pour lesquels des *floors* peuvent être réglés. Une gestion de session et des services de *floors* sont fournis comme méthodes prédéfinies pour la classe Collaboration dans l'implémentation de module LuaConference, mais peuvent être redéfinis (Listage 3.5).

Pour chaque canal d'un type de collaboration, les droits permanents de chaque rôle d'envoyer/recevoir d'autres rôles sont spécifiés à travers les matrices Key. Un exemple détaillé est analysé au paragraphe 3.4 pour les canaux audio et vidéo. De fausses valeurs invalident une connexion et le contrôle d'interface correspondant. Par défaut tous les canaux sont ouverts (les valeurs vraies dans la matrice n'ont pas besoin d'être spécifiées, uniquement les connexions bloquées doivent être indiquées avec les valeurs fausses).

Les matrices Intention sont utilisées pour la définition des intentions actuelles sur l'utilisation d'un canal, et par défaut, sont aussi remplies avec les valeurs vraies. Les entrées fausses spécifient les connexions initialement bloquées, mais le contrôle correspondant reste actif (par exemple, pour les canaux audio et vidéo, les entrées fausses seraient utilisées pour spécifier qu'uniquement les flux souhaités devraient être créés au début d'une visioconférence entre les participants, selon leurs rôles).

Les canaux contrôlables (ceux avec des réglages pouvant être changés par les participants et pour lesquels les *floors* peuvent être établis pendant la conférence) sont définis; tous les autres restent libres (sans contrôle d'utilisation, et n'apparaîtront pas à l'interface de contrôle de conférence). Les rôles par défaut pour le propriétaire de la conférence (le participant qui a démarré la conférence) et pour les nouveaux participants sont aussi définis.

Par exemple, laissez-nous montrer une définition simplifiée pour une session de guidage de puits, avec quelques rôles et politiques de base. Dans cet exemple (listage 3.1), les utilisateurs RV ne peuvent pas envoyer ou recevoir des événements de caméra à/de quiconque, et les opérateurs ne reçoivent ni n'envoient des flux vidéo au début de la session (mais ils peuvent les recevoir après, si souhaité, étant donné que les clés ne sont pas fausses).

```
 -- relie les noms de canaux aux index de canal d'application
appl_channels = {cmd=0, cam=1, annot=2, cursor=3, avatar=4, audio=5, video=6}
channel_type = {cmd=1, cam=1, annot=1, cursor=1, avatar=1, audio=n, video=n}

 -- définе un type de Collaboration
 WellSteering = Collaboration {
      type = "wellsteering",

      -- canaux contrôlables
       channels = {"cmd", "cam", "annot", "cursor", "audio", "video"},

     -- canaux initialement sous contrôle d'utilisation (floor control)
     floors = {"cmd", "cam"},

      -- défine les rôles
      LeadGeoscientist = Role {type = "leader"},
      Geoscientist = Rôle {type = "géoscientiste"},
      VrGeoscientist =  Rôle {type =  "vr"},
      OnSiteOperator = Rôle {type = "opérateur"},

      -- défine les matrices basées sur les rôles pour chaque canal (toutes les autres
      -- paires de rôles, pour tous les canaux, sont vraies par défaut
      K_in = { ["cam"]= {["vr:all"]=false} },
      K_out = { ["cam"]= {["vr:all"]= false } },
      I_in= { ["video"]= {["operator:all"]= false } },
      I_out= { ["video"]= {["operator:all"]= false } },

      -- rôle par défaut assumé par le propriétaire
      default_owner_role = "leader",
       -- rôle par défaut pour les autres participants
      default_participant_role = "geoscientist",
 }
  -- Obs: K_out, K_in, I_out, I_in sont vrais pour toutes les autres paires de rôles,
 --          pour tous les canaux
```

LISTAGE 3.1 – Définition d'une collaboration de guidage de puits.

3.3.5 LuaConférence

```
-- Gestion de séance
function include_participant(participant, role)
function exclude_ participant (participant)

-- Floor
function grant_floor(ch, participant, requester)
function request_floor(ch, demandeur)
function get_floor_controller(ch)

-- Régulation
-- définir le type de collaboration, les rôles,  etc.
function set_type(type)
function get_type()
function get_collaboration_types()
function init_channels()
function set_role(participant, role)
function get_role(participant)
-- retourner à la liste de rôles pour la collaboration
function get_collaboration_roles(collab_type)

-- Enregistrement de commandes
 function log_command(commande, sender)

-- Contrôle de canaux
-- rassemble la matrice des participants nxn pour le  canal, étant donné
-- la matrice de contrôle basée sur les rôles M  (one of K_out, K_in, I_out, I_in)
function get_control_matrix(M, ch)

-- Association tardive
 function late_join(participant)
```

LISTAGE 3.2 – Quelques méthodes définies dans LuaConférence, et utilisées par la Conférence.

L'interface fournie par LuaConfé\erence à la Conference de l'application hôte reflète les services fournis par ce module: gestion de session, contrôle d'utilisation et de canal, règlement des rôles des participants et types de conférences, association tardive.

L'appel des méthodes LuaConference par la Conference (non détaillé ici, voir [Lua] et Appendice 3) suit le protocole conventionnel pour Lua, utilisant une pile auxiliaire (parce que Lua est un langage dynamiquement typé avec ramasse-miette). Ci-dessus nous montrons quelques-unes des méthodes principales (Listage 3.2).

Dans la direction opposée, l'interface fournie par la Conference à LuaConference permet, par exemple, de manipuler les commutateurs de contrôle des Participants, ou d'envoyer des commandes enregistrées à des associés tardifs. Ces fonctions doivent être préalablement enregistrées dans l'environnement Lua, et les arguments et les résultats sont aussi passés à travers une pile.

```
// Contrôle de canaux
static int set_channel_out(lua_State* L);
static int set_channel_in(lua_State* L);
static int set_channel_local(lua_State* L);
static int set_channel_matrixes(lua_State* L);

// Invocation de commande (retardataire)
static int run_command(lua_State* L);
```

LISTAGE 3.3 – Quelques méthodes de Conférences accédées par LuaConférence

Les méthodes LuaConference peuvent être redéfinies en Lua, d'une manière orientée objet. Laissez-nous considérer comme exemple la définition de collaboration pour une session de "*classroom*" (salle de classe), avec quelques rôles et politiques de base.

```
-- définir un type de Collaboration
ClassRoom = Collaboration {
        type = "classroom",

        -- define des canaux contrôlables, les autres sont toujours des canaux  ouverts
       channels = {"cmd", "cam", "annot", "cursor"},

       -- définir des canaux initialement contrôlés (tous les autres sont libres par défaut,
       --      mais peuvent être changes par les paroles du propriétaire
        floors = {cmd=true, cam=true},

        -- définir les rôles
        Teacher = Role { type = "teacher" },
        Student = Role { type = "student" },

         -- define les matrices basées sur les rôles pour chaque canal
        K_in = { ["video"]= {["student:student"] = false} },
         K_out = { ["video"]= {[" student:student "] = false} },
         I_in = { ["audio"]= {["student:student"] = false} },
         I_out = { ["audio"]= {[" student:student "] = false} },
         -- rôle assumé par le propriétaire
         owner_role = "instructeur",
         -- rôle par défaut pour les autres participants
          default_role = "étudiant"
}
```

LISTING 3.4 – Définition d'une collaboration de salle de classe.

ClassRoom:grant_floor pourrait être redéfini pour établir une politique de contrôle spécifique pour ce type de conférence. Ici, la politique par défaut est montrée.

```
-- Méthode refédinie pour un type de Collaboration
function ClassRoom:grant_floor(ch, session, requester)
        if floor[ch] == nil then       -- floor array  for the collaboration
                -- no floor for this channel
                 return
        end

        if requester == owner or requester == floor[ch]
        then
                 floor[ch] = session
        else
                 -- grant_floor NIÉ
        end
end
```

LISTAGE 3.5 – Méthode redéfinie pour une collaboration de salle de classe

3.4 Visioconférence

Les canaux de communication audio, en particulier, et vidéo sont fondamentaux pour une collaboration synchrone efficace, un fait bien connu dans le TCAO [OO95, IT94]. Ils sont aussi largement attendus de la part des utilisateurs dans notre contexte d'application, et sont supportés dans NetGocad à travers l'utilisation de l'outil de visioconférence multiplateformes, CSVTool (Collaboration Supportée par Vidéo), développé en coopération entre Petrobras et le groupe Tecgraf à l'université PUC-Rio.

L'utilisation d'un outil sur mesure permet une intégration étroite de ce service dans le système coopératif, avec aucune duplication de fonctionnalités de gestion de session, et également le contrôle direct des flux audio et vidéo selon les politiques de coordination définies.

CSVTool est implémenté avec JMF (Java Media Framework) [JMF], qui emploie un code source de haut niveau et soustrait des codecs et des détails de protocole de transmission. Il a été conçu dans le but de fournir une communication multimédia intégrée aux applications coopératives, mais peut aussi être utilisé comme outil de visioconférence indépendant.

En mode intégré, les flux audio et vidéo sont démarrés pour les conférences continues en utilisant les informations de session déjà disponibles dans l'application

hôte. Dans notre cas, les connexions peuvent être créées et les flux individuels contrôlés (activés et désactivés) directement de l'interface utilisateur de NetGocad. Le contrôle des droits d'émission et de réception pour les participants peut être établi selon leurs rôles dans les types spécifiques de conférence, et des flux individuels peuvent être encore contrôlés à travers le GUI par chaque participant (si l'interface n'est pas bloquée par le propriétaire de la conférence). Ceci permet une meilleure utilisation des ressources de bande passante et de traitement, étant donné que parfois les réglages prédéfinis ne conviennent pas bien aux conditions opérationnelles.

Toutes les informations échangées parmi les clients, sauf les flux audio et vidéo, passent à travers le serveur. Les messages les plus courants sont rajout et enlèvement de participants, qui impliquent la création ou l'enlèvement de flux. Le serveur n'est pas sujet à un trafic surchargé parce qu'il ne reçoit pas le "trafic lourd" – les flux – qui est directement transmis entre les clients, de poste à poste.

La communication client/serveur est implémentée dans CORBA, et la communication parmi les clients pour la transmission de flux est faite en RTP (Protocole de Temps Réel).

En plus de la transmission audio et vidéo sur des sessions à multi-participants et multi-plateformes, CSVTool fournit quelques caractéristiques intéressantes :

- le flux vidéo envoyé par chaque participant peut être commuté de l'image capturée par la caméra vers l'écran capturé, pour permettre l'utilisation de la vidéo pour un affichage à distance de l'opération d'interface, comme analysé au paragraphe 2.3.2.1, ou pour la présentation d'autres contenus sur l'écran et pour des vérifications de consistance;
- un outil textuel de conversation, qui est propice dans certaines situations (par exemple, lorsque quelqu'un a des problèmes avec des dispositifs de capture);
- clichés, utiles pour documenter la session de travail.

L'utilisation de solutions de visioconférence par des tiers n'était pas considérée satisfaisante, à cause des limitations que ceci imposerait: des restrictions dans la création de conférences avec des utilisateurs multiples, spécificité de plateforme, et difficulté dans le développement de la fonctionnalité demandée avec une intégration propre dans l'application.

3.4.1 Outils associés

Il y a actuellement un nombre de programmes et applications qui supportent la visioconférence. Tous offrent des ressources pour la capture vidéo, le codage, la transmission et l'exhibition de flux audio et vidéo. Dans le contexte de ce projet, du fait que nous avons eu besoin de fournir un support de communication intégré pour les applications coopératives existantes, certains aspects devinrent cruciaux pour le choix de l'outil à utiliser: adaptablité, souplesse et architecture de logiciel. Trois outils principaux ont été étudiés : NetMeeting, VIC et JMF (quelques-unes d'autres solutions

de recherches ont aussi été considérées, mais la documentation et le support disponibles n'ont pas été jugés satisfaisants).

NetMeeting est un outil de conférence audio et vidéo qui par défaut permet la communication entre deux personnes. La communication vidéo impliquant un grand nombre d'utilisateurs est uniquement possible en utilsant un MCU (Multipoint Control Unit). D'autres ressources disponibles sont un outil de conversation de texte, un tableau blanc partagé, partage d'applications et accès à des ordinateurs à distance [MNM]. Le problème principal pour son utilisation est, dans notre cas, qu'il est restreint à la plateforme MS Windows.

Un autre outil qui a été étudié et testé, était VIC, développé par le Groupe de Recherche de Réseau au Laboratoire National Lawrence Berkeley en collaboration avec l'Université de Californie, Berkeley [VIC]. VIC est un logiciel libre (*open source*), implémenté en C++, multiplateformes, et distribué gratuitement. Il offre un support à divers utilisateurs et valide une transmission multidiffusion, qui demande MBONE (*multicast backbone*) pour opérer à travers Internet. Malgré le fait d'avoir des caractéristiques basiques nécessaires à ce projet, le code source de VIC est écrit en langage de bas niveau, et a été considéré difficile à réutiliser.

Une fonctionnalité alternative similaire à VIC, implémentée à Java, est JMF (Java Media Framework). Cet API a un code haut niveau, et une indépendance des codecs et des détails de protocole de transmission. Ceci facilite sa compréhension et, par conséquent, l'implémentation de modules complexes. JMF a été adopté dans ce travail comme une plateforme basique pour le développement de CSVTool, et est décrit au prochain paragraphe.

3.4.2 Java Media Framework

JMF [JMF] est un API pour incorporer des médias basés sur le temps en applications Java. Il est extensible et permet l'utilisation de plugins pour supporter des types de média additionnels ou pour réaliser un traitement sur mesure. L'API fournit la fonctionnalité pour capturer, traiter, stocker, présenter et transmettre des flux de données médias, et aussi pour contrôler les paramètres de playback. Les principales caractéristiques de cet API sont qu'il :

- supporte la capture et l'accès à des données médias crues;

- permet le développement d'applications de flux médias et conférences dans Java;

- permet à des développeurs et à des fournisseurs de technologie avancés d'implémenter des solutions sur mesure basées sur l'API existant pour intégrer de nouvelles caractéristiques dans l'environnement-cadre existant ;

- permet le développement de multiplexeurs, de démultiplexeurs téléchargeables, codecs, processeurs et rendeurs d'effets ;

- est une plateforme indépendante et relativement aisée à programmer.

Dans le but de gérer la transmission de flux médias en temps réel, le JFM RTP API est défini, fournissant un support à RTP (Real-Time Transport Protocol). RTP valide la

transmission et la réception de flux à travers le réseau, en fournissant des fonctions de transport de réseau bout à bout convenant aux applications transmettant des données en temps réel, telles que audio, vidéo ou données de simulation, sur des services de réseau multidiffusion ou unidiffusion. Le transport de données est augmenté par un protocole de contrôle (RTCP) dans le but de permettre un monitorage de la distribution des données d'une manière modulable aux grands réseaux multidiffusion et de fournir un contrôle minimum et des fonctionnalités d'identification minimales. RTP et RTCP sont conçus pour être indépendants des couches de réseau et de transport sous-jacentes.

Une application construite sur JMF a un flux de données similaire à celui d'un pipeline, comme illustré Figure 3.8. Les flèches indiquent le flux de données.

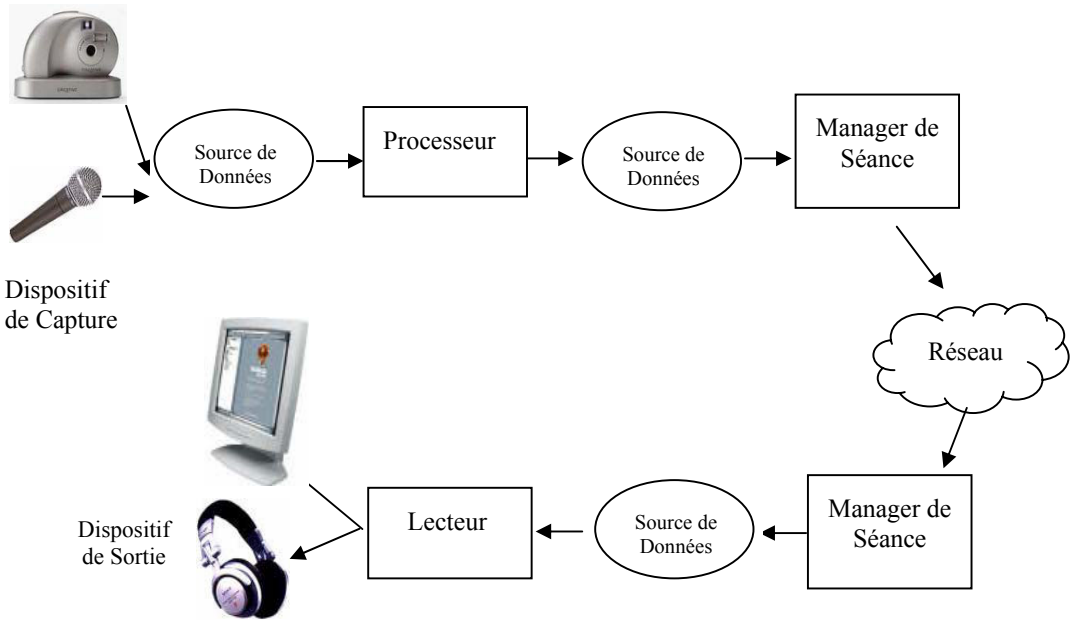


FIG. 3.8 – Pipeline pour capturer, traiter, transmettre et présenter les médias basés sur le temps.

Dans la première partie du flux de données, une application de visioconférence capture des données réelles audio et/ou vidéo, les traite et les transmet à travers le réseau en utilisant une session RTP séparée pour chaque type de média. Un dispositif de capture peut être un microphone ou une carte de capture vidéo. Le format du flux média dépend du traitement réalisé par le dispositif; certains dispositifs distribuent des

données crues, non compressées, tandis que d'autres peuvent distribuer des données compressées. La compression vidéo est le processus le plus demandeur de CPU dans le pipeline. Des contrôles de capture sont disponibles pour la spécification des formats et taux de données.

Des flux RTP arrivantes peuvent être lues localement, rangées dans un fichier, ou les deux. Un Lecteur séparé est utilisé pour chaque flux reçu par le Manager de la Session (il y a un Manager de Session RTP associé pour chaque type de média sortant ou arrivant). Les Lecteurs traitent un flux d'entrée distribué par une source de données, et le rendent à une heure précise sur un dispositif qui supporte les médias présentés, comme vidéo ou cartes de son. JMF définit la Source de Données comme abstraction de données qui encapsule le flux média comme une cassette vidéo. Pour gérer le transfert du contenu média, une Source de Données encapsule l'endroit des médias, le protocole et le logiciel utilisés pour la distribution.

Les Lecteurs sont très similaires à des Processeurs. Un Processeur est juste un type spécialisé de Lecteur qui fournit le contrôle sur le traitement qui est réalisé sur le flux média d'entrée, comme les effets, le mixage, l'encodage, et la composition en temps réel. En plus de rendre les données médias aux dispositifs de présentation, un processeur peut les sortir à travers une Source de Données. Les processeurs peuvent utiliser des codecs pour réaliser l'encodage et le décodage de données, et pour ajuster la qualité de compression. Une plus grande compression conduit à un plus grand usage de CPU et temps d'attente pendant la présentation.

Un Manager de Session JMF est utilisé pour coordonner une session RTP. Il garde une pile des participants de la session et les flux qui ont été transmis. Il permet aussi la définition de méthodes qui valident les applications à initialiser et démarrent la participation dans une session, enlèvent les flux individuels créés par l'application, et closent toute la session JMF.

3.4.3 Modes d'opération CSVTool

En plus d'avoir été conçu pour fournir des canaux de communication audio et vidéo intégrés avec des applications coopératives, telles NetGocad, CSVTool peut aussi être utilisé comme un outil de visioconférence autonome.

Le mode d'opération est défini pendant le démarrage, par le moyen d'arguments spécifiques de ligne de commande. Nous nous référons au Mode 1 lorsque l'outil opère comme un outil de visioconférence autonome, et au Mode 2 lorsque intégré à une application coopérative. Actuellement, les deux modes d'opération sont implémentés comme une seule application Java.

Indépendamment du mode opération, CSVTool est divisé en deux modules, le serveur et le client. Le serveur est un module indépendant, responsable de la gestion des participants et des visioconférences. Il gère toutes les informations échangées entre les clients, sauf les flux audio et vidéo, et opère d'une manière transparente quel que soit le mode d'exécution du client (1 ou 2), échangeant le même ensemble de messages

dans les deux cas. Le noyau du module du client, qui gère les flux, est aussi le même pour les deux modes d'opération, comme montré dans les Figs. 3.9 et 3.10.

Dans les paragraphes suivants, les principales caractéristiques et différences entre les deux modes d'opération de CSVTool sont présentées.

3.4.3.1 Mode 1 – Autonome

En Mode 1, les clients se connectent à un serveur central fixé, dont l'endroit doit être connu. Ce serveur accepte les demandes des clients pour la création de visioconférences et garde la liste de tous les utilisateurs connectés, ainsi que la liste de visioconférences actives avec leurs participants respectifs. Il informe aussi les participants lorsqu'il y a une altération dans le groupe de la session (par exemple, un nouveau participant est connecté), redéfinissant les transmissions RTP à la nouvelle configuration du groupe. Un diagramme de cette structure est présenté dans la Figure 3.9, qui montre aussi les deux types de transmission de données (CORBA et RTP).

Lorsqu'un nouveau client se connecte au serveur, il reçoit une liste des conférences auxquelles il est autorisé à participer, ainsi que les informations concernant les utilisateurs actuellement enregistrés dans le système. Toute altération de données de serveur est envoyée à tous les utilisateurs connectés. Le coordinateur de la conférence est la seule personne capable de clore la conférence.

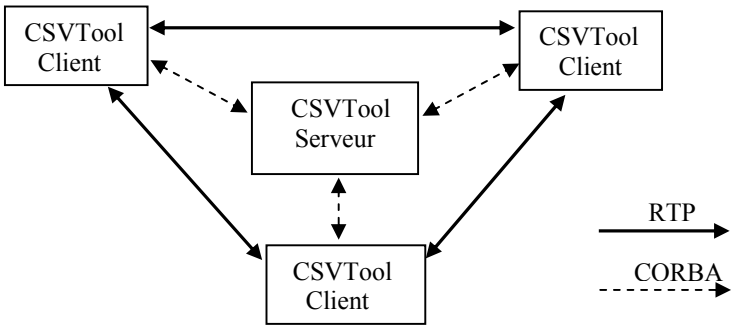


FIG. 3.9 – Structure de communication entre le client CSVTool et le serveur en mode d'opération autonome (Mode 1).

.

3.4.3.2 Mode 2 – Intégré

Ce mode d'opération a été conçu pour être aisément couplé à des applications coopératives. Dans ce mode, CSVTool est utilisé pour la création d'une visioconférence pour une session de collaboration ayant déjà lieu. L'application hôte

est responsable de l'initialisation de la visioconférence, sans le besoin de configuration manuelle et inclusion de participants, comme cela arrive en Mode 1. Indépendamment de qui dans le groupe a démarré la visioconférence, le rôle de coordination est assigné au propriétaire de la conférence informé par l'application hôte.

L'application hôte peut régler les matrices de connexion définissant les flux actifs, comme décrit ci-dessous, et verrouiller ou déverrouiller l'interface pour la modification de la configuration par les utilisateurs.

Opération CSV avec NetGocad

Lorsqu'un participant NetGocad démarre une visioconférence, la Conférence lance un serveur CSV à travers son propriétaire (Figure 3.10) et demande à tous les Participants de démarrer les clients CSV, qui se connecteront automatiquement au serveur (dont l'adresse leur est envoyée).

Le serveur CSV est ensuite responsable de la gestion des clients et des flux audio et vidéo. Toute la communication entre la Conference NetGocad et le Server CSV est faite via CORBA, à travers le propriétaire de la conférence. Des flux audio et vidéo RTP sont toujours créés directement entre les clients, de poste à poste.

Les matrices de contrôle se déplacent entre les clients CSV et le serveur, si nécessaire, au moyen d'échange de messages. Les matrices Key et Intention sont envoyées par la Conference NetGocad à CSV, qui les utilise pour régler l'état de connexion (comme expliqué en détail au paragraphe suivant). CSVTool actualise les matrices Key reçues pour indiquer la disponibilité des dispositifs de capture à chaque machine des participants, testés au démarrage.

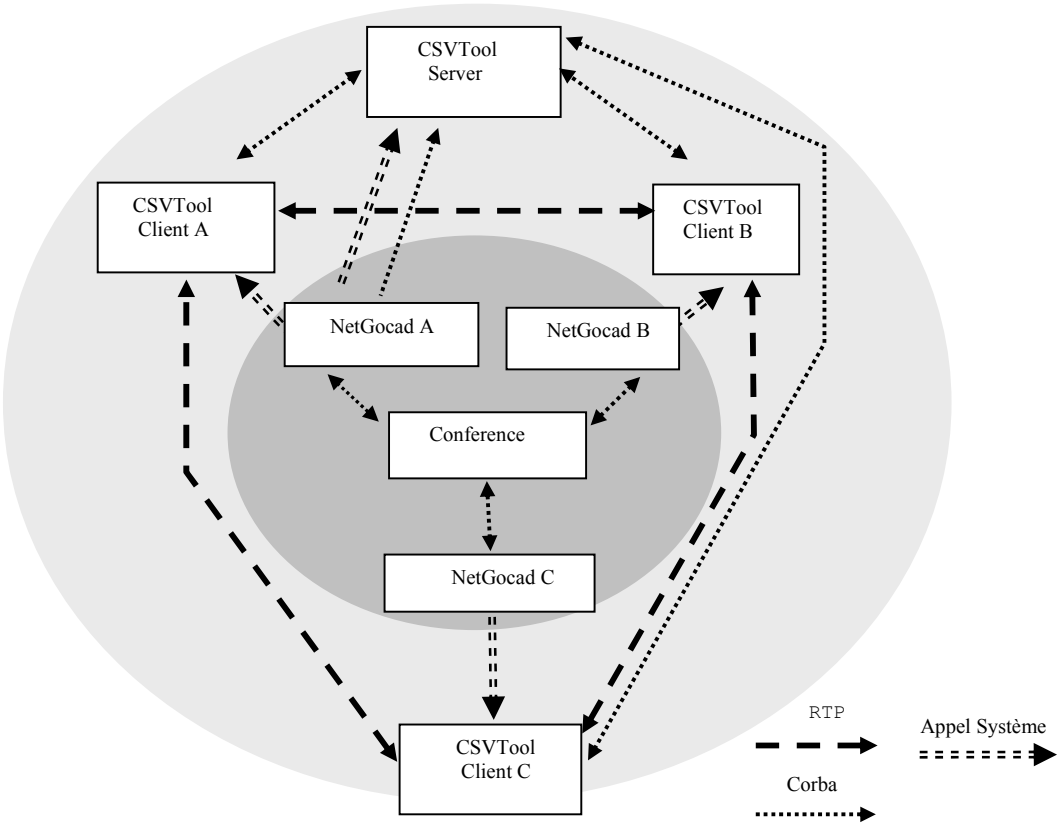


FIG. 3.10 – CSV avec NetGocad. Voir aussi Figure 4.2 pour les images des interfaces de l'utilisateur.

En complément à la matrice de connexion, une matrice External Intention (Intention Externe) est saisie dans CSV pour refléter les connexions qui ne sont pas établies parce qu'un seul côté a décidé de ne pas valider le flux. Cette information est utilisée pour la sélection de l'icône approprié pour représenter l'état de la connexion à l'interface de l'utilisateur (Figure 3.15). Comme les utilisateurs ont des fenêtres individuelles relatives les unes aux autres, les intentions des participants des deux côtés d'une connexion pour chaque canal peuvent être explicitement indiquées.

Lorsqu'un nouveau participant entre dans une conférence il sera automatiquement rajoutée à la conférence CSVTool. L'application hôte est responsable de l'envoi de cette information au serveur CSVTool, indiquant les flux actifs à travers les matrices de contrôle, selon les politiques de coordination actuelles et les rôles des participants. Par exemple, dans une session de formation, initialement des flux vidéo sont uniquement créés de l'instructeur aux apprentis, qui sont capables d'envoyer des flux à

l'instructeur. Dans une session libre, chaque participant peut envoyer/recevoir des flux à/de tous les autres.

Le réglage automatique des connexions, cependant, n'est pas suffisant pour résoudre le problème de bande passante et de limitations de traitement au moment de l'initialisation. Dépendant des politiques de coordination, tous les participants peuvent commencer à envoyer et recevoir des flux au début de la session, quel que soit leur réel intérêt. Par conséquent, une stratégie pour un contrôle individuel par les participants sur chaque connexion possible a été développée. Elle prend en compte les intentions dynamiques d'envoi et de réception de tous les utilisateurs dans la session, en plus des informations statiques dans la matrice Key définissant leurs droits selon le type de session et les dispositifs de capture disponibles.

Au paragraphe suivant, nous décrivons les aspects de l'implémentation associés à cette stratégie.

3.4.4 Contrôle de Participation

Sans mécanisme pour le contrôle de participation, le nombre de connexions RTP peut facilement surcharger les CPU et les réseaux. Considérant l'existence de sites à faible largeur de bande, le problème est encore plus considérable. Dans le cas de surcharge du réseau ou du traitement, uniquement les connexions les plus importantes devraient être maintenues actives. En observant la qualité des flux, les utilisateurs peuvent enlever des connexions en faveur des plus essentielles, jusqu'à ce que la présentation des données atteigne un niveau adéquat. Le niveau de compression vidéo doit aussi être utilisé comme une ressource supplémentaire pour contrôler le taux entre la qualité, la performance et la pertinence des données.

Pour implémenter ces politiques de participation, chaque connexion individuelle doit être gérée indépendamment. Pour cela, les connexions sont représentées par les valeurs booléenes organisées comme des matrices carrées, selon les intentions et les permissions pour envoyer et recevoir les flux à/de chaque participant (comme décrit au paragraphe 3.3.3 en termes plus généraux). Les tailles des matrices augmentent lorsque de nouveaux participants entrent dans la session. Nous définissons quatre types de matrices booléennes pour traiter chaque type de média :

Key (Clé)– Forme le format de la session collaborative. Elle est générée dans le serveur, prenant en compte les dispositifs de capture disponibles dans l'ordinateur de chaque participant et aussi le type de la session (conceptuellement, invalider une connexion due au rôle d'un participant dans un certain type de conférence est équivalent à éteindre le dispositif nécessaire pour envoyer ou recevoir les informations concernées).

Intention – Garde les intentions d'envoi/réception aux/de autres participants. Dans l'implémentation de CSVTool elle est subordonnée à la matrice Key. Cela signifie que lorsque la clé est fausse pour un champ donné, ce champ est aussi faux dans la matrice

intention (ainsi la clé est déjà prise en compte). Cette matrice est individuelle pour chaque participant.

Connection (Connexion)– Résulte de la compilation des matrices intention de tous les participants. Tenant compte de toutes les intentions d'envoi et de réception, cette matrice exprime continuellement les connexions valables: une connexion est établie lorsque les intentions d'envoi et de réception des participants en corrélation sont vraies.

External Intention (Intention Externe) – Complémentaire à la matrice de connexion, cette matrice auxiliaire reflète les connexions qui ne sont pas établies parce que juste un côté a décidé de ne pas valider le flux. Par exemple, chaque participant A veut envoyer de l'audio à B mais B ne veut pas recevoir, il est utile de prévenir B (considéré dans ce cas, d'une perspective de A, comme un participant externe) que A veut parler. Les valeurs de cette matrice sont juste considérées lorsqu'une connexion donnée n'est pas établie.

Ces matrices sont utilisées pour contrôler et limiter les connexions actives, et fournir aussi un feedback aux utilisateurs concernant leur état. Les données sont stockées dans CSVTool et dépendent du type de matrice. Dans Touche et Connexion, uniquement des lignes sont utilisées. Pour l'Intention, les lignes représentent les intentions envoyées, alors que les colonnes représentent les intentions reçues. Tous les participants sont associés avec une ID qui est utilisée comme un index lors de l'accès aux lignes et colonnes des matrices.

Aux paragraphes suivants, nous analysons ces matrices à partir des perspectives du client et du serveur. Nous présentons aussi quelques exemples de matrices, représentant les configurations pour une situation hypothétique donnée qui éclaircit la stratégie proposée.

3.4.4.1 Perspective du client

Lorsqu'un utilisateur démarre l'application du client, les informations concernant les dispositifs de capture utilisés, avec les données personnelles, sont envoyées au serveur, qui initialise les matrices Key, Connetion et External Intention pour chaque participant. Ces trois matrices, y compris les données d'autres participants éventuels, sont ensuite renvoyées au client. Ces informations sont utilisées pour configurer l'interface graphique de l'utilisateur (voir paragraphe 3.4.4.3) qui reflète quelles connexions peuvent être établies. A ce moment, aucune connexion n'est active.

Le serveur est informé chaque fois que les clients changent leurs intentions d'envoyer ou de recevoir des flux. Les matrices Connetion et External Intention sont reconstruites et envoyées à tous les clients, qui actualisent leurs interfaces et les connexions qui ont été changées. Dans ce processus, des connexions RTP peuvent être créées ou enlevées pour maintenir la cohérence de la session. La Figure 3.11 présente quelques configurations vidéo pour une session avec trois participants (une configuration similaire est utilisée pour l'audio).

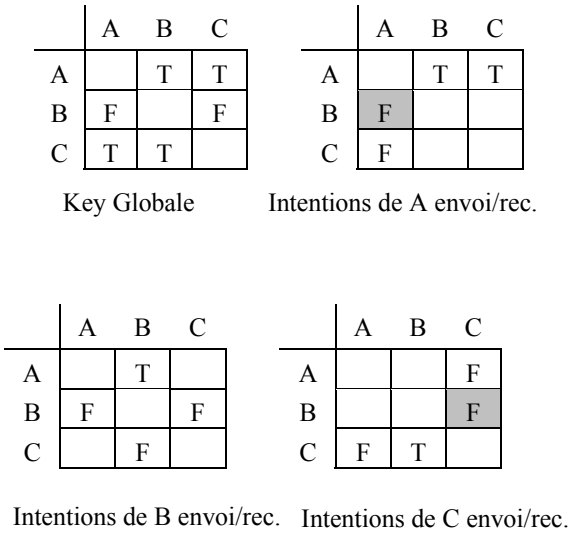


FIG. 3.11 – Exemple de matrices pour contrôler les permissions (Key) et les intentions. Les intentions d'envoyer sont représentées par des lignes, les intentions de recevoir par des colonnes.

En observant la matrice Key, on peut remarquer que les participants A et C peuvent envoyer une vidéo à tous les autres (les première et troisième lignes de la matrice Key, qui ont uniquement des valeurs vraies). La restriction de B à envoyer la vidéo aux autres (de fausses valeurs sur la seconde colonne de la matrice Key) peut être la conséquence de l'indisponibilité des dispositifs de capture. Ceci prévient aussi les participants de la réception vidéo de B, qui est exprimée aux matrices intention de A et C (le 'F' sombre dans les matrices intention de A et C).

Les matrices restantes représentent les intentions des utilisateurs sur l'envoi et la réception de vidéo. Par exemple, A veut envoyer à B et C (première ligne de la matrice intention de A) et ne veut pas recevoir de C (troisième élément de la première colonne de la matrice intention de A). A ne peut pas recevoir de B parce que B n'est pas supposé envoyer une vidéo. En observant les matrices intention de B et C, il est possible de vérifier que B veut recevoir une vidéo uniquement de A; C veut envoyer une vidéo à B et ne veut pas recevoir de A.

3.4.4.2 Perspective du serveur

Le serveur agit comme manager des participants, contrôlant les flux audio et vidéo échangés entre eux. Au moyen d'un schéma de diffusion de message, il communique

aux participants l'entrée et la sortie des autres, construit et envoie de nouvelles matrices de connexion lorsque les intentions changent, et redirige les messages de texte aux participants spécifiques.

Il y a quatre matrices dans le serveur pour stocker les intentions d'envoyer et recevoir, deux pour l'audio et deux pour la vidéo. Les matrices Connetion et External Intention sont construites à partir de ces matrices. Lorsque le serveur reçoit une matrice intention d'un participant avec une ID donnée, ses valeurs sont copiées vers les matrices d'envoi et de réception respectives. Les intentions d'envoyer sont copiées vers les lignes de la matrice globale d'intentions d'envoyer, et les intentions de réception vers les colonnes de la matrice globale d'intentions de recevoir, de sorte que les opérations directes booléennes peuvent être réalisées entre ces matrices pour construire la matrice Connetion. La Figure 3.12 montre les matrices du serveur correspondant aux matrices du client présentées dans la Figure 3.11.

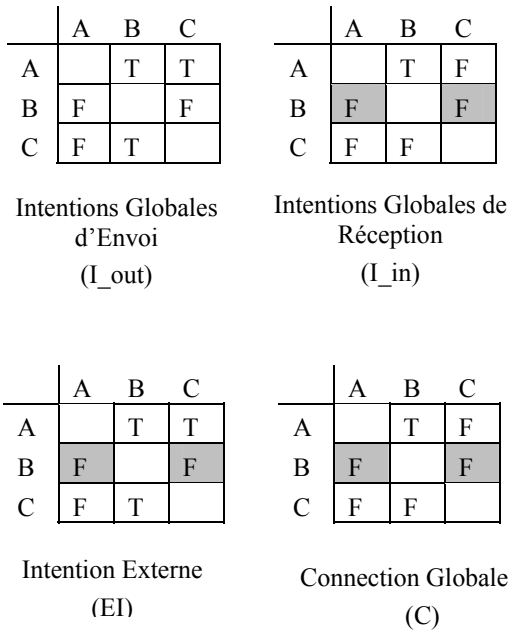


FIG. 3.12 – Les matrices globales du serveur pour un canal, correspondant aux matrices du client sur la Figure 3.11.

La configuration de l'interface correspondante est montrée sur la Figure 3.13. L'unique connexion vidéo survient entre A et B, étant donné que A veut envoyer à B (première ligne de la matrice globale d'intentions d'envoyer), et B veut recevoir de A (seconde colonne de la matrice globale d'intentions de recevoir). A n'envoie pas à C

parce que C, même sachant que A veut envoyer une vidéo, ne veut pas la recevoir. La même chose arrive entre C et B.

En construisant la matrice Connetion, il n'est pas nécessaire de tenir compte de la matrice Key parce qu'elle est déjà reflétée dans la matrice Intention et dans l'interface graphique, invalidant les boutons pour lesquels les actions ne sont pas autorisées. Par conséquent, les formules suivantes spécifient les matrices Connetion et External Intention à partir des matrices globales d'intentions d'envoi et de réception.

$$C_{ij} = I\_out_{ij} \wedge I\_in_{ij}$$
$$EI_{ij} = I\_out_{ij} \vee I\_in_{ij}$$

3.4.4.3 Interface de l'utilisateur

L'interface graphique de l'utilisateur joue un rôle important comme un moyen de permettre un contrôle de l'utilisateur et une conscience à propos des intentions d'autres de communication sur les connexions audio et vidéo. Ceci reflète la configuration actuelle des matrices reçues à partir du serveur (Key, Connetion et External Intentions) au moyen de boutons stylés, comme présenté sur la Figure 3.13. Cette figure montre la perspective des participants A et B, selon le cas d'usage hypothétique présenté au paragraphe précédent.

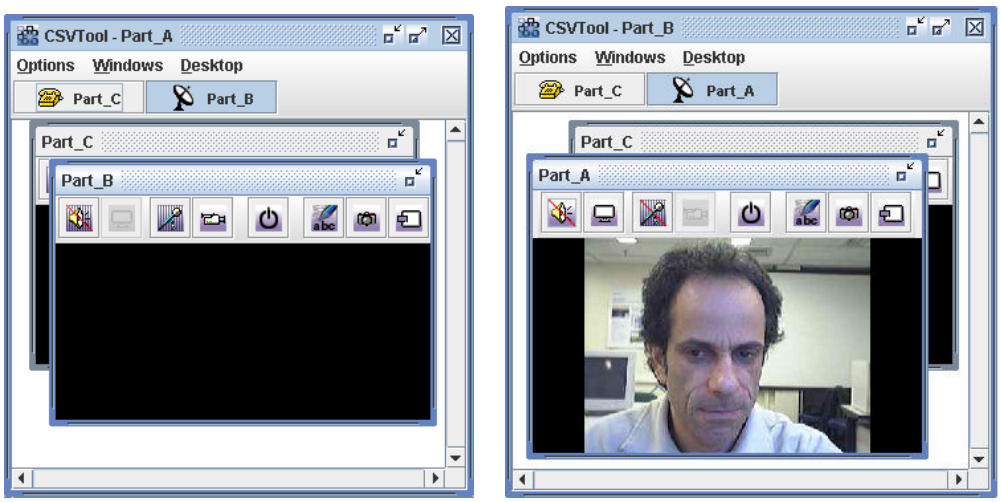


FIG. 3.13 –CSVTool GUI à partir des perspectives des participants A et B.

A travers la barre d'outils (*toolbar*), les intentions de l'utilisateur local sont exprimées, et l'utilisateur peut aussi être conscient des flux qui sont actifs, ceux qui peuvent être activés, et ceux qui ne le peuvent pas. Des boutons supplémentaires permettent l'utilisation de messages de texte, instantanés, et la transformation de fenêtres internes en cadres séparés (Figure 3.14). Chaque bouton de contrôle audio/vidéo peut assumer cinq configurations différentes (Figure 3.15).

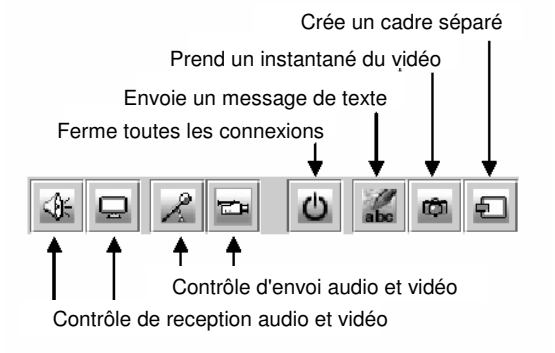


FIG. 3.14 – Barre d'outils pour contrôle d'intentions d'envoi et de réception

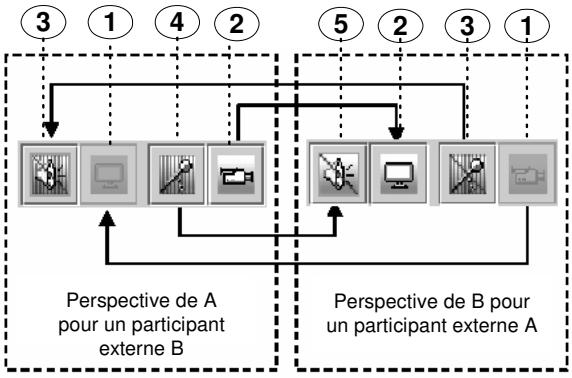


FIG. 3.15 – Interface CSVTool. Chaque bouton de contrôle audio/vidéo peut assumer cinq configurations différentes : 1. Invalide: le flux respectif ne peut pas être activé parce que le dispositif de capture est indisponible ou invalide en raison du rôle du participant dans le type de conférence actuel – le bouton devient gris; 2. Active: la connexion est active; 3. Off: les deux participants ne veulent pas activer le flux – le bouton est barré et hachuré; 4. Attente: le participant local veut activer le flux, mais le participant à distance ne veut pas – l'arrière-plan du bouton est hachuré; 5. Externe: le participant à distance veut activer le flux, mais le participant local ne veut pas – le bouton est barré.

En appuyant sur n'importe quel bouton, la matrice Intention locale est envoyée au serveur. Après compilation, les nouvelles matrices Connetion et External Intention retournent du serveur et les boutons sont actualisés, aussi bien que les flux envoyés et reçus par le participant local.

Lorsque CSVTool est démarré à partir de NetGocad, ces matrices sont configurées automatiquement selon les définitions basées sur les rôles pour le type de conférence, et peuvent être verrouillées optionnellement par le propriétaire de la conférence.

3.4.5 Transmission d'écran

Comme analysé au chapitre 2, pour la formation et la consultation, l'affichage à distance de l'opération d'interface de l'utilisateur est très important – l'observation du travail de quelqu'un d'autre est un moyen très efficace de transfert de la connaissance. Dans ce but, l'outil de visioconférence peut être utilisé pour la transmission de l'écran d'un utilisateur au lieu de l'image capturée par la caméra, en appuyant simplement sur un bouton dans le moniteur de capture.

Cette ressource fournit une manière simple d'utiliser le flux vidéo pour l'affichage à distance de l'interface de l'utilisateur, à travers l'outil de visioconférence standard (voir Figure 4.7). La résolution vidéo est limitée aux réglages actuels. Dépendant du choix de l'utilisateur, la compression du logiciel est employée pour réduire tout l'écran à la résolution actuelle, ou une fenêtre non compressée peut être envoyée comme indiqué dynamiquement avec la position du curseur. Nous avons fixé la résolution maximale autorisée à 640 x 480 pixels, 10 trames/seconde, mais l'utilisation de résolutions et de fréquences de trames plus élevées est possible, si suffisamment de bande passante et de capacité de traitement sont disponibles à tous les sites impliqués.

3.4.6 Contrôle de concurrence

Lorsqu'une nouvelle connexion est établie entre deux points, les objets JMF responsables de l'envoi et de la réception du flux doivent être créés des deux côtés. La création de ces objets implique l'initialisation du dispositif entrée ou sortie et la préparation de l'objet responsable de traiter le flux. Dans quelques situations, comme l'initialisation pour une caméra, ce processus peut durer quelques secondes. Pendant ce temps, l'utilisateur peut essayer de changer le statut de la connexion. Une situation similaire peut arriver lorsqu'une connexion est close et que l'objet est détruit.

Dans ces cas, les objets construits ne peuvent pas être détruits jusqu'à ce que le processus de construction soit terminé. La même chose s'applique au processus de destruction. Si les altérations de l'utilisateur sont simplement ignorées, nous pouvons arriver à un statut d'exécution d'erreur, avec un objet actif dans une connexion non établie ou vice-versa. Pour résoudre ce problème, CSVTool associe chacun de ces objets à un sémaphore, indiquant que l'objet est en train d'être altéré. Les changements dans la matrice de connexion sont ignorés pour un objet pendant que cet objet est altéré. Lorsque l'altération de l'objet est terminée, la matrice de connexion locale est

vérifiée pour contrôler si l'état actuel de l'objet reflète l'état actuel de la matrice. Sinon, l'objet est enlevé ou créé.

3.5 Réalité virtuelle

Un plugin de réalité virtuelle Go2VR [GO2] permet aux participants en session de collaboration de travailler avec affichage immersive, visualisation stéréoscopique et dispositifs tridimensionnels avec suivi de position (*tracking*). Go2VR a été conçu pour être complètement compatible avec NetGocad, de sorte que l'utilisateur RV soit capable de participer à la session de groupe à travers les mécanismes décrits auparavant. Go2VR est basé sur une application de réalité virtuelle, GocadVR [WBCP99], qui a été développée comme une version spécialisée indépendante de Gocad, pas compatible avec d'autres plugins (non supportés à l'époque).

Dans ce travail nous sommes particulièrement intéressés par l'utilisation combinée de Go2VR et NetGocad pour permettre l'exécution de sessions de collaboration impliquant à la fois des utilisateurs de réalité virtuelle et d'ordinateur de bureau. Comme analysé au paragraphe 1.4.3, nous pensons que ce type de collaboration peut faciliter l'accomplissement de certaines activités en combinant l'interaction spatiale accrue d'utilisateurs immergés avec le travail de pairs externes sur ordinateurs de bureau, responsables de la réalisation de tâches pour lesquelles des manipulations plus longues de l'interface de menus sont nécessaires. Comme habituellement un nombre très limité d'environnements d'immersion est disponible dans une société, ceci permet aussi la collaboration entre les utilisateurs en immersion et ceux d'ordinateurs de bureau.

Naturellement, la collaboration à distance impliquant des utilisateurs RV multiples – comme fournie par beaucoup de systèmes académiques et quelques systèmes commerciaux – est aussi une possibilité intéressante, à exploiter encore complètement dans le contexte des activités E&P.

3.5.1 Fonctionnalité du plugin RV

Go2VR étend Gocad avec les fonctionnalités suivantes :

- des affichages multiparois, avec des configurations arbitraires (CAVE, Centres de Réalité à écran courbe, PowerWalls ou Workbenches);
- un rendu multi-CPU et multi-pipe;
- une visualisation stéréoscopique, suivi de la tête et dispositifs d'entrée 3D ("baguettes") pour la navigation, sélection et manipulation d'objets ;
- mode de navigation "fly-though";
- intégration aisée de nouveaux dispositifs d'interaction;
- affichage direct dans l'environnement d'immersion d'objets et de méthodes de visulation générés par Gocad et ses plugins.

L'implémentation de Go2VR repose sur Gocad, sur le Multipipe SDK de SGI et sur des bibliothèques d'interaction de VSP Technology.

Quand le plugin Go2VR est chargé avec Gocad, cela permet de démarrer une interface immersive, selon le fichier de configuration définissant le type d'environnement et les ressources graphiques disponibles.

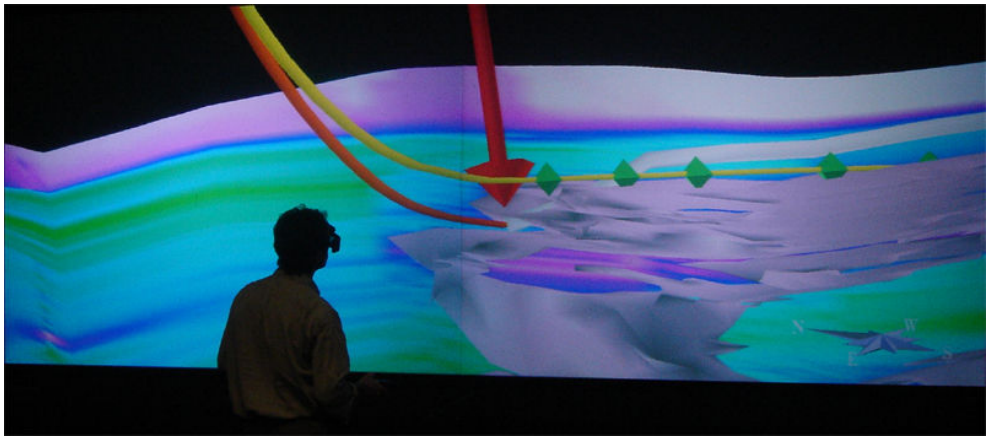


FIG. 3.16 – Go2VR. Un participant de réalité virtuelle analysant un forage de puits réel (rouge) et le puits prévu (jaune). La flèche rouge est une annotation créée par un autre participant.

3.5.2 Collaboration à distance

Lorsque NetGocad est chargé avec Go2VR, l'utilisateur RV devient capable de participer à une session collaborative. Comme Go2VR travaille comme un plugin Gocad ordinaire, la plupart des mécanismes basiques rajoutés par NetGocad sont fonctionnels automatiquement. Les questions qui ont eu besoin d'être gérées pour la compatibilité étaient :

- La création d'un mécanisme pour diffuser les télépointeurs et les annotations, étant donné qu'ils sont introduits par NetGocad dans Gocad à travers la redéfinition d'observateurs pour la caméra, mais Go2VR utilise une implémentation de caméra indépendante.
- La création d'outils d'interaction 3D spécifiques pour les annotations et télépointeurs, utilisés par la baguette 3D.
- La définition d'une caméra spécifique lancée par NetGocad pour une synchronisation à sens unique avec Go2VR. Normalement les mouvements de la caméra de l'utilisateur de l'ordinateur de bureau ne devraient pas être transmis à l'environment RV, étant donné que les utilisateurs en immersion sont

suivis et ainsi ont besoin d'avoir des points de vue individuels, mais une caméra secondaire avec la même étiquette que celle du RV permet à l'utilisateur de l'ordinateur de bureau d'observer le point de vue de l'utilisateur RV (voir paragraphe 3.3.2.2.).

- Comme d'habitude, le point de vue de l'utilisateur RV devrait aussi être diffusé comme la position de l'avatar. En fait, plus de canaux devraient être alloués pour diffuser la position et l'orientation suivi de la baquette de l'utilisateur, et éventuellement d'autres points suivis, dans le but de permettre une description d'avatars plus riche. Actuellement les avatars ne sont représentés que par des positions de visualisation, mais les télépointeurs créés et diffusés par le participant RV gardent les informations de direction de la baguette.

Comme d'autres participants dans une session de groupe peuvent être soit des utilisateurs RV ou non, les canaux individuels de collaboration fournis par NetGocad ont besoin d'être gérés selon le type d'interface employé par chaque participant. La définition de quels canaux sont actifs ou non pour chaque participant, selon leurs rôles, peut être donnée pour la conférence comme décrit au paragraphe 3.3.5.

Actuellement, Go2VR est entièrement fonctionnel pour des buts de visualisation, mais pour la modélisation et la collaboration à distance il est encore à un stade de prototype. Quelques outils d'interaction tridimensionnelle initiale sont disponibles, mais pour l'utilisation opérationnelle des nouveaux manipulateurs ont besoin d'être développés. Pour une collaboration à distance plus efficace, quelques problèmes doivent aussi être résolus, spécialement l'implémentation d'une caméra à pilotage automatique dans Gocad pour la collaboration ordinateur de bureau -RV. Actuellement, la caméra Gocad défaut peut être réglée à distance avec le frustum immersif envoyé par un participant RV, mais ne permet pas une navigation en mode *fly-through* (à travers le modèle) indépendante (la navigation *fly-through* est possible avec le plugin Go4Space [Go4] pour le SpaceMouse).

3.5.3 Collaboration hétérogène

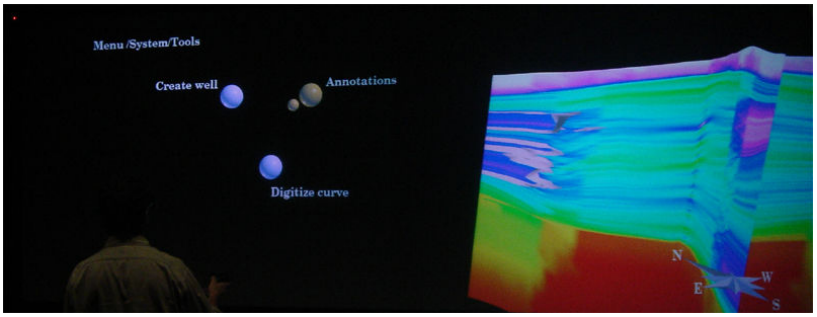
Dans ce paragraphe, nous commentons plus en détail notre motivation pour supporter une collaboration hétérogène à distance. Notre opinion est basée principalement sur des expériences de travail avec Go2VR et Inside Reality [IR], et sur des discussions avec les utilisateurs. L'intégration de la RV dans un système coopératif hétérogène et multimodal permet l'utilisation de l'interface RV pour quelques tâches spécifiques, en combinaison avec d'autres interfaces spécialisées utilisées par d'autres participants, toutes opérant sur un modèle partagé.

Dans des environnements immersifs, l'interaction avec des interfaces d'utilisateur régulières basées sur des menus conventionnels a beaucoup de limitations. Les résolutions normalement atteignables pour une visualisation stéréo sont relativement faibles pour de grands affichages, étant donné que la résolution pleine utilisée pour un affichage sur un mur immersif est généralement équivalente à celle d'un écran d'ordinateur de bureau standard.

Des barres de menu fixes ne sont pas employées, et les menus devraient être restreints à de petites portions de l'écran pour éviter de bloquer la vue de la scène. La zone totale de pixels disponible pour les menus est faible, ainsi moins d'options sont accessibles à chaque instant. Les boutons doivent également être relativement grands pour être lisibles et faciles à choisir à distance, particulièrement si l'interaction à lancer de rayons (*ray casting*) est employée.

Go2VR emploie un paradigme de menu circulaire ("pie-menus" [CHWS88, PIE]), dans lequel les options ne sont pas sélectionnées en pointant, mais par des déplacements relatifs du dispositif d'interaction suivi. Seulement les opérations plus plausibles à utiliser dans l'environnement immersif devraient être directement accessibles dans les menus 3D, pas le grand ensemble d'opérations disponible dans la version standard de l'application.

Par exemple, dans une interface optimisée pour une planification de puits en immersion, les opérations de modélisation et de visualisation doivent être directement disponibles, mais pas celles nécessaires à une modélisation plus compliquée, y compris les widgets avec beaucoup de champs de texte à remplir. Si leur utilisation est nécessaire, par exemple pour une grande correction régionale du modèle, cela peut être mieux fait par un autre participant dans la session collaborative, utilisant une interface conventionnelle.



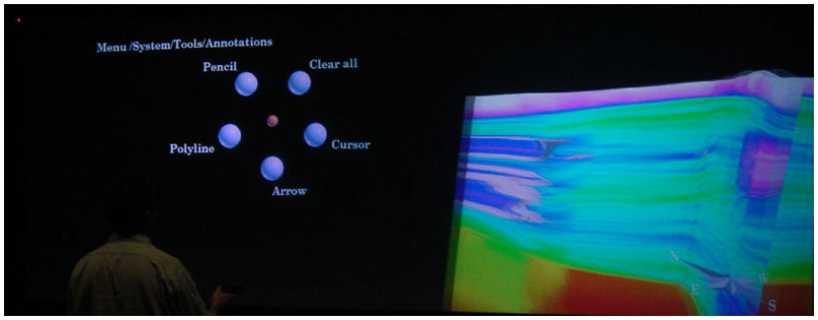


FIG. 3.17 – Go2VR. Les menus immersifs utilisés pour la sélection d'annotations NetGocad.

Il est aussi important de noter que pour un utilisateur en immersion, une interaction locale 3D avec des objets en portée (comme les cibles et les trajectoires de puits) est améliorée, mais des manipulations plus générales du modèle (par exemple, l'édition sur des grandes zones) demandant des changements rapides de points de vue sont plus facilement faits d'un point de vue distant, avec un paradigme de navigation "examinateur" et une interaction basée sur la souris.

3.5.4 Haptique

Comme analysé au paragraphe 1.4, l'utilisation de dispositifs haptiques (à retour d'effort) a le potentiel pour améliorer considérablement la performance d'interaction dans quelques tâches de modélisation difficiles, comme la mise en forme précise de surfaces géologiques et trajectoires de puits [Har04]. Un système, ReachinGeoEditor [RGE], a été développé en projet conjoint entre Reachin Technologies, Chevron et EarthDecision Sciences, et est intégré à Gocad à travers un module d'extension.

En fait GeoEditor est une application indépendante, développée avec une bibliothèque propriétaire et une interface d'utilisateur graphique adaptée pour le Reachin Display System (qui comprend un bras à retour d'efforts Phantom, un moniteur stéréoscopique et un cadre de support avec un miroir semi-transparent). Elle est lancée à travers Gocad, chargée avec le plugin spécifique, et communique avec lui en employant une version simplifiée du protocole NetGocad dans le but de transférer des modèles à partir de Gocad vers GeoEditor et en retour avec un simple clic de souris. La manipulation du modèle ne peut pas être faite de façon synchronisée en utilisant l'ordinateur de bureau et les interfaces haptiques; les objets ont besoin d'être transférés explicitement entre les deux applications après la modélisation.

Nous avons testé ce système, combiné avec NetGocad, en situations de collaboration dans lesquelles un participant travaillant d'un site à distance où le système haptique

n'est pas disponible coopère avec un utilisateur local équipé du GeoEditor. L'utilisateur local transfère une surface au système Reachin, l'édite et ensuite le renvoie à la session partagée. Le participant à distance voit le résultat du travail sans savoir qu'il a été fait avec un dispositif haptique auxiliaire.

Le fonctionnement du système haptique à partir d'un plugin complet permettrait son utilisation directe combinée avec NetGocad, comme partie d'une session hétérogène de modélisation de groupe avec une fonctionnalité de collaboration intégrale (comme analysé ci-dessous pour la réalité virtuelle immersive).

3.6 Acquisition de données à distance

Le plugin d'acquisition de données en temps réel gWLog [CR03, Cam02] peut être employé conjointement à NetGocad pour une intégration automatique dans la séance de travail de données recueillies dans des sites à distance, d'une manière flexible et configurable.

Il est composé de deux modules distincts: un client (le plugin lui-même) et un serveur de données installé au site surveillé, qui obtient les données du système de forage et envoie les commandes Gocad aux clients connectés pour actualiser l'objet du puits. Les méthodes Gocad et les objets sont automatiquement externalisés et accédés par le serveur à travers le langage interprété Lua. L'interface pour Gocad est créée avec l'utilisation de l'outil *tolua* [Lua], qui basé sur les fichiers de tête génère toutes les fonctions nécessaires de l'interface pour permettre leur utilisation à travers Lua (voir Appendice 3).

Dans l'implémentation de ce module d'extention, CORBA est aussi utilisé pour la communication entre le serveur et les clients, mais pas directement. D'ailleurs, le langage Lua est aussi utilisé comme une couche intermédiaire pour garantir l'adaptabilité dynamique du serveur de données, de telle sorte qu'il puisse être aisément personnalisé au temps de passage pour recueillir des données en différents formats. En fait, comme analysé à l'Appendice 3, deux composants intermédiaires sont utilisés: gLua, un plugin Lua pour Gocad, et LuaORB, responsable pour fournir un support pour le développement d'applications incorporant les objets CORBA à travers Lua d'une manière simple et dynamique. En utilisant l'introspection de CORBA (Interface Repository) et une construction de méthodes dynamique (Dynamic Invocation Interface – DII), LuaORB offre une interface à travers laquelle les programmes Lua peuvent définir et utiliser les nouvelles interfaces en temps d'exécution.

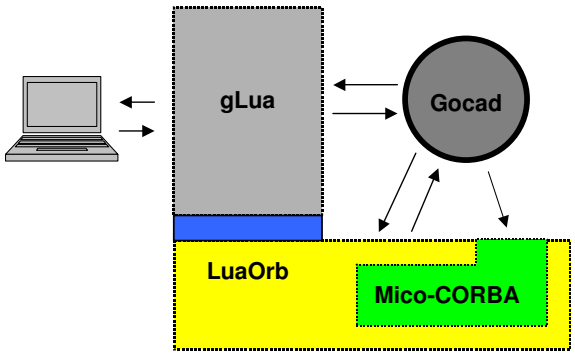


FIG. 3.18 – Interaction entre le plugin Lua et Gocad à chaque site.

Chapitre 4

Application

4.1 Introduction

Dans ce chapitre, nous présentons des exemples d'applications du système dans différentes situations: une opération réelle de guidage de puits et quelques situations de test, reproduisant des cas d'usage analysés au paragraphe 1.6.

Jusqu'à présent, des versions intermédiaires du système ont été opérationnellement développées; cependant, la plupart de nos tests impliquant des conférences avec des participants multiples ont été restreints à une situation de recherche, à la division de Technologie de Réservoir du Centre de Recherche de Petrobras. Un point intéressant est que ce groupe est responsable des études intégrées de modélisation de réservoir, et par conséquent il comprend des professionnels de toutes les différentes disciplines impliquées dans le flux de travaux de réservoirs (géophysique, géologie, ingénierie de réservoirs et ingénierie de puits). Aussi, comme les études sont entreprises en coopération avec des unités opérationnelles qui sont dispersées géographiquement dans le pays, la consultation à distance est une activité de routine dans le groupe.

Créer les conditions pour la conduite d'évaluations systématiques avec un système expérimental dans les conditions opérationnelles chargées de compagnies de pétrole et de gaz n'est pas simple. Ceci est vrai pour les systèmes mono-utilisateur, et spécialement difficile pour les systèmes coopératifs dont tester présuppose la participation de différentes personnes à différents sites. Pour la planification et le guidage de puits, étant donné la manière critique en temps dans laquelle les opérations sont menées et les très hauts investissements engagés, interrompre le travail pour tester un système de recherche pendant des opérations réelles n'est pas possible. Obtenir l'autorisation pour la révélation des résultats opérationnels est aussi très difficile, voire impossible, dans l'environnement des affaires hautement confidentielles des compagnies de pétrole.

Pour ces raisons, excepté le premier cas de guidage de puits, où les détails du modèle réel ne sont pas montrés, nous présentons ici des exemples de cas simulés (mais basés sur des données réelles). Le modèle utilisé représente un réservoir réel, déjà libéré pour la publication, mais est affiché avec des noms et des coordonnées spatiales fictifs. Les participants des tests sont des spécialistes du groupe de Technologie de Réservoir.

Les résultats présentés sont uniquement qualitatifs. A partir de maintenant notre intention est de mener des évaluations formelles contrôlées du système dans des conditions les plus proches possibles des opérationnelles. Pour cela, des cas de simulation ont été préparés en utilisant des données post mortem à partir des opérations réelles de forage de puits.

4.2 Géoguidage 1

Comme analysé au paragraphe1.4.3, pendant le forage d'un puits les données acquises du bout du puits doivent être envoyées au bureau, d'où l'opération est contrôlée, pour supporter la prise de décision concernant le guidage de sa trajectoire. Le géologue responsable pour l'opération locale à la tour de forage reçoit les instructions du bureau concernant les actions à mener (corrections de la trajectoire du puits, opérations de diagraphie, etc.).

Dans ce but, NetGocad a été employé systématiquement dans un nombre d'opérations réelles chez Petrobras, en combinaison avec le plugin gWLog, qui rajoute de la fonctionnalité à Gocad pour lui permettre d'agir comme un système de surveillance à distance (paragraphe3.6). L'utilisateur connecté directement au serveur est un géoscientiste supervisionnant l'opération à partir du bureau, qui a le contrôle sur l'édition du modèle. L'opérateur à la tour de forage, connecté au bureau à travers NetGocad, reçoit automatiquement les actualisations comme si elles étaient des opérations ordinaires réalisées par le contrôleur de conférence.

Le spécialiste utilise alors la synchronisation de la caméra 3D et les annotations pour discuter du progrès de l'opération, analyser les données reçues, et discuter des décisions prises. Les modifications de la trajectoire du puits ou du modèle définies par le géoscientiste responsable sont automatiquement vues par l'opérateur, qui ensuite a une meilleure compréhension des instructions reçues qu'habituellement, sans ce type de ressource.

Dépendant du type de lien de réseau disponible au site de forage de puits, une visioconférence peut ou non être utilisée. Pour beaucoup de sites sur terre, les connexions de mise en réseau disponibles permettent l'utilisation audio et vidéo à la fois. Pour des endroits distants et au large, souvent uniquement des connexions satellite limitées et chères sont disponibles, et ensuite uniquement l'audio peut être autorisée, ou pas du tout de communication audio/vidéo sur IP (uniquement des chats textuels et des conversations téléphoniques ordinaires).

L'utilisation de la politique simple décrite est satisfaisante pour ce type de collaboration face à face: l'édition est bloquée à l'opérateur, tous les autres canaux sont libres. La négociation pour le contrôle de la caméra est faite pendant la discussion par le canal audio. La liberté pour une manipulation simultanée du modèle (au contraire de l'établissement d'un contrôle de tours pour l'utilisation de la caméra) est intéressante, étant donné qu'elle donne l'impression que les deux participants sont réellement en train d'interagir avec l'objet partagé, même si en fait ils se disputent le contrôle. Avec deux utilisateurs seulement et une bonne communication audio, la négociation est plutôt aisée (mais l'implémentation de la saisie et de la libération automatique du contrôle d'utilisation basée sur l'activité d'interaction devrait être considérée, pour améliorer la dynamique de la collaboration).

L'utilisation du système améliore considérablement la collaboration entre le bureau et la tour de forage. Elle apporte aussi une grande satisfaction aux opérateurs qui jusqu'à maintenant étaient restreints au rôle du suivi de commandes aveugles, sans la compréhension claire de la situation de subsurface donnée par une visualisation interactive 3D.

FIG. 4.1 – Application du système pour la surveillance du forage de puits au site de forage.

4.3 Géoguidage 2

Dans ce cas d'usage NetGocad est utilisé dans une situation simulée de géoguidage. Pour le type de conférence *wellplanning* (planification de puits) trois rôles sont définis: *leader* (géoscientiste chef d'équipe, responsable des décisions principales), *geoscientist*

(géoscientiste, responsable principal des analyses techniques), et *operator* (opérateur, responsable des actions à la tour de forage). Eventuellement d'autres géoscientistes pourraient aussi participer à la session (un rôle spécifique peut être défini pour les observateurs avec aucun droit de contrôle).

Dans cet exemple, les canaux de collaboration reflètent les réglages définis pour ce type de conférence au paragraphe 3.3.4. Remarquez que l'opérateur a initialement le canal vidéo automatiquement éteint (Figure 4.5), en raison de ses restrictions de mise en réseau dans ce cas d'usage. Les canaux de collaboration de commande et de caméra sont initialement sous contrôle d'utilisation, ainsi uniquement le leader, qui est le contrôleur, peut émettre des commandes et manipuler la caméra (et passer le contrôle aux autres participants). Comme propriétaire de la conférence, il peut aussi établir des *floors* pour chaque canal et prendre ou passer le contrôle à tout moment. Les annotations et les télépointeurs sont initialement libres. L'audio est aussi ouvert à tous les participants.

Ci-après nous présentons des images des écrans des trois participants. Remarquez que le géologue utilise un système à deux écrans, qui fournit assez d'espace pour l'affichage d'autres types d'informations généralement employées (ici, une seconde caméra Gocad non synchronisée) et pour une plus grande fenêtre de visioconférence.

Leader

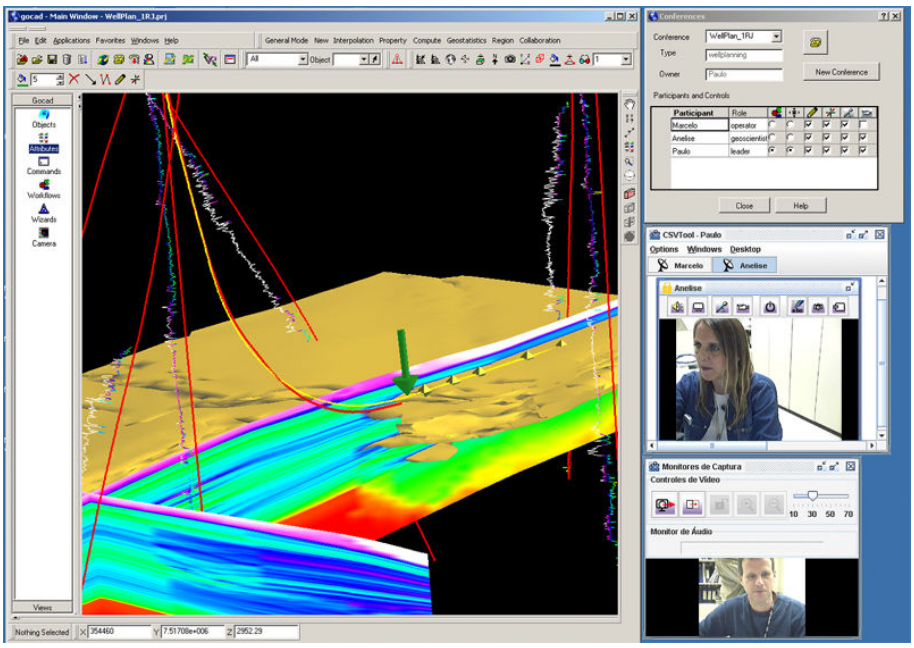


FIG. 4.2 – Interface de l'utilisateur du leader (remarquez le moniteur de capture dans le coin inférieur droit). La communication vidéo avec l'opérateur n'est pas active. Le leader utilise une flèche verte pour annoter le modèle et discuter du problème avec le groupe: le puits perforé (rouge) est dévié du puits planifié (jaune) et touche une formation de roche schisteuse, qui doit être évitée.

Géologue

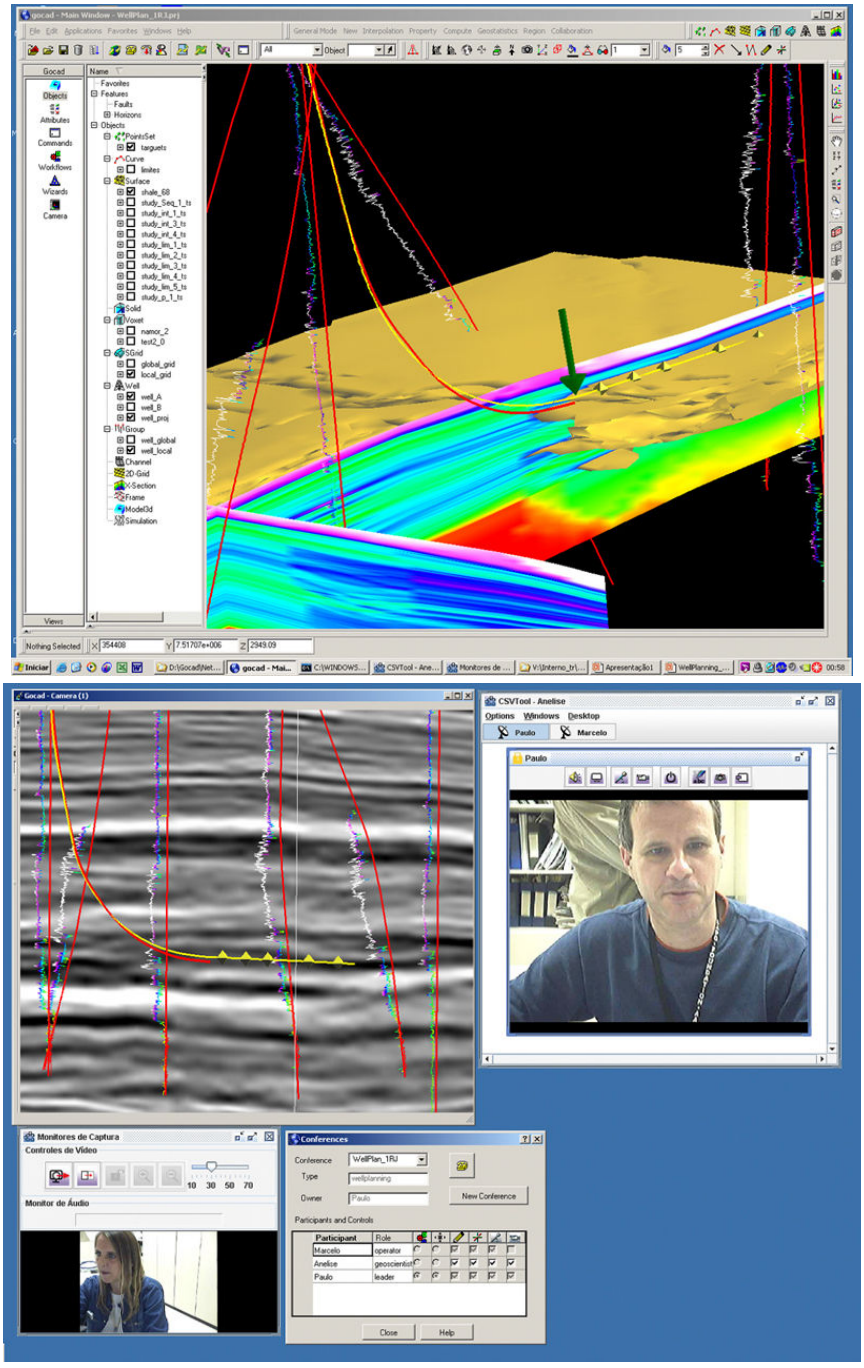


FIG. 4.3 – Interface utilisateur à deux écrans du géologue (remarquez le moniteur de capture dans le coin inférieur gauche ; voir aussi la Figure 4.4).

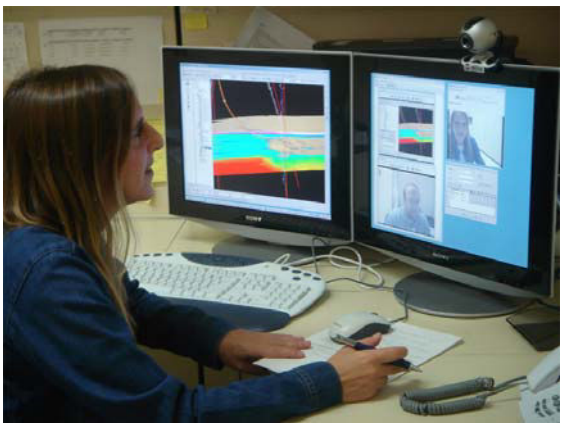


FIG. 4.4 – Géologue utilisant une interface utilisateur à deux écrans.

Opérateur

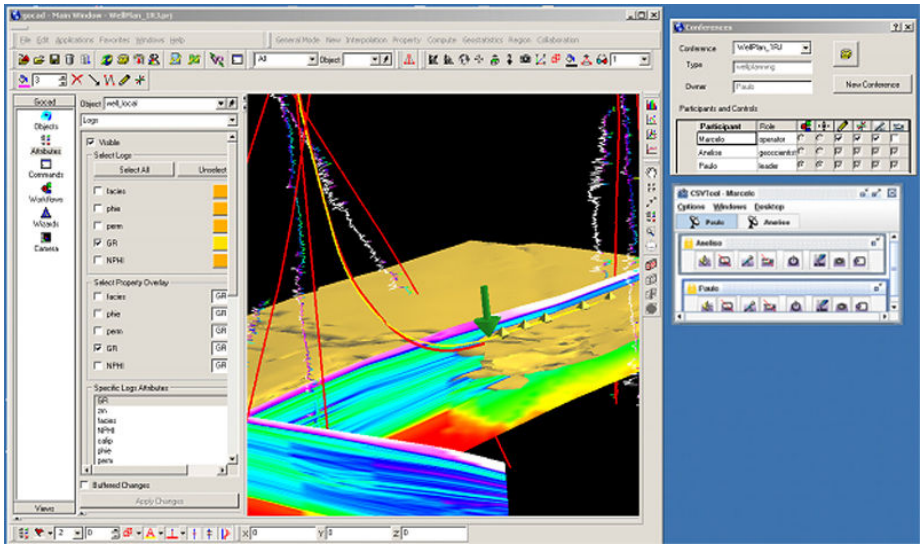


FIG. 4.5 – Interface utilisateur de l'opérateur, avec flux vidéo non actifs (uniquement audio).

4.4 Formation

Dans ce cas d'usage de formation, le *teacher* (instructeur) est le propriétaire de la conférence, et a le contrôle d'utilisation sur tous les canaux utilisés, sauf audio et vidéo. Des *students* (apprentis) ne sont pas autorisés à envoyer de l'audio ou vidéo aux autres (voir Listage 3.4, paragraphe 3.3.5). L'instructeur envoie parfois son écran capturé sur le canal vidéo, au lieu de l'image capturée par la caméra; il peut opter d'envoyer tout l'écran compressé, ou juste une partie, avec une meilleure résolution (comme sur la Figure 4.6).

Leader

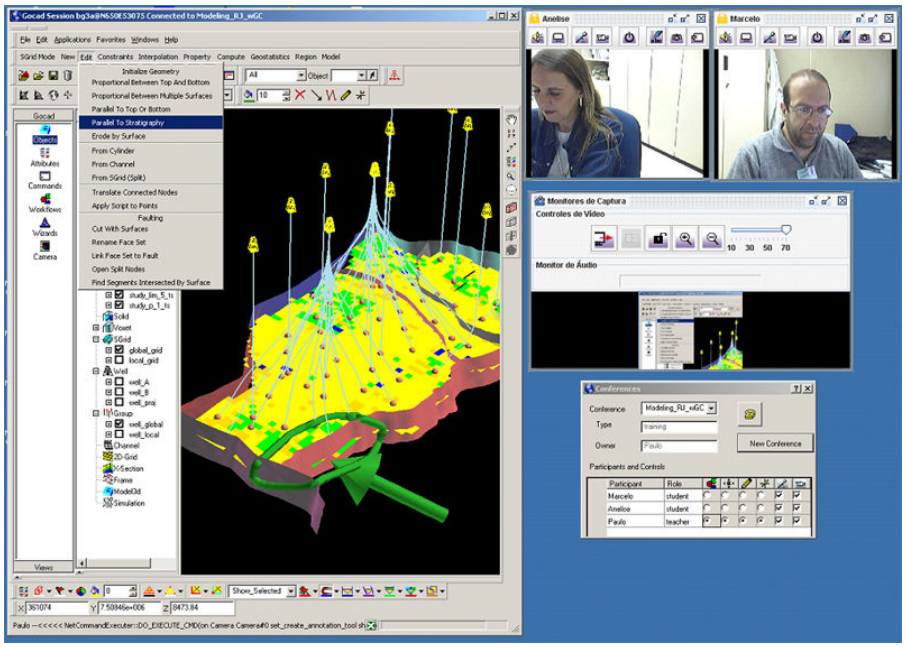


FIG. 4.6 – Un cas d'usage de cours. L'instructeur est en train d'envoyer une partie de son écran capturé (une fenêtre autour de la position de la souris) à travers le flux vidéo (remarquez le moniteur de capture de visioconférence) pour montrer aux apprentis comment manipuler l'interface utilisateur pour réaliser une certaine opération. Il utilise aussi des annotations pour indiquer un problème sur le modèle.

Géoscientiste

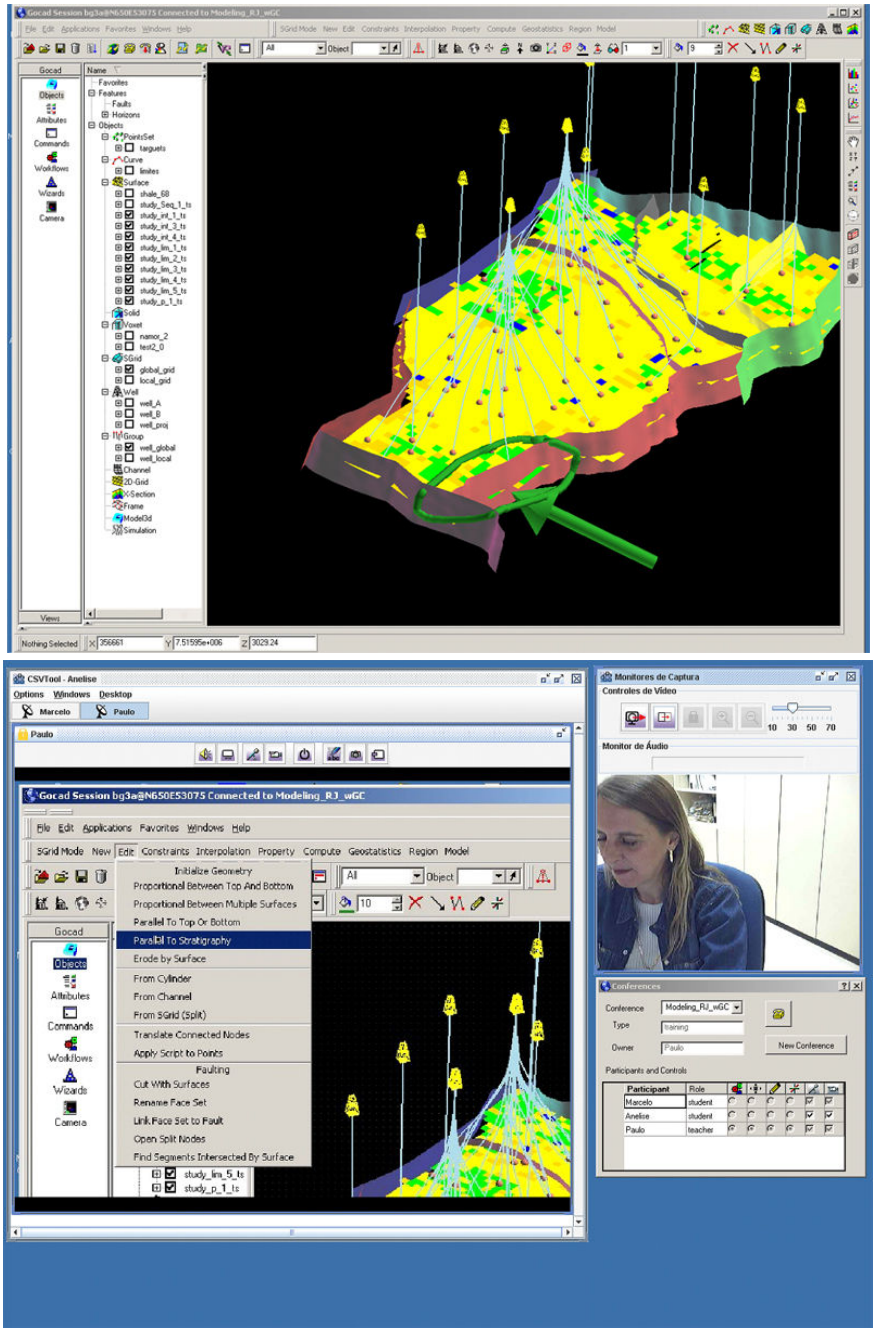


FIG. 4.7 – Géoscientiste recevant l'écran capturé du leader à travers le flux vidéo, et voyant ses annotations dans la caméra 3D.

4.5 Ordinateur de bureau - réalité virtuelle

Dans ce cas d'usage, nous montrons une collaboration entre des utilisateurs d'ordinateur de bureau et de réalité virtuelle. L'idée est de permettre au géoscientiste d'utiliser l'environnement d'immersion pour discuter d'une opération simulée de guidage de puits avec un collègue à distance utilisant une interface d'ordinateur de bureau régulière (dont l'écran est montré sur la Figure 4.8). Une caméra ouverte à la session de bureau est utilisée pour l'échange d'annotations et de télépointeurs avec l'environnement immersif, où l'utilisateur RV peut analyser le modèle en utilisant une navigation *fly-through*.

En plus de la communication audio, une vidéo permet à l'utilisateur au bureau d'observer le travail dans l'environnement immersif. La qualité d'image n'est pas très bonne en raison de la visualisation stéréoscopique et des conditions généralement sombres dans la pièce (spécialement avec l'utilisation de projecteurs CRT). Dans l'exemple, la vidéo n'est pas envoyée du bureau à l'utilisateur RV (une fenêtre vidéo ordinaire pourrait être affichée par-dessus la scène 3D dans l'environnement RV, mais ceci empêche le sens d'immersion).

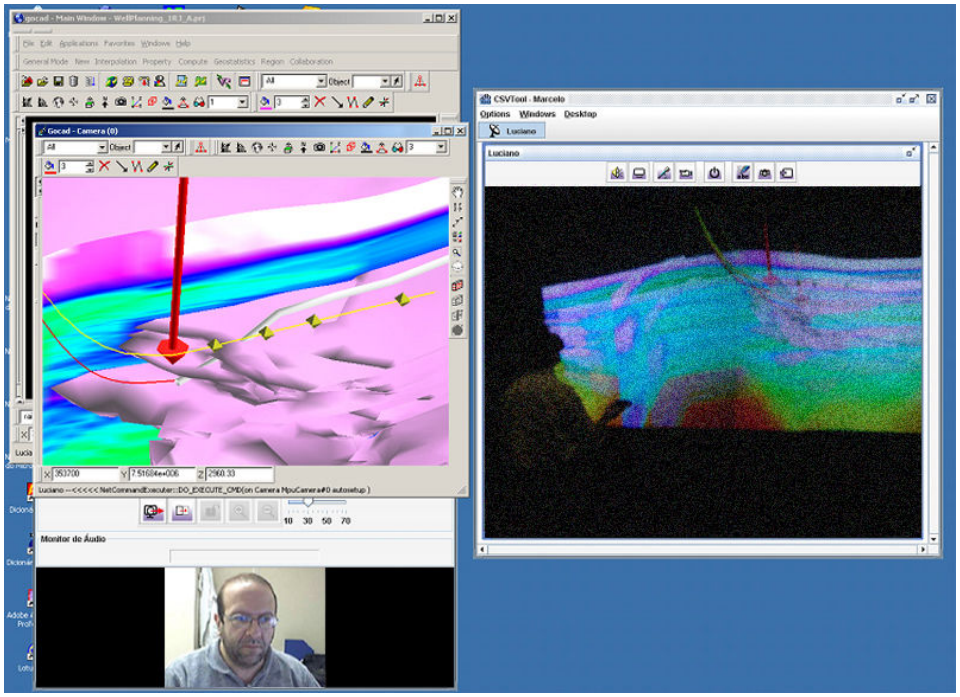


FIG. 4.8 – Ecran d'opérateur pendant la collaboration avec l'utilisateur RV. Le participant RV voit l'annotation de l'opérateur (flèche rouge).

Comme analysé au paragraphe 3.5, le plugin Go2VR est actuellement fonctionnel pour les buts de visualisation, mais ses fonctionnalités de modélisation et collaboration à distance sont encore au stade de prototype. Des outils d'interaction sont disponibles pour la création d'annotations et pour la mise en forme d'objets simples tels que les points, courbes et surfaces, mais seule une version préliminaire de manipulateurs pour l'interaction avec les puits a été implémentée. D'autres outils sont nécessaires pour la manipulation interactive de volumes sismiques, cibles et diagraphies de puits, sections, etc. La caméra synchronisée avec l'environnement RV doit aussi être améliorée (paragraphe 3.5.2).

Comme nous l'avons analysé auparavant, considérant que l'interface réalité virtuelle peut apporter de réels avantages pour la conception et l'analyse de trajectoires spéciales de puits [Gru04], nous pensons que ce modèle hétérogène de travail présente des perspectives prometteuses pour la collaboration à distance. Cependant, pour qu'une évaluation formelle soit possible et pour son utilisation en conditions opérationnelles, la fonctionnalité du système doit être complétée.

4.6 Discussion

Jusqu'ici l'application du système, soit opérationnellement ou en conditions de test, a reçu une impression positive des utilisateurs, qui en général n'ont pas été préalablement exposés aux outils de modélisation collaborative synchrone. Cependant, jusqu'à présent, les évaluations ont été uniquement qualitatives, et montrent aussi que beaucoup de caractéristiques doivent être améliorées. Par exemple, une certaine confusion avec l'utilisation des contrôles arrive souvent pour le composant de la visioconférence seul. Peut-être trop d'informations (conscience de groupe excessive) sont proposées maintenant concernant l'état des connexions (paragraphe 3.4.4.3), ou peut-être c'est une question de conception d'interface.

Pour une collaboration face à face, les utilisateurs préfèrent en général travailler avec des contrôles d'utilisation libres (*free floors*) pour tous les canaux et employer une communication directe pour éviter les conflits (qui rend l'opération plus simple puisqu'ils n'ont pas besoin de s'inquiéter du contrôle de canaux). Ne pas employer le contrôle d'utilisation pour le canal de commande, cependant, peut mener aisément à des problèmes de consistance, à moins que les participants n'acceptent clairement leurs droits d'utilisation d'opérations d'édition. Dans les conférences avec plus de participants, le contrôle d'utilisation pour les commandes semble essentiel pour éviter les conflits et les malentendus concernant les opérations non perçues réalisées par les autres. Néanmoins, le contrôle d'utilisation est actuellement très restrictif, étant donné que toutes les opérations sont bloquées pour les participants sans le contrôle (*floor*). Nous avons besoin d'étudier plus loin l'utilisation de mécanismes pour permettre plus de travail concurrent, comme la création de canaux séparés pour différents types de

commandes (paragraphe 3.3.2.1) ou de mécanismes pour le verrouillage automatique d'objets et d'outils (paragraphe 2.5.2).

A partir de maintenant nous envisageons d'utiliser des méthodes d'évaluation pour pouvoir mesurer l'utilisabilité d'éléments individuels. Nous envisageons de mener les évaluations du système à différents stades, en commençant par la communication audio et vidéo uniquement, ensuite la collaboration sur ordinateurs de bureau avec des types différents de conférence, et finalement des sessions de collaboration impliquant des utilisateurs de réalité virtuelle. Une première étude d'évaluation pour le composant de visioconférence est en train d'être menée avec le Groupe de Recherche d'Ingénierie Sémiotique à la PUC-Rio [SERG]. Les évaluations de collaboration avec des types hétérogènes d'interfaces demanderont l'étude et l'adaptation des méthodes d'évaluation pour collecticiel [PG02, BGG01, GDS99], un domaine de recherche active avec peu de méthodes établies [PB03].

Nous avons travaillé sur la préparation de cas de test réalistes avec des données réelles enregistrées pendant des opérations de forage de puits, avec le but de simuler des situations demandant une prise de décision collaborative et ensuite de comparer les performances de différents groupes, pour accéder aux bénéfices potentiels de sessions hétérogènes de collaboration impliquant des utilisateurs d'ordinateur de bureau et de réalité virtuelle pour la planification de puits. Ce type de travail pose beaucoup de questions intéressantes; cependant, une évaluation formelle est difficile [TSW03, Ste02] et demandera encore l'amélioration préalable de nos outils, comme analysé auparavant.

Conclusion

Le développement de systèmes coopératifs est un effort difficile, impliquant de nombreuses questions au-delà de la fonctionnalité fournie par les applications mono-utilisateur de base. Ce fait, conjointement aux aspects sociaux impliqués dans leur introduction dans les situations opérationnelles, contribue à faire une collaboration à distance efficace encore rare dans la plupart des domaines techniques.

Dans le cas spécifique de la modélisation et de la visualisation en géosciences, les applications opérationnelles sont le résultat de grands efforts de développement, menés au long de plusieurs années, et fournissent des fonctionnalités de domaine relatives à de multiples disciplines. L'utilisation des boîtes à outils et d'environnements-cadres disponibles pour supporter le développement des fonctionnalités de collaboration devient impraticable, parce qu'elle demande la création d'applications conformes aux architectures prédéfinies.

Dans cette thèse, nous avons proposé une architecture qui permet la transformation d'une application existante en un système coopératif exclusivement à travers la composition dynamique de modules d'extension (*plugins*). Cette architecture supporte l'intégration à l'application de services de coopération, d'une communication multimédia fournie à travers un composant développé sur mesure, et d'un schéma de coordination extensible basé sur les rôles. Les mécanismes de diffusion et de contrôle d'utilisation (*floor control*), conçus pour les canaux de collaboration et de communication introduits, tiennent compte des exigences de modélisation et de visualisation dans les géosciences. Nous considérons aussi un modèle de travail dans lequel les sessions de collaboration peuvent impliquer l'utilisation d'interfaces d'utilisateur hétérogènes par des participants à distance (ordinateur de bureau et réalité virtuelle).

Une partie considérable de ce travail concernait l'analyse des besoins particuliers pour la collaboration synchrone à distance dans ce domaine d'application, basé sur une étude de champ étendue d'activités collaboratives, comme analysé au chapitre 1. Les caractéristiques des artéfacts partagés (modèle de géoscience 3D) ont été considérées, ainsi que les processus de travail coopératif, les cas d'usage des applications cibles, les besoins de l'utilisateur et du programmeur et leurs conséquences pour l'architecture proposée.

Au chapitre 2, la solution a été caractérisée dans le contexte du champ de Travail Coopératif Assisté par Ordinateur (TCAO) et de stratégies alternatives de collaboration à distance. Pour la géomodélisation collaborative, différentes formes de collaboration sont en fait complémentaires. Les solutions à collaboration explicite avec des applications répliquées, comme celle proposée ici, sont essentielles pour permettre un travail indépendant aux participants d'une session, et aussi en conditions de largeur de bande de réseau limitée. Les solutions à collaboration implicite, d'un autre côté, pourraient être intéressantes pour permettre l'utilisation partagée d'applications mono-

utilisateur standard et d'autres ressources centralisées (ensembles de données très grands, processeurs graphiques, licences de l'application), mais demandent une mise en réseau à haute largeur de bande à tous les participants et fournissent une conscience de groupe limitée (à travers des mécanismes externes à l'application). De toute façon, l'utilisation combinée des techniques de flux audio et vidéo est essentielle pour la communication personnelle et aussi pour la communication basée sur la vidéo en général (par ex., pour permettre l'observation d'un site à distance, le partage d'écran, expliquer le fonctionnement du système, et pour d'autres formes avancées de téléprésence).

Dans ce chapitre, nous avons aussi analysé les caractéristiques de l'architecture du système à partir d'un point de vue conceptuel, et des questions considérées importantes comme le contrôle d'utilisation, le contrôle de concurrence et la conscience de groupe. Nous avons examiné en particulier les difficultés pour fournir un contrôle de concurrence en géomodélisation collaborative.

Les solutions proposées ont été décrites en détails au chapitre 3. Profitant des motifs de conception suivis par l'application de base, nous avons introduit une architecture simple de plugins basée sur CORBA et un schéma de contrôle unifié pour traiter les différents "canaux de collaborations" employés. Pour chaque type de canal différent (commandes, interaction graphique et flux media), les exigences de communication ont été identifiées et des schémas spécifiques de diffusion et de contrôle ont été proposés, d'une manière intégrée. Un modèle a été défini pour la spécification de droits de contrôle sur les canaux, selon les types spécifiques de collaboration et les rôles des participants. L'implémentation utilise un langage interprété dynamiquement typé, léger et extensible avec une syntaxe très simple (Lua).

L'utilisation d'un composant de visioconférence multiplateformes sur mesure (paragraphe 3.4) permet une étroite intégration dans le système de flux multiples audio et vidéo avec un contrôle souple selon les politiques définies. Une attention spéciale a été donnée à la fourniture des éléments de conscience de groupe concernant l'état de toutes les connexions selon les droits et intentions des utilisateurs, pour un emploi efficace des ressources de mise en réseau.

L'intégration d'un plugin de réalité virtuelle permet l'exécution de sessions hétérogènes impliquant l'utilisation de différents types d'interfaces d'utilisateur par différents participants, selon leurs tâches spécifiques. Nous croyons que ce type de collaboration a un grand potentiel pour améliorer la productivité dans certaines activités.

Dans le processus de développement, des versions intermédiaires du système ont été développées, et sont actuellement en utilisation opérationnelle. A partir de cette expérience il est clair que la collaboration à distance est en fait une technologie prometteuse et nécessaire, qui une fois effective aura une grande diffusion et deviendra une pratique standard dans ces activités.

En ce moment, quelques aspects de l'implémentation n'ont pas encore atteints le stade d'être formellement évalués et utilisés opérationnellement, compte tenu des implications et coûts élevés de telles évaluations dans des situations opérationnelles.

Par conséquent, comme futur travail nous avons un programme de recherche considérable à poursuivre. Nous devons améliorer la fonctionnalité du plugin de collaboration, comme analysé aux chapitres 3 et 4, et aussi sa robustesse. D'autres importantes caractéristiques attendues de systèmes coopératifs ont besoin d'être considérées à plus long terme, comme les sessions de travail privées, la gestion de versions, les mécanismes d'annulation multi-utilisateurs, la vérification de consistance et le support pour le travail asynchrone, entre autres.

Nous devons compléter la fonctionnalité de modélisation du composant de réalité virtuelle, et aussi rechercher l'intégration d'un système haptique, à travers une coopération avec d'autres groupes de recherche ou entreprises, pour être capables d'exploiter efficacement les possibilités intéressantes du modèle de collaboration impliquant l'utilisation d'interfaces d'utilisateur hétérogènes. L'hypothèse que ce type de travail peut apporter des avantages opérationnels doit être questionnée avec l'utilisation de méthodologies d'évaluation de collecticiel (un domaine de recherche active, étant donné que peu de méthodes établies existent même pour les systèmes coopératifs conventionnels). Comme analysé au chapitre 4, nous commençons à mener des évaluations du système par étapes, en utilisant des cas de test basés sur des données réelles et sur différentes prises de décision collaboratives.

Finalement nous envisageons d'introduire des fonctionnalités de collaboration à distance dans d'autres applications internes en utilisant les principes et les méthodes analysées. Nous attendons de bonnes opportunités pour exploiter ces possibilités dans le cadre de projets continus au Centre de Recherche de Petrobras pour le développement de technologies de visualisation et collaboration à distance appliquées aux géosciences et à l'ingénierie, basées sur la perspective que les systèmes coopératifs appliqués à la modélisation et à la visualisation auront un important impact sur les processus de travail coopératif dans le secteur en amont de l'industrie du pétrole et du gaz.

Bibliographie

[AB94] V. Anupan and C. Bajaj. Shastra: An Architecture for Development of Collaborative Applications. *IEEE Multimedia*, Vol. 1, Number 2, 39-49, 1994.

[Ack01] M. Ackerman. The Intellectual Challenge of CSCW: The Gap Between Social Requirements and Technical Feasibility, *Human Computer Interaction*, 2001.

[ARH02] A. Agrawal, K. Ramani, and C. Hoffmann. CADDAC: Multi-Client Collaborative Shape Design System with Server-Based Geometry Kernel. In *Proceedings of ASME 2002 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2002.

[BF91] E.A. Bier and S. Freeman. MMM: A user interface architecture for shared editors on a single screen. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, UIST'91, 79–87, November 1991.

[BFD+03] C.A.M. Barbosa, B. Feijó, M. Dreux, R.N. Melo, J. Bento and S. Scheer. An Object Model for Collaborative CAD Environments. *Journal of Integrated Design and Process Science*, V7, N2, 2003

[BGG01] K. Baker, S. Greenberg, C. Gutwin. Heuristic Evaluation of Groupware Based on the Mechanics of Collaboration. In *Proc. 8th IFIP Working Conference on Engineering for HCI* (EHCI), 2001.

[BIS+99] L. Bricker, K. Inkpen, J. Stewart, B. Myers, and S. Poltrock. Single Display Groupware: Exploring Computer Support for Co-Located Collaboration. Panel submitted to ACM CHI'99, 1999

[Bla99] B. Blakley, *CORBA Security: An Introduction to Safe Computing with Objects*, Addison Wesley, 1999

[BMC04] M. Bolas, I. McDowall, D. Corr. New Research and Explorations into Multiuser Immersive Display Systems. *IEEE Computer Graphics and Applications*, vol. 24, no. 1, 18-21, January/February 2004.

[Bro99] F. Brooks, What's Real About Virtual Reality, *IEEE Computer Graphics and Applications*, Vol. 19, N. 6, 16-27, Nov./Dec. 1999.

[Cam02] J. L Campos. Real-Time Well Drilling Monitoring using Gocad. In *Proceedings of the 22nd Gocad Meeting*. INPL/ENSG, France, 2002.

[CAVS] Collaborative AVS, In http://www.tacc.utexas.edu/cavs/welcome.html

[CCI99] R. Cerqueira, C. Cassino, R. Ierusalimschy, Dynamic Component Gluing Across Different Componentware Systems. In *International Symposium on Distributed Objects and Applications* (DOA'99), 362-371, IEEE Press, 1999

[CGS+96] J. Coleman, John, A. Goettsch, A. Savchenko, H. Kollmann, K. Wang, E. Klement, and P. Bono. "TelelnViVoTM: towards collaborative volume visualization environments". Computers & Graphics, 20(6), 801-811, 1996

[CHWS88] J. Callahan, D. Hopkins, M. Weiser and B. Shneiderman. An empirical comparison of pie vs. linear menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM Press, 95-100, 1988

[CM96] M. Cortez and P. Mishra. DCWPL : A programming language for describing collaboration work. In *ACM Conference on Computer Supported Cooperative Work*, (CSCW'96), Novembre 1996.

[CSM03] G. Caumon, C. H. Sword, and J.L. Mallet. Constrained modifications of non-manifold b-reps. In *8th ACM Symposium on Solid Modeling and Applications*, June 2003.

[CR03] J. L Campos and L. P. Reis. gWLog – Ferramenta para Monitoramento de Perfuração de Poços em Tempo Real (gWLog - A tool for the survey of well drilling in real time). ConnecTI - Petrobras Information Technology Conference, 2003

[DB92] P. Dourish and V. Bellotti. Awareness and coordination in shared workspaces. In *Proceedings of the 1992 ACM Conference on Computer-Supported Cooperative Work* (CSCW '92), ACM Press, New York, NY, 107-114, 1992

[Dew99] Dewan, P. Architectures for collaborative applications. In *Computer Supported Co-operative Work*, M. Beaudouin-Lafon, Ed., vol. 7 of Trends in Software. John Wiley & Sons, 169-193, 1999.

[DFL+00] A. van Dam, A.S. Forsberg, D.H. Laidlaw, J.J. LaViola, and R.M. Simpson. Immersive VR for Scientific Visualization: A Progress Report.

In *IEEE Computer Graphics and Applications*, Vol. 20, No. 6, 26-52, Nov./Dec. 2000..

[DG97] H.P. Dommel and J.J. Garcia-Luna-Aceves. Floor,Control for Multimedia Conferencing and Collaboration. In *ACM Multimedia*'97, 5(1), Jan. 1997.

[DGI] Dynamic Graphics Incorporated, Earth Vision. In http://www.dgi.com/earthvision/index.shtml

[DHJ+00] M. Daily, M. Howard, J. Jerald, C. Lee, K. Martin, D. McInnes, and P. Tinker. Distributed design review in virtual environments. In *Proceedings of the Third international Conference on Collaborative Virtual Environments (CVE '00)*. E. Churchill and M. Reddy, Eds. ACM Press, 57-63, 2000.

[Edw94] W. K. Edwards. Session management for collaborative applications. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (CSCW '94). ACM Press, New York, NY, 323-330, 1994

[Edw96] W. K. Edwards. Polices and Roles in Collaborative Applications. In: *Proceedings of the ACM Conference on Computer Supported Cooperative Work,* 11-20, 1996.

[EG89] C.A. Ellis and S. J. Gibbs. Concurrency Control in Groupware Systems. In *SIGMOD Conference*, vol. 18, 399-407, 1989

[EGR91] C.A. Ellis, S. J. Gibbs, and G. L. Reins. Groupware: Some Issues and Experiences. *Communications of the ACM*, vol 34, no. 1, January 1991.

[Ehr99] K. Ehrlich. Designing Groupware Applications. In M.Beaudouin-Lafon, (ed.): Computer Supported Co-operative Work, Vol. 7, Trends in Software. John Wiley & Sons, 1-28, 1999.

[Ell99] C.A. Ellis, , Workflow Technology. In *Computer Supported Co-operative Work*, M. Beaudouin-Lafon, Ed., vol. 7 of Trends in Software. John Wiley & Sons, 29-54, 1999.

[EW94] C. A. Ellis and J. Wainer: A Conceptual Model of Groupware. In *Proceedings of the Conference on Computer Supported Cooperative Work* (CSCW'94), ACM Press, 79-88, 1994.

[EVD+02] F. Evans, W. Volz, G. Dorn, B. Fröhlich, and D. M. Roberts.. Future trends in oil and gas visualization. In *Proceedings of the Conference on Visualization '02*, IEEE Computer Society, 567-570, 2002.

[FBM02] C. Ferraris, P. Brunier, C. Martel. Constructing collaborative pedagogical situations in classrooms : a scenario and role based approach. *Computer Support for Collaborative Learning 2002 (CSCL 2002)*, January 2002

[FTK95] G. Fitzpatrick, W. J. Tolone and S. M. Kaplan. Work, Locales and Distributed Social Worlds. *Proceedings of the ECSCW'95*, September, 1-16, 1995

[FIC96] L.H Figueiredo., R. Ierusalimschy, W. Celes. Lua: an extensible embedded language. *Dr. Dobb's Journal*, volume 21, number 12, 26-33, 1996

[FT00] J. A. Friesen and T.D. Tarman, "Remote High-Performance Visualization and Collaboration", IEEE Computer Graphics and Applications, Vol. 20, No. 4, Jul./Aug., 45-49, 2000.

[GCL89] J.J. Garcia-Luna-Aceves, E.J. Craighill, R. Lang. Floor management and control for multimedia computer conferencing. In *Proceedings of the 2nd IEEE Comsoc International Communications Workshop*, Ottawa, Canada, 1989

[GDG] Gocad Developer's Guide, Earth Decision Sciences.

[GDS99] J. L. Gabbard, D. Hix and J.E. Swan, User-Centered Design and Evaluation of Virtual Environments. *IEEE Computer Graphics and Applications*, Vol. 19, N. 6, 51-59, Nov./Dec. 1999.

[GHJV95] E. Gamma, R. Helm, R. Johnson, and J. Vlissiedes. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison Wesley, 1995

[GM94] S. Greenberg and D.Marwood. Real time groupware as a distributed system: Concurrency control and its effect on the interface. In *Proceedings of the ACM CSCW'94 Conference on Computer Supported Cooperative Work*, 207–217, October 22–26, 1994

[GOC] GOCAD Research Consortium, In http://www.gocad.org

[GO2] Go2VR, VSP Technology. In http://www.vsp-technology.fr/en/go2vrsoftware_us.html

[GO4] Go4Space, VSP Technology. In http://www.vsp-technology.fr/en/go4spacesoftware_us.html

[GCDV96] J. Gomes, B. Costa, L. Darsa, and L. Velho. Graphical objects. *The Visual Computer*, 12(6):269-282, 1996.

[GPS04] C.Gutwin, R.Penner, and K. Schneider. Group awareness in distributed software development. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work (CSCW '04)*, ACM Press, NY, 72-81, November 2004

[GR99] S.Greenberg and M. Roseman, Groupware toolkits for synchronous work. In *Computer Supported Co-operative Work*, M. Beaudouin-Lafon, Ed., vol. 7 of Trends in Software. John Wiley & Sons, 135-168, 1999.

[Gre91] Saul Greenberg. C*omputer-supported Cooperative Work and Groupware*. Academic Press, 1991.

[Gre99] Chris Greenhalgh. *Large Scale Collaborative Virtual Environments*. London: Springer, 1999.

[GRO] Groove. In http://www.groove.net

[Gru92] J. Grudin. Why CSCW applications fail: problems in the design and evaluation of organizational interfaces. In *Groupware: Software for ComputerSupported Cooperative Work*, D. Marca & G. Bock, eds, ' IEEE Press, 552-560, 1992

[Gru94a] J. Grudin. CSCW: History and Focus, Comm. ACM, may 1994

[Gru94b] J.Grudin. Computer-supported cooperative work: Its history and participation. IEEE Computer 27(5) 19-26, 1994

[Gru04] K. Gruchalla. Immersive Well-Path Editing: Investigating the Added Value of Immersion. *IEEE Virtual Reality 2004,* 157-164, 2004.

[GSGP04] P. Gore, D. Schmidt, C. Gill and I. Pyarali. The Design and Performance of a Real-Time Notification Service. In *Proceedings of the 10th IEEE Real-Time Technology and Applications Symposium*, May 2004

[Har04] C. Harding, 2004. Modeling Geoscience Data in a Multisensory Virtual Environment. Computing in Science and Engineering. 6, 1, 89-92, Jan. 2004.

[HBRP94] R. D.Hill, T. Brinck, S.L. Rohall, J.F. Patterson, and W. Wilner. The Rendezvous architecture and language for constructing multiuser applications. *ACM Transactions on Computer-Human Interaction* 1, 2, 81-125, 1994

[HSFP99] G. Hesina, D.Schmalstieg, A.Fuhrmann, and W.Purgathofer. Distributed open inventor: A practical approach to distributed 3d graphics. *In Virtual Reality Software & Technology '99 (VRST'99)*, ACM, 1999.

[HV99] M. Henning and S. Vinoski, *Advanced CORBA Programming with C++,* Addison Wesley, 1999

[IDPa] Immersive Drilling Planner, BP Center for Visualization, Univeristy of Colorado at Boulder. In http://www.bpvizcenter.com/Research_Consortia/Research_ImmersiveDrilling.php

[IDPb] Drilling Planner, Earth Descision Sciences. In http://www.earthdecision.com/products/drillingplanner.html

[Ier03] R. Ierusalimschy, *Programming in Lua,* Lua.org, 288 pages, December 2003

[IFC96] R. Ierusalimschy, L. Figueiredo, and W. Celes. Lua – an extensible extension language. *Software: Practice and Experience*, 26(6), 1996.

[IR] Inside Reality, Schlumberger Limited. In http://www.oilfield.slb.com/content/services/software/virtual/index.asp

[Ish99] H. Ishi. Integration of shared workspace and interpersonal space for remote collaboration. In *Computer Supported Co-operative Work*, M. Beaudouin-Lafon, Ed., vol. 7 of Trends in Software. John Wiley & Sons, 83-102, 1999.

[IT94] E.A. Isaacs and J.C. Tang, What Video Can and Cannot Do for Collaboration: A Case Study, *Multimedia Systems*, Vol. 2, 63-73, 1994.

[JL01] A. Johnson and J. Leigh, Tele-Immersive Collaboration in the CAVE Research Network. In *Collaborative Virtual Environments: Digital Places and Spaces for Interaction*, 225-243, Springer Verlag, 2001

[JMF] Java Media Framework Home Page, Sun Microsystems. In http://java.sun.com/products/java-media/jmf/

[JE98] G. Johnson, G. and Elvins, T. T. 1998. Introduction to collaborative visualization. *SIGGRAPH Comput. Graph*. 32, 2, 8-11, 1998.

[KBAW94] R. Kazman, L. Bass, G. Abowd and M.Webb. SAAM: A method for analyzing the properties of software architectures. In Proceeding of International Conference on Software Engineering, ICSE'94, 81–90, May 1994.

[Lei05] S. Leikness, I. Osvoll. Success Factors in Troll Geosteering. In Offshore Technology Conference, 2005.

[LJDB99] J. Leigh, A. Johnson, T. DeFanti, M. Brown. A Review of Tele-Immersive Applications in the CAVE Research Network. In *Proceedings IEEE VR99*, 180-187, 1999.

[LL02] D. Li and R. Li. Transparent sharing and interoperation of heterogeneous single-user applications. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work* (CSCW '02). ACM Press, 246-255, 2002

[LM98] D. Li and R. Muntz. Coca: Collaborative objects coordination architecture. In *ACM Conference on Computer Supported Cooperative Work*, CSCW'98, 179-188, 1998.

[LM99] Li, D. and Muntz, R. R. 1999. A collaboration specification language. In Proceedings of the 2nd Conference on Domain-Specific Languages PLAN '99. ACM Press, 149-162, 1999

[LN02] Y. Laurillau and L. Nigay. Clover architecture for groupware. In *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work (CSCW '02)*. ACM Press, 236-245, 2002

[Lua] Lua.org In:http:// www.lua.org

[LWM 99b] D. Li, Z. Wang, and R.R. Muntz. Got COCA? A New Perspective in Building Electronic Meeting Systems. In *Proceedings of WACC'99 Conference on Work Actwzties Coordinatzon and Collaboration*, 1999.

[LYS+01] J. Leigh, O. Yu, D. Schonfeld, R. Ansari, E. He and A. Nayak, J. Ge and N. Krishnaprasad, K. Park and Y. Cho, L. Hu, R. Fang and A. Verlo, L. Winkler, and T. A. DeFanti, Adaptive Networking for Tele-Immersion. In *Proceedings of the Immersive Projection Technologies/Eurographics Virtual Environments (IPT/EGVE)*, 199-208, 2001.

[Mac99] W.E. Mackay. Media Spaces: environments for informal multimedia interaction. In *Computer Supported Co-operative Work*, M. Beaudouin-Lafon, Ed., vol. 7 of Trends in Software. John Wiley & Sons, 55-82, 1999.

[Mal02] J.L. Mallet. *Geomodeling*. Oxford University Press, 2002.

[MC90] T. W. Malone,and K. Crowston. What is Coordination Theory and How Can It Help Design Cooperative Work Systems? In Proceedings of the Conference on Computer Supported Cooperative Work (CSCW'90), ACM Press, 357-370, October 1990.

[MC94] T.W. Malone and K. Crowston. *The interdisciplinary study of coordination,* ACM Comput Surveys 26, 87–119, 1994

[MD96] J. Munson and P. Dewan. A concurrency control framework for collaborative systems. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, 278-287, November 1996.

[MF98] B. MacIntyre and S. Feiner. A distributed 3d graphics library. In *SIGGRAPH '98*, 361–370, 1998.

[MMB01] P. Molli, H. S. Molli, C. Bouthier. State Treemap: An Awareness Widget for Multi-Synchronous Groupware. In *Proceedings of the Seventh international Workshop on Groupware (*CRIWG). IEEE Computer Society, 106-114, 2001

[MNM] Microsoft Windows NetMeeting, In http://www.microsoft.com/windows/netmeeting/

[Niel94] J. Nielsen. *Usability Engineering*. Morgan Kaufmann Publishers, 1994.

[NW98] T.J. Nam, and D.K. Wright. CollIDE: A Shared 3D Workspace for CAD. In *Conference on Network Entities*, UK, 1998.

[OMG] Security Service Specification, Object Management Group. In http://www.omg.org/technology/documents/formal/omg_security.htm#Security_Service

[ONE] OneSpace CoCreate, In http://www.cocreate.com/products.cfm/ProdFamilyID/1/ProductID/46

[OO91] G. M. Olson and J. S. Olson. User-centered design of collaboration technology. Journal of Organizational Computing, 1, 61-83, 1991

[OO95] J. Olson and G. M. Olson, What mix of video and audio is useful for small groups doing remote real-time design work. In *Proceedings of SIGCHI'95*, ACM Press, 362-368, 1995.

[OOTF] Office of the Future project: http://www.cs.unc.edu/Research/stc/

[OS] Open Spirit, Open Spirit Coprporation. In http://www.openspirit.com

[Pat94] J. F Patterson. A Taxonomy of Architectures for Synchronous Groupware Applications. *In Proceedings of the CSCW'94 Workshop on Software Architetures for Cooperative Systems*, 1994

[PB03] R.O. Prates, S.D.J Barbosa. Avaliação de Interfaces de Usuário – Conceitos e Métodos. In *Anais do XXIII Congresso Nacional da Sociedade Brasileira de Computação* (SBC), 2003

[PG02] D. Pinelle, C. Gutwin. Groupware Walkthrough: Adding Context to Groupware Usability Evaluation. In In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing Our World, Changing Ourselves (CHI'02)*, ACM Press, 455-462, 2002.

[PHRM90] J. F. Patterson, R. D. Hill, S. L. Rohrdl, and W. S. Meeks. Rendezvous: An Architecture for Synchronous Multi-user Applications. In *Proceedings of the Conference on Computer-Supported Cooperative (CSCW 90)*, ACM, 317-328, 1990

[PIE] Pie menu central. In http://www.piemenu.com

[PKS+00] K.S.Park, Y. Cho, N. Krishnaprasad, C. Scharver, M. Lewis, J. Leigh, A. Johnson. CAVERNsoft G2: A Toolkit for High Performance Tele-Immersive Collaboration. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology 2000*, pp. 8-15, 2000.

[PKL00] K. S. Park, A. Kapoor, and J. Leigh. Lessons learned from employing multiple perspectives in a collaborative virtual environment for visualizing scientific data. In *Proceedings of the Third international Conference on Collaborative Virtual Environments (CVE '00)*, E. Churchill and M. Reddy, Eds.. ACM Press, 73-82, 2000.

[PR00] A. Puder and K. Romer, *Mico: an open source CORBA implementation*. Morgan Kaufmann, 2000

[Pra99] A. Prakash. Group editors. In *Computer Supported Co-operative Work*, M. Beaudouin-Lafon, Ed., vol. 7 of Trends in Software. John Wiley & Sons, 103-133, 1999.

[PS94] A. Prakash and H.S. Shim. DistView: support for building efficient collaborative applications using replicated objects. In *Proceedings of ACM Conference on Computer Supported Cooperative Work 1994*, 153–164, 1994.

[PW97] A. Pang and C. Wittenbrink. Collaborative 3D visualization with CSpray. *IEEE Computer Graphics and Applications*, 17(2), 32-41 , 1997.

[RF02] A.B. Raposo, H. Fuks. Defining Task Interdependencies and Coordination Mechanisms for Collaborative Systems. In A. M. Pinna-Dery, K. Schmidt and P. Zaraté (eds.), *Cooperative Systems Design: A Challenge of the*

Mobility Age (Frontiers in Artificial Intelligence and Applications, Vol. 74),88 – 103, IOS Press, Amsterdam, 2002.

[RG96a] M. Roseman and S.Greenberg. Building real time groupware with GroupKit, a groupware toolkit. *ACM Transactions on Computer–Human Interaction*, 3(1):66–106, March 1996.

[RG96b] M. Roseman and S.Greenberg. TeamRooms: Network places for collaboration. In *Proceedings of the ACM CSCW'96 Conference on Computer Supported Cooperative Work*, 16–20, November 1996.

[RGE] Reachin GeoEditor, In http://www.reachin.se/products/reachingeoeditor/

[RSVW94] W. Reinhard, J. Schweitzer, G. Völksen, and M. Weber. CSCW Tools: Concepts and Architectures. *Computer* 27, 5, 28-36, May 1994

[RS93] J. Rekers and I. Sprinkhuizen. A LOTOS Specification of a CSCW Tool. In *Proceedings of Design of Computer Supported Cooperative Work and Groupware Systems*, 1993

[RWC+98] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs. 1998. The office of the future: a unified approach to image-based modeling and spatially immersive displays. In *Proceedings of the 25th Annual Conference on Computer Graphics and interactive Techniques* SIGGRAPH '98, ACM Press, 179-188, 1998

[Sal95] D. Salber. *De l'interaction homme-machine individuelle aux systèmes multi-utilisateurs.* Thése de Doctorat, Université Joseph Fourier – Grenoble 1, 1995.

[SB92] K. Schmidt and L. J. Bannon: Taking CSCW Seriously – Supporting Articulation Work. *Computer Supported Cooperative Work (CSCW) – An International Journal*, vol. 1, nos. 1-2, 7-40, 1992.

[SBD99] J. Stewart, B. Bederson, and A.Druin. Single Display Groupware: A model for co-present collaboration. In *Human Factors in Computing Systems: CHI 99*, ACM Press, 286-293, 1999

[SBM+97] R. Strom, G. Banavar, K. Miller, A. Prakash, and M. Ward. Concurrency control and view notification algorithms for collaborative replicated objects. In *Proceedings of the 17th International Conference on Distributed Computing Systems*, IEEE Computer Society Press, 194–204, 1997.

[SBMK01] D. H. Sonnenwald, R. E. Bergquist, K. L. Maglaughlin, E. Kupstas-Soo and M. C. Whitton. Designing to Support Collaborative Scientific Research Across Distances: The nanoManipulator Environment. In *Collaborative Virtual Environments: Digital Places and Spaces for Interaction,* E. E. Churchill, D. N. Snowdon and A. J. Munro (Eds), Springer Verlag, 202-224, 2001

[SC02] Sun, C. and Chen, D.: Consistency Maintenance in Real-Time Collaborative Graphics Editing Systems. *ACM Transactions on Computer-Human Interaction* ,Vol. 9, Issue 1, 1-41, 2002

[Sch02] K. Schmidt, Remarks on the complexity of cooperative work. In Pascal Salembier and Tahar Hakim Benchekroun (eds.): Cooperation and Complexity in Sociotechnical Systems, [special issue of] Revue des sciences et technologies de l'information (RSTI), série RAI, vol. 16, no. 4-5, Hermes, Lavoisier, 443-483, 2002

[SKSH96] C. Schuckmann, L. Kirchner, J. Schummer, and J.M. Haake. Designing object-oriented synchronous groupware with COAST. In *Proceedings of the ACM Conference on Computer-Supported CooperativeWork*, 30–38, 1996.

[SE98] C. Sun, and C.A. Ellis. Operational transformation in real-time group editors: Issues, algorithms, and achievements. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, ACM, 59–68, 1998

[SERG] Semiotic Engineering Research Group, In http://www.serg.inf.puc-rio.br/

[SG96] M. Shaw and D. Garlan, S*oftware Architecture: Perspectives on an Emerging Discipline*. Prentice Hall, 1996.

[SGI] SGI OpenGL Vizserver, In http://www.sgi.com/software/vizserver

[Sha94] D. Shapiro, The limits of ethnography: combining social sciences for CSCW. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (CSCW '94). ACM Press, 417-428, 1994

[Ste02] A. Steed. Aspects of Usability in Collaborative Virtual Environments. IEEE Virtual Reality 2002 Course Notes: Usability Evaluation Techniques for Virtual Reality Technologies: Human Factors Issues, 2002

[Sum98] Robert Summers. *Official Microsoft NetMeeting Book*. Microsoft Press, 1998.

[SJZ+98] C. Sun, X .Jia, Y. Zhang, Y. Yang, and D.Chen. Achieving convergence, causality preservation, and intention preservation in real-time cooperative editing systems. *ACM Trans. Comput.-Hum. Interact.* 5, 1, 63-108 , 1998.

[SWC76] J. Short, E.Williams, and B. Christie. *The Social Psychology of Telecommunications*, Wiley and Sons, 1976.

[SZ99] S. Singhal and M. Zyda, *Networked Virtual Environments: Design and Implementation,* Addison-Wesley, 1999

[TEL] Tele-immersion at the Electronic Visualization Laboratory (EVL) .In http://www.evl.uic.edu/cavern/

[TER] Teraburst.V2D In http://www.ipvideosys.com/products_v2d.html

[Tie01] D. Tietze, A Framework for Developing Component-based Cooperative Applications, Ph.D. Dissertation, Technischen Universitat Darmstadt, 2001

[TH02] K. Tushingham and P. Hodgson. Collaborative interpretation environments re-energize the workplace: so what's the next phase ? First Break , EAGE, Vol. 20.3, 172-176, 2002.

[TR03] F.E.H. Tay, A. Roy. CyberCAD: A Collaborative Approach in 3D-CAD Technology in a Multimedia-Supported Environment. *Computers in Industry*, Vol. 52, Number 2, 127-145, 2003

[TSW03] J. G. Tromp, A. Steed , and J. R.Wilson. Systematic usability evaluation and design issues for collaborative virtual environments. In *Presence: Teleoper. Virtual Environ.* 12, 3 (Jun. 2003), 241-267, 2003

[VIC] VIC Videoconferencing Tool. In: http://www-mice.cs.ucl.ac.uk/multimedia/software/vic/index.html

[WBCP99] C. Winkler, F. Bosquet, X. Cavin, J. C. Paul: Design and Implementation of an Immersive Geoscience Toolkit. *IEEE Visualization*, 429-432, 1999.

[ZCV00] Y. Zhuang, L. Chen and R. Venter. CyberEye: An Internet-Enabled Environment to Support Collaborative Design. *Concurrent Engineering: Research and Applications*, 8~3, 213–229, 2000.

Appendix 1: NetGocad's user interface

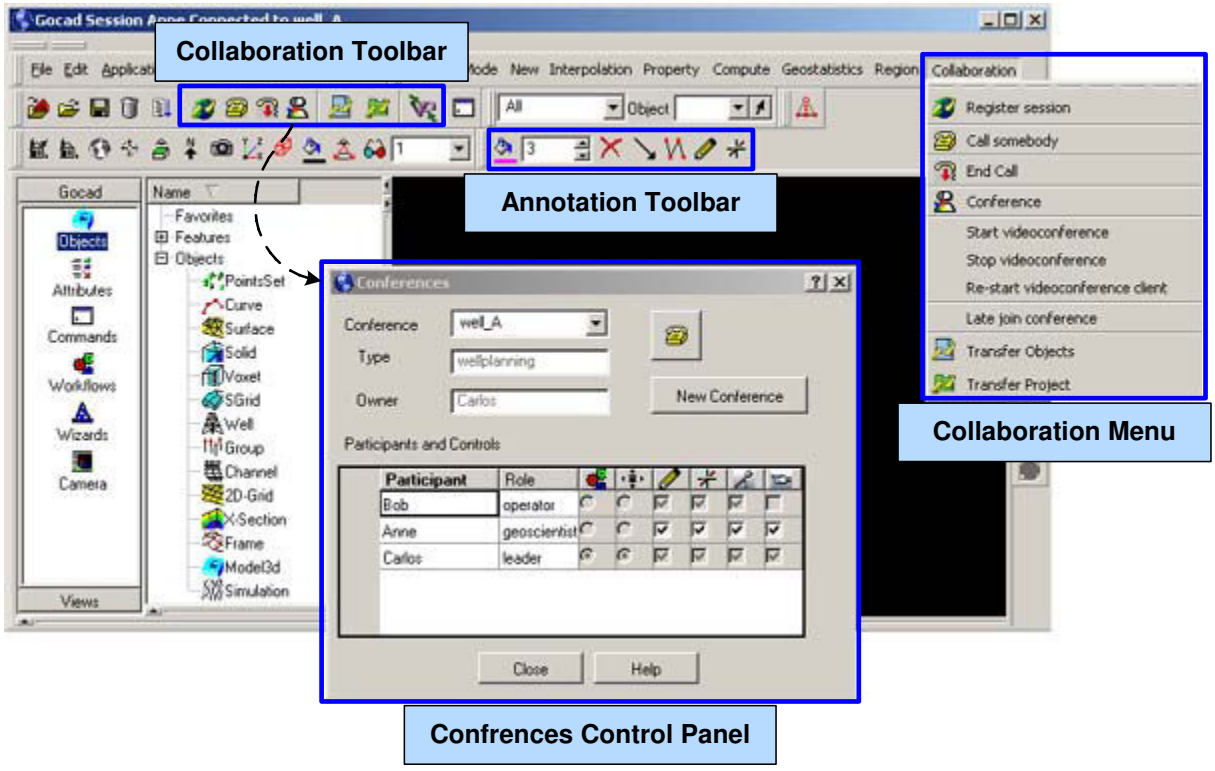


FIG. A3.1 – NetGocad's interface.

Collaboration Toolbar

 register the participant for collaboration

 call a conference (through the conference owner)

 disconnect the participant from the conference

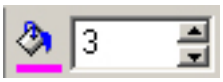
 open the conference control panel

 open a dialog for the broadcast of individual objects

 transfer the current project

 open a camera synchronized with a VR participant

Annotation Toolbar



change color and width of annotations

 clear annotations

 arrow annotation

 polyline annotation

 free-hand annotation

 telepointer (styles can be chosen among arrow, cross and text label)

Appendix 2: NetGocad IDL

In this Appendix we present a condensed version of the interface of the main NetGocad classes (CORBA IDL) described in section 3.3 (auxiliary classes and methods are not shown).

```
module GMeeting {      // defines a namespace

    // Forward declarations
    //====================
    interface Manager;
    interface Partner;
    interface Conference;
    interface Participant;


    // type definitions
    //====================
    typedef sequence<Participant> ParticipantSeq;
    typedef sequence<Conference> ConferenceSeq;
    typedef sequence<string> NameSeq;
    typedef sequence<boolean> SeqBoolean;
    typedef sequence<short> SeqInt;

   // Collaboration channels
   enum ChannelType {command_channel, camera_channel,
                     annotation_channel,cursor_channel,
                     avatar_channel, audio_channel,
                     video_channel, project_channel,
                     N_CHANNELS};

//============
// Manager of the overall conferencing system.

interface Manager {

  readonly attribute string name;
  void register_manager();
  boolean connect(in string caller_name, in string role,
                  in string partner_name);
  void disconnect(in string caller_name);
  void signal_participants(in string signal_type);

  //-- Participant
  boolean register_participant(in Participant participant);
  void unregister_participant(in string name);
  Participant find_participant (in string participant_name);
  NameSeq get_participant_names();
```

```
   //-- Conference
      boolean create_conference(in string name, in string owner_name,
                                in string conf_type);
      void destroy_conference(in Conference conference);
      Conference find_conference (in string conference_name);
      NameSeq get_conference_names();

      //====  Access to LuaConference  ======
      NameSeq get_collaboration_types();
      NameSeq get_collaboration_roles(in string collab_type);

   };

   //========
   // Partner: superclass for Participant and Conference

   interface Partner {
      attribute string name;
      oneway void run_command( in string cmd, in string client_name);
      oneway void set_camera_parameters( in CameraControl param,
                                         in string client_name);
      oneway void set_cursor ( in NetCameraCursor cursor,
                               in string client_name );
      oneway void set_avatar( in CameraControl param,
                               in string client_name);
      void broadcast_project(in NetProject project, in string
                             client_name);

   };

   //============
   // Conference

   interface Conference : Partner {
      void  include_participant(in  Participant  participant,  in  string
role);
      void exclude_participant(in Participant participant);
      NameSeq get_participant_names();
      Participant find_participant (in string participant_name);
      void destroy();   // destroys conference
      boolean accept_client_connect( in string remote_client_name,
                                     in string client_role);
      readonly attribute string owner;

      //==== Control policy (logic implemented in LUA)
      readonly attribute string type;
      string role(in string participant_name);
      string channel_controller(in short channel);
      void set_channel_controller (in short channel,
                                   in string controller_name,
                                   in string requester_name);
      void set_channel_out (in short channel, in string
                            participant_name,
                           in boolean value);
      boolean channel_out (in short channel, in string
                           participant_name);
      void set_channel_in (in short channel, in string participant_name,
```

```
                          in boolean value);
      boolean channel_in (in short channel, in string participant_name);
      SeqInt defined_channels();
      string get_commands(in string sesssion_name, in string cmd_type);

      // Get control matrixes for each channel
      //======================================
      SeqBoolean get_key_snd (in short channel);
      SeqBoolean get_key_rcv (in short channel);
      SeqBoolean get_int_snd (in short channel);
      SeqBoolean get_int_rcv (in short channel);

      // Late join
      //==========
      void late_join(in string sesssion_name);

      // Videoconference
      //===============
      void start_videoconference(in string requester);
      void stop_videoconference(in string requester);
      boolean videoconference_on();
      void start_videoconference_clients();
      void restart_videoconference_client(in string name);
      void  new_videoconference_participant(in  string  name,  in  short
vc_id,
                                            in boolean audio,
                                            in boolean video);
      void deleted_videoconference_participant(in short vc_id);
      void set_videoconference_matrixes();

   //=========
   // Participant

   interface Participant : Partner {

     // "Client methods"
     //=================
      void client_connect( in Partner partner, in string show_name);
      boolean client_accept_connect( in string caller_name );
      boolean client_accept_conference_connect( in string caller_name,
                                                in string caller_role,
                                                in string conf_name);
      void client_disconnect();
      boolean client_is_connected();
      Partner client_partner();
      void client_signal(in string signal_type);
      string host_id();
      void set_channel_out(in short channel, in boolean value);
      boolean channel_out(in short channel);
      void set_channel_in(in short channel, in boolean value);
      boolean channel_in(in short channel);
      string get_commands(in string cmd_type);

      // Videoconference
      //================
       boolean start_videoconference(in string name, in string host_id);
       boolean stop_videoconference();
       boolean start_videoconference_server(in string name,
```

```
                                          in string conf_type);
    boolean stop_videoconference_participant(in string coord);
    void set_videoconference_application_control(in boolean value);
    void set_videoconference_key_matrixes(in SeqBoolean snd_audio,
                                          in SeqBoolean snd_video,
                                          in SeqBoolean rcv_audio,
                                          in SeqBoolean rcv_video);
    void set_videoconference_intention_matrixes(
                                          in SeqBoolean snd_audio,
                                          in SeqBoolean snd_video,
                                          in SeqBoolean rcv_audio,
                                          in SeqBoolean rcv_video);

};
```

Appendix 3: Lua and related plugins

For the sake of completeness, we provide here a brief overview of the Lua language, based on [CCI99], and of the plugins used to interface Gocad and gWLog (section 3.6) using Lua [Cam02]. The descriptions are mainly adapted from [CCI99] and [Cam02], with some comments. Extensive material on Lua, including downloads, documentation and community information can be obtained in [Lua].

The Lua Language

Lua is an interpreted programming language [FIC96], which was originally designed to serve as a single configuration language in projects of Tecgraf /PUC-Rio for Petrobras. It is freely available, and due to its simplicity, efficiency (it is faster the Perl), extensibility and portability (it is coded in standard C and the entire source fits in one diskette), it is currently being employed in many products and prototypes in academic institutions and companies, in diverse application domains [Lua]. Surprisingly for the authors, it has become one the preferred (if not *the* preferred) configuration language in the game community.

Lua integrates in its design data-description facilities, reflexive facilities, and familiar imperative constructs. On the "traditional" side, it is a procedural language with usual control structures (whiles, ifs, etc.), function definitions with parameters and local variables. On the less traditional side, Lua also provides functions such as first-order values, closures, and dynamically created associative arrays (called tables) as a single, unifying data-structuring mechanism.

The simple example of Lua code below shows the implementation of the `map` function in Lua, which receives a list `a = (a1, a2, ..., an)` (represented by a table with indexes 1,2,…,n) and a function `f`, and returns a new list `(f(a1), f(a2), …, f(an))`.

```
function map (a, f)
    local b = {}  -- creates a new table
    local i=1
    while a[i] do
        b[i] = f(a[i])
        i = i+1
    end
    return b
end
```

There is no notion of a "main" program in Lua; it only works embedded in a host client. Lua is provided as a library of C functions to be linked to host applications. The host can invoke functions in the library to execute a piece of code in Lua, write and read Lua variables, and register C functions to be called by Lua code. Moreover, fallbacks can be specified to be called whenever Lua does not know how to proceed. This way, Lua can be augmented to cope with rather different domains, thus creating customized programming languages sharing a single syntactical framework

The type table implements associative arrays, that is, arrays that can be indexed not only by integers but also by strings, reals, tables, and function values. Associative arrays are a powerful language construct; many algorithms are simplified to the point of triviality because the required data structures and algorithms for searching them are implicitly provided by the language. Most typical data containers, such as ordinary arrays, sets, bags, and symbol tables, can be directly implemented by tables. Tables can also implement records by simply using field names as indices. Lua supports this representation by providing `a.name` as syntactic sugar for `a["name"]`.

Tables are created with special expressions called *constructors*. The simplest constructor is the expression `{}`, which returns a new empty table. An expression like `{n1=exp1, n2=exp2, …}` creates a new table and stores in each field `ni` the result of `expi`. A typical example is the creation of a point:

```
point1 = {x=10, y=30}
```

Constructors can also be used to build lists, with the syntax `{exp1, exp2, …}`. This expression creates a new table and stores in each field `i` the result of `expi`. Therefore after the assignment:

```
days = {"Sun", "Mon", "Tue", Wed", "Thr", "Fri", "Sat"}
```

the expression `days[3]` will result in the string `"Tue"`.

In Lua, functions are first-class values, and can therefore be assigned to table fields. This feature allows the implementation of some interesting object-oriented facilities, which are made easier by some syntactic sugar. For instance, the expression:

```
receiver:foo(params)
```

is a syntactic sugar for

```
receiver.foo(receiver, params)
```

That is, the function stored in field `foo` from object `"receiver"` is called, with `receiver` as its first argument (which plays the role of `self`). A more complete treatment of Lua's object-oriented facilities may be found in [IFC96].

Lua is implemented as a small library of C functions, written in ANSI C, and compiles unmodified in all known platforms. The implementation goals are simplicity, efficiency, portability, and low embedding cost.

The Lua Plugin

The Lua plugin, written by João Luiz Campos at Tecgraf, provides a binding library to integrate Lua and Gocad. This library is responsible for translating the Gocad's object methods to Lua. This library is not required by NetGocad, only by gWLog, and is described here as an example of the powerful features that can be accomplished through the integration between Lua and Gocad, which could be further explored in the future (with this mechanism the LuaConference module could execute programs written in Lua using Gocad commands).

Writing a bind between Gocad and Lua is not an easy task, since it requires the exportation of all object method types. In order to facilitate this task, an application called tolua is used [Lua] which receives as input a C++ class definition and maps all object types and methods to Lua automatically. As a result we have a C++ file that implements the Gocad object using the Lua API. The implementation of the Lua binding also ensures portability by hiding from the programmer the Lua API changes. For instance, as the Lua API changed from version 3.x to 4.x, the new bind was generated using the new version of the tolua program, guaranteeing the compatibility with the plugin and the new Lua version used.

Besides the Lua bind, the Lua plugin is also composed of all the objects needed to integrate the Lua language to the Gocad interface: a Lua terminal (GUI library) that was added to the Gocad stack bar manager and objects that can load and save Lua programs from the Lua terminal (ASCII library).

The LuaOrb plugin

LuaOrb [CCI99] is a binding between Lua and CORBA. As usual, this binding defines mappings between Lua and IDL types. However, unlike other bindings, LuaOrb relies on the reflexive facilities of CORBA and the dynamic nature of Lua. The main goal of LuaOrb is to offer a more suitable support for the development of open applications that allow the dynamic incorporation of CORBA objects.

Using the CORBA features for introspection (Interface Repository) and dynamic construction of method calls (Dynamic Invocation Interface - DII), LuaOrb offers a binding where scripts can incorporate and make effective use of new objects' interfaces at runtime. Moreover, as in other bindings, Lua can use external CORBA servers transparently; in the same way it uses internal objects. Using the CORBA's Dynamic Skeleton Interface (DSI), LuaOrb also supports the dynamic implementation of new CORBA object servers. LuaOrb uses proxies for each external CORBA object to be handled by a program.

The key point in this language binding is its dynamic nature. Every step, from type checking to method identification and invocation, is done at runtime. Therefore, a program can incorporate new objects without any additional declarations and without recompilation. Because of this dynamic nature, some cohesion between data types is possible. Therefore, many changes in an IDL interface do not affect its uses in Lua, such as reordering and removing of structure fields, or changes between IDL types that have

similar representations in Lua (e.g. short and long, or array and sequence). This cohesion works recursively and allows, for instance, Lua tables to be automatically converted to an array of records.

To hide from the final application changes that can be made in LuaOrb bound with CORBA, as done with Lua, LuaOrb was incorporated within Gocad by means of a plugin. The plugin implements two objects: one to initialize the CORBA and LuaOrb interfaces, and one to implement a CORBA dispatcher that binds the CORBA and Qt event handlers. The new dispatcher object was needed because the one in the Mico-CORBA implementation [PR00] does not work with the Qt version used.

Résumé

UNE ARCHITECTURE POUR LA COLLABORATION SYNCHRONE À DISTANCE APPLIQUÉE À LA VISUALISATION ET À LA MODÉLISATION DANS LES GÉOSCIENCES

Les systèmes de modélisation collaborative sont encore rares dans la plupart des domaines techniques, surtout dans les géosciences, parce que leur développement implique beaucoup de questions techniques et organisationnelles et généralement demande une nouvelle conception majeure des applications mono-utilisateur déjà opérationnelles.
Dans cette thèse, nous proposons une architecture permettant la transformation d'une application existante dans un système coopératif, exclusivement au moyen de la composition dynamique de modules d'extension de temps d'exécution (run-time plugins). Celle-ci tient compte des demandes de modélisation et visualisation dans les géosciences et supporte l'intégration dans l'application de base de services de coopération, de communication multimédia et d'un schéma de coordination simple et extensible basé sur les rôles, fournissant des mécanismes intégrés de diffusion et de contrôle d'utilisation (floor control) pour les canaux de collaboration présentés.
Nous envisageons aussi un modèle de travail dans lequel des sessions de collaboration peuvent impliquer l'utilisation d'interfaces d'utilisateurs hétérogènes (ordinateur de bureau et réalité virtuelle) par les participants à distance et nous analysons l'application du système dans des cas de figure opérationnels.

Mots-clés : TCAO, collaboration à distance, collecticiel synchrone, architecture collaborative, module d'extension, contrôle d'utilisation, co-conception, géomodélisation, visualisation coopérative.

Abstract

AN ARCHITECTURE FOR SYNCHRONOUS REMOTE COLLABORATION APPLIED TO VISUALIZATION AND MODELING IN THE GEOSCIENCES

Collaborative modeling systems are still rare in most technical domains, particularly in the geosciences, because their development involves many technical and organizational issues and usually requires major redesign of operational single-user applications.
In this thesis we propose an architecture that enables the transformation of an existing application into a collaborative system, exclusively through the dynamic composition of run-time plugins. It takes into account the requirements of modeling and visualization in the geosciences, and supports the integration into the base application of cooperation services, multimedia communication, and a simple and extensible role-based coordination scheme that provides integrated broadcasting and floor-control mechanisms for the introduced collaboration channels.
We also consider a work model in which collaboration sessions may involve the use of heterogeneous user interfaces by the remote participants (desktop and virtual reality), and discuss the application of the system in operational scenarios.

Keywords: CSCW, remote collaboration, synchronous groupware, collaborative architecture, plugin, floor control, collaborative modeling, geomodeling, collaborative visualization.

LORIA
Laboratoire Informatique et Analyse des Données
Rue du Doyen Marcel Roubault – 54500 VANDOEUVRE – Tél. 03 83 59 64 35