Visualisation scientifique collaborative

Thèse

présentée à la faculté des Sciences de l'Université de Neuchâtel
pour obtenir le grade de docteur ès science par

Steve Casera

steve.casera@unine.ch

La soutenance a été réussie le 26 juin 2007

Composition du jury :

Professeur Peter Kropf
Directeur de thèse
Université de Neuchâtel

Professeur Hans-Heinrich Nägeli
Co-directeur de thèse
Université de Neuchâtel

Professeur Hanspeter Bieri
Université de Berne

Professeure Franziska Tschan
Université de Neuchâtel

Université de Neuchâtel

FACULTE DES SCIENCES
Secrétariat-Décanat de la faculté
- Rue Emile-Argand 11
- CP 158
- CH-2009 Neuchâtel

IMPRIMATUR POUR LA THESE

Visualisation scientifique collaborative

Steve CASERA

UNIVERSITE DE NEUCHATEL

FACULTE DES SCIENCES

La Faculté des sciences de l'Université de Neuchâtel,
sur le rapport des membres du jury

Mme F. Tschan Semmer,
MM. P. Kropf (directeur de thèse),
H.-H. Naegeli
et H. Bieri (Berne)

autorise l'impression de la présente thèse.

Neuchâtel, le 23 août 2007

Le doyen :
T. Ward

UNIVERSITE DE NEUCHATEL
FACULTE DES SCIENCES
Secrétariat-décanat de la faculté
Rue Emile-Argand 11 - CP 158
CH-2009 Neuchâtel

Résumé

Mots clés : collaboration, visualisation scientifique, travail collaboratif, IHM, *awareness*, ergonomie

Keywords : collaboration, scientific visualization, Computer Supported Cooperative Work, Human Computer Interaction, awareness, usability

La visualisation scientifique est utilisée habituellement pour résoudre des problèmes complexes et analyser des données. La collaboration joue un rôle crucial dans le domaine de la visualisation scientifique. En effet, dans les domaines d'application concernés, il est fréquent que l'on doive recourir aux compétences d'un spécialiste situé à une certaine distance.

Dans cette thèse, nous présentons les différentes notions jouant un rôle essentiel dans le domaine de la collaboration, puis nous montrons que, parmi les systèmes de visualisation scientifique collaborative actuellement disponibles, certains sont mieux adaptés à la collaboration alors que d'autres cherchent avant tout une grande efficacité. Nous présentons un système qui satisfait ces deux exigences. D'une part, il s'agit de mettre à disposition des outils efficaces permettant la collaboration, afin que tous les participants puissent intervenir et communiquer entre eux. D'autre part, l'utilisation de ces outils doit être intuitive et permettre une avance rapide dans le travail.

Lorsque plusieurs participants collaborent à distance, il faut assurer que le temps d'attente pour l'affichage des objets graphiques soit le plus petit possible. Ceci doit être assuré pour tous les participants indépendamment de leur séparation physique. Afin de réduire ce temps d'attente, nous avons analysé les différentes possibilités de transmettre les données de visualisation entre les ordinateurs. Il en résulte plusieurs configurations possibles. Nous avons développé une heuristique qui, pour chaque action demandée par les participants, choisit une configuration adéquate.

Concernant la facilité d'utilisation, nous avons développé des outils pour faciliter la collaboration. Pour vérifier la facilité d'utilisation de notre système, nous avons conduit une expérimentation lors de laquelle des participants ont utilisé notre système pour accomplir une tâche assez simple. Nous avons comparé les résultats obtenus en fonction des moyens mis à disposition. Nous n'avons pas pu mettre en évidence un effet significatif imputable à la disponibilité de nos outils. Cependant, il en est ressorti que l'audio, comparé au *chat* seul, permet d'améliorer les performances et de faciliter la communication.

Remerciements

Un bon nombre de personnes ont contribué de près ou de loin à la réalisation de cette thèse. J'aimerais les remercier toutes même si je ne pourrais ici en donner la liste exhaustive. Je tiens ainsi à remercier :

- Mon jury de thèse, pour leurs corrections et leurs conseils.
- Mes parents, pour leur soutien.
- Mes amis de l'institut d'informatique et de l'institut de psychologie du travail et des organisations (IPTO).
- Toutes les personnes ayant aimablement accepté de prendre part à l'expérimentation de ZoomIn.
- Hervé Sanglard pour avoir réaliser le projet ZoomIn initial.
- Les personnes ayant participé au développement de ZoomIn dans le cadre de travaux de semestre ou de diplôme, Alain Conod, Daniel Varrin, Jacques Oberli, Christophe Chasles, Christophe Jeannotat, Abdelhak Said et Luca Petraglio.

Table des matières

Table des figures

Table des tableaux

Chapitre 1 Introduction

La **visualisation** est un outil essentiel dans l'étude de phénomènes complexes, qui incluent des domaines aussi divers que la physiologie humaine, l'évolution du climat au cours des siècles, les marchés financiers internationaux ou bien la conception d'avions. La visualisation permet de représenter des données d'une manière pertinente.

Les résultats fournis par la visualisation aident les spécialistes à analyser leurs données et à étudier des phénomènes, tout en développant les connaissances de chacun.

La visualisation est présente dans la vie de tous les jours dans de nombreux secteurs de notre société, comme par exemple la recherche, l'éducation, la sécurité, la santé publique ou la vie de famille. Les cartes ou les animations météorologiques présentées quotidiennement à la télévision sont des exemples de visualisation. Les visualisations sont de plus en plus partagées grâce à leur publication sur Internet.

1.1 Contexte

La visualisation est un terme générique dont la signification varie en fonction de la situation et du domaine d'application. Par exemple, la distinction entre **visualisation scientifique (VS)** et **visualisation d'information** est peu claire [83], cette dernière se concentrant sur la visualisation d'information synthétique, par exemple des statistiques produites par un utilisateur, au contraire de la visualisation scientifique qui se concentre sur des informations produites par des simulations numériques ou des mesures. Néanmoins, il est difficile d'établir une frontière stricte entre ces deux types de visualisation.

A ses débuts, le travail du scientifique consistait à observer directement un phénomène et d'en tirer des conclusions, par exemple mesurer le temps de chute d'un objet pour comprendre les lois de la gravité. De nos jours, le scientifique enregistre (automatiquement ou non) des informations à des intervalles de temps pouvant être courts avec des instruments spécialisés. Ensuite, ces données brutes doivent être interprétées par des techniques efficaces. La visualisation aide les utilisateurs à explorer ou à comprendre des données par une représentation visuelle, cela nettement plus rapidement et aisément qu'en lisant des pages de chiffres ou de textes.

Contrairement à de nombreuses disciplines de l'informatique, où l'ordinateur sert à remplacer l'être humain par une tâche automatisée, la visualisation sert à assister l'être humain dans son travail en étendant ses capacités. L'utilisateur est actif, procède à de nombreuses interactions ainsi qu'à l'exploration visuelle, tout cela dans un processus complexe impliquant ses connaissances ainsi que son expérience.

D'un point de vue historique, la VS a vu le jour il y a une vingtaine d'année. Il y a encore quelques années, l'un des problèmes fondamentaux se trouvait au niveau matériel : en effet, pour faire fonctionner correctement un système de VS, il fallait une carte graphique spécialisée qui pouvait coûter plusieurs dizaines de milliers de francs. Aujourd'hui, en particulier grâce au progrès dans le domaine des jeux et des divertissements multimédia, de telles performances sont atteignables avec un matériel coûtant moins de cent francs. De même, du point de vue de la puissance de calcul, des ordinateurs de bureau coûtant quelques milliers de francs possèdent une puissance de calcul supérieure aux supercalculateurs d'il y a

quelques années. Cela permet aujourd'hui à la plupart des personnes intéressées d'utiliser un système de VS sur leur propre ordinateur.

La VS sert à explorer les données produites par des simulations numériques ou des mesures et à y identifier des phénomènes intéressants. Ce besoin d'**exploration** est une de ses principales caractéristiques. La VS demande souvent beaucoup de ressources de calcul et implique un volume important de données. Les simulations peuvent générer énormément de données et doivent ainsi être analysées par un système de VS. Par exemple, le LHC (*Large Hadron Collider*) au CERN (Centre Européen pour la Recherche Nucléaire) qui est actuellement en développement, va produire environ 300Mb de données par seconde, ce qui correspond à 15 pétaoctets de données ou plus de 5 millions de DVD par année [17].

Pour qu'un système de visualisation scientifique soit employé par le plus grand nombre d'utilisateurs possible, il se doit d'être **ergonomique**. Cela implique qu'un tel système doit atteindre le résultat prévu de manière efficace et intuitive. Concevoir un système de visualisation ergonomique est une tâche complexe, en partie à cause de la difficulté à mesurer précisément le degré d'ergonomie d'un système.

En VS, la complexité d'analyse et de synthèse des problèmes implique de partager ses connaissances. De plus, il est rare que les problèmes complexes soient résolus par une seule personne car ce type de problèmes demande l'expertise de spécialistes de plusieurs domaines. Cela implique un besoin de **collaboration**.

Une évolution s'est produite au niveau des connexions réseaux. Il y a quelques années, le grand public se connectait avec une connexion à faible débit, maintenant il n'est plus rare de se connecter à un réseau à haut débit. De plus en plus de personnes peuvent donc se permettre d'utiliser un système de VS pour collaborer, malgré le grand volume d'informations à transférer.

1.2 But de la visualisation scientifique collaborative

Cette section présente les **motivations** conduisant à effectuer des recherches dans le cadre de la **visualisation scientifique collaborative (VSC)**. Elle examinera des cas concrets d'utilisation de la **VSC**, ce qui permet de démontrer l'**utilité** de nos travaux.

1.2.1 Pourquoi collaborer ?

Un des avantages de la **collaboration** en VS est de pouvoir profiter des connaissances des différents participants afin de pouvoir découvrir des phénomènes intéressants.

Comme dans beaucoup d'autres domaines, les personnes utilisant un système de visualisation éprouvent le besoin de collaborer. Ces personnes sont souvent des spécialistes d'un domaine particulier et sont dispersées géographiquement. Dans ce document, les personnes utilisant ensemble un système collaboratif sont appelés **participants**. Il est évident que la collaboration entraîne un surcroît de travail (*overhead*), par exemple pour apprendre à utiliser un système collaboratif.

Un système destiné à être utilisé par une seule personne à la fois sur un seul ordinateur est appelé **mono-utilisateur**. Du point de vue du nombre de participants, nous distinguons la visualisation **en solitaire** (un seul participant) et la **visualisation collaborative** (plusieurs

participants). Une visualisation collaborative peut utiliser un système mono-utilisateur à condition que les participants se trouvent au même endroit.

1.2.2 Exemples concrets d'utilisation

La VSC peut être utile dans tous les domaines qui ont besoin de la VS. Comme exemple concret d'utilisation, on peut citer la médecine avec en particulier le domaine de la télémédecine [103]. Nous allons considérer trois exemples d'utilisation de la VSC.

Simulation : numéricien et océanographe

Il est fréquent qu'une personne produise, par une simulation, une description des phénomènes étudiés sous la forme de données numériques et qu'une autre personne se charge de leur interprétation. Par commodité, ces deux personnes vont travailler sur le même système et vont collaborer, dans un premier temps pour s'assurer que les données produites représentent bien le phénomène simulé et ensuite pour les interpréter. Par exemple, un océanographe modélise les courants dans la mer par un jeu d'équations aux dérivées partielles, alors qu'un numéricien écrit un programme résolvant ces équations, ce qui permet de produire des données numériques décrivant les courants marins, les variations de température et de salinité. A un moment, il faut qu'ils travaillent ensemble grâce à un système de VSC pour valider le lien entre les données produites et les phénomènes observés.

Ingénierie : conception d'avions

Pour la réalisation d'un nouvel avion, des spécialistes de différents domaines comme des ingénieurs, des spécialistes des matériaux et des spécialistes des écoulements de fluides devront collaborer. Souvent, ces spécialistes travaillent sur des sites différents, parfois très éloignés les uns des autres. Pour collaborer efficacement, ces spécialistes doivent pouvoir travailler sur une même visualisation.

Apprentissage : enseignement et démonstration

L'apprentissage, comme dans le cadre de l'enseignement ou de démonstrations, est un domaine dans lequel l'utilisation de la VSC se révèle utile. Par exemple, cela peut être un océanographe qui désire présenter à des étudiants des tourbillons se produisant dans une mer ou un spécialiste voulant présenter ses résultats de recherche à ses collègues situés à d'autres endroits géographiques. Le fait d'avoir accès à une même visualisation permet à tous les participants de poser des questions et de pouvoir interagir avec le système.

1.3 Définitions de base

Tout d'abord, nous énonçons l'**objectif général** de nos travaux :

*Réaliser un système **ergonomique** qui permet à des **participants** situés en différents lieux géographiques de **collaborer** sur une même **visualisation scientifique**.*

Afin de mieux saisir nos objectifs spécifiques, les quatre termes clefs de cet énoncé vont être examinés en détail : ***collaborer*** (section 1.3.1), ***participants*** (section 1.3.2), ***visualisation scientifique*** (section 1.3.3) et ***ergonomique*** (section 1.4).

Par la suite, nous allons préciser nos objectifs spécifiques ainsi que les besoins en découlant, énumérer les défis qui se posent en fonction de ces besoins et présenter les fonctionnalités désirées du système.

1.3.1 Collaboration

Collaborer signifie travailler avec d'autres personnes à une œuvre commune. Roschelle [84] précise cette notion en disant que la collaboration est le processus qui consiste d'une part à résoudre un problème à plusieurs, d'autre part à construire et à maintenir une représentation commune du problème.

Cette définition est affinée grâce à la description de trois aspects liés à la collaboration : répartition temps-espace, niveau applicatif et distinction entre coopération-collaboration.

1.3.1.1 Répartition dans le temps et dans l'espace

Nous pouvons distinguer plusieurs types de collaborations en fonction de la répartition dans le temps et l'espace.

Répartition dans le temps

Nous distinguons deux cas :

- Les activités des participants s'effectuent toujours en même temps. Toute action d'un participant est immédiatement visible par les autres participants. Il s'agit alors de collaboration **synchrone**.

- Les activités des participants peuvent s'effectuer à des moments différents. On parle alors de collaboration **asynchrone**.

Répartition dans l'espace

Nous distinguons deux cas :

- les participants se trouvent en un **même lieu** ;

- les participants sont situés en des **lieux différents**, qu'ils soient situés dans des pièces différentes d'un même bâtiment ou sur des continents différents.

Cas possibles en fonction de la répartition dans le temps et espace

Cette répartition conduit à une matrice temps-espace [28] comme indiqué dans le tableau 1.1 .

Tableau 1.1 : Matrice temps-espace

Espace / Temps	**Même lieu**	**Lieux différents**
Même moment	1	3
Moments différents	2	4

Il faut donc considérer quatre cas :

1) Même lieu / même moment : les participants se trouvent l'un à côté de l'autre ; c'est la situation où des participants travaillent devant le même écran.

2) Même lieu / moments différents : les participants se trouvent en un même endroit mais pas en même temps ; c'est ce qui se produit par exemple lorsqu'ils utilisent un unique calendrier pour noter leurs rendez-vous.

3) Lieux différents / même moment : les participants travaillent à distance et de manière synchrone, c'est la situation d'une vidéoconférence (avec un système tel que Netmeeting [64]) ou d'une conversation par un système de *chat*.

4) Lieux différents / moments différents : les participants travaillent quand ils le veulent à des endroits différents ; c'est la situation d'un échange de vues par emails ou d'un développement de logiciel avec un système tel que CVS (*Concurrent Versions System*) [32].

Le tableau 1.2 indique une classification de pratiques et de technologies traditionnelles adaptées à ces quatre cas.

Nos efforts se porteront sur le cas « lieux différents au même moment ». En effet, des spécialistes de différents domaines se trouvent souvent éloignés les uns des autres et l'utilisation d'un système collaboratif leur apporte la possibilité de partager leurs connaissances ainsi qu'un gain en temps et en argent par rapport à l'utilisation d'un système non-collaboratif.

Tableau 1.2 : Exemples de technologies traditionnelles utilisées dans ces quatre cas

Espace / Temps	**Même lieu**	**Lieux différents**
Même moment	Conversation de vive voix	Téléphone, vidéoconférence
Moments différents	Tableau d'affichage	Courrier postal, email

En VS, le cas de la collaboration au même moment est le cas habituel. En effet, un travail collaboratif dans le domaine de la VS nécessite de nombreux échanges rapides entre les participants. Une situation asynchrone requiert donc beaucoup trop de temps d'attente entre deux interactions et sera donc peu utile en pratique. Travailler de manière asynchrone n'est donc guère pratique en raison des longs temps d'attente qui en découlent. Malgré cela nous garderons à l'esprit cette situation : le mode asynchrone s'impose parfois lorsque les participants vivent dans des fuseaux horaires avec un grand décalage. Il s'agira donc de prévoir des mécanismes destinés à permettre de travailler de manière asynchrone. Un mécanisme élémentaire consiste, par exemple, à permettre à un participant de sauvegarder sa session de VS et ensuite à la transmettre aux autres participants.

1.3.1.2 Niveau applicatif

Dans la plupart des cas, les ordinateurs des différents participants ne sont pas identiques, pas plus que les connexions qui les relient (par exemple une connexion LAN, un réseau WIFI ou un réseau dédié à fibre optique) n'est homogène. Un système de VSC aspirant à une certaine généralité doit en tenir compte. Les participants doivent avoir accès aux mêmes informations, néanmoins il faut, dans la conception d'un système de VSC, tenir compte du fait que certaines

mesures de sécurité peuvent rendre cet accès impossible, donc il faut prévoir des modes de travail adaptés à de telles situations.

1.3.1.3 Collaboration et coopération

Les termes « coopération » et « collaboration » sont parfois utilisés avec des significations différentes ; il est utile d'exhiber la différence entre ces deux termes. Dillenbourg [22] définit les différences entre ces deux termes.

Dans le cadre d'une **coopération**, il y a une répartition nette du travail entre les participants. De façon concrète, chacun est responsable d'une sous-tâche. Par la suite, le résultat des sous-tâches des participants sont assemblés pour constituer le résultat final. Dans une coopération, chaque participant est responsable de sa propre production, même s'il doit, dans la plupart des cas, interagir avec les autres participants pour que le résultat final soit cohérent.

Dans le cadre d'une **collaboration**, il n'y a aucune répartition du travail explicite et préalable entre les participants. Ces derniers contribuent tous, selon leurs compétences, à chaque étape à l'élaboration du résultat final. Il n'est donc guère possible, une fois le travail terminé, d'identifier les contributions des différents participants.

En raison de l'intensité des échanges qu'implique la collaboration, pour Dillenbourg, une communication synchrone est exigée, tandis qu'une communication asynchrone est souvent suffisante pour la coopération. Dans notre contexte, comme les participants travaillent sans se repartir à l'avance le travail, le terme « collaboration » est donc employé. Par conséquent, une communication synchrone est utilisée.

1.3.2 Participants

1.3.2.1 Communication entre participants

Comme nous venons de le voir, la collaboration implique une intense communication entre les participants. Il s'agit donc d'examiner comment les participants **communiquent** dans des cas concrets ; cela permet d'en tirer des exigences dont nous devrons tenir compte dans nos travaux.

Nous considérons le cas d'une collaboration entre plusieurs participants. Un seul participant communique des informations aux autres, puis un autre participant prend le relais. En effet, si plusieurs participants émettent des informations en même temps, il est très difficile de pouvoir profiter des connaissances des autres. Dans la pratique, s'il y a un grand nombre de participants, un animateur est désigné pour gérer les actions des participants. Cela se produit typiquement dans le cadre de présentations. Des sous-groupes de participants peuvent aussi se former, collaborer et ensuite un porte-parole communique leurs résultats à tous les autres participants

Par conséquent, pour qu'un système soit utilisé par des participants situés en différents lieux géographiques, il doit permettre d'émettre et de recevoir de l'information, de définir des droits d'accès et de désigner un animateur chargé de gérer ces droits d'accès.

1.3.2.2 Public visé

Nous considérerons deux situations caractéristiques :

- Dans le cadre d'une **présentation**, seul une minorité des participants prennent la parole alors que les autres écoutent.

- Dans le cadre d'une **recherche de phénomènes**, la majorité des participants prennent la parole à un moment ou un autre. En pratique, le groupe ne comporte qu'un petit nombre de participants. L'expérience nous montre qu'un tel groupe comporte au plus cinq participants.

Notre public visé est composé de participants utilisant un système de VS et désireux de disposer d'une extension leur permettant de collaborer à distance ou de faire des démonstrations.

Nous ne considérerons pas les groupes de recherche composés d'un grand nombre de participants. En effet, de tels groupes sont rares en VSC. Par contre, il est réaliste d'avoir de grands groupes dans le cadre de présentations. C'est pourquoi, dans nos travaux, nous nous baserons sur un nombre de participants allant de 2 à 5 tout en ayant la possibilité d'avoir plus de participants, ce qui permet d'effectuer des présentations.

1.3.3 Visualisation scientifique

Par visualisation scientifique, nous entendons la visualisation de données scientifiques. Par exemple, il peut s'agir de la représentation du mouvement de l'air sur le fuselage d'un avion ou de celui des particules au cours d'une réaction chimique. Dans la visualisation scientifique, l'utilisateur cherche à produire des représentations graphiques exprimant certains aspects de données numériques afin de pouvoir les interpréter et de comprendre les phénomènes sous-jacents.

En VS, l'utilisateur procède à de nombreuses et diverses opérations, tout cela dans un processus complexe. La figure 1.1 montre une vue simplifiée de la transformation des données brutes en connaissance [51]. Les données peuvent aller du simple champ scalaire jusqu'à des données complexes exprimant par exemple les valeurs mesurées par le LHC (section 1.1). Le système de visualisation produit des images sur la base d'une spécification comprenant le matériel, les algorithmes et les paramètres nécessaire au fonctionnement des algorithmes. L'utilisateur interprète ces images puis explore les représentations en fonction de ses besoins et de ses connaissances. Malgré sa simplicité, cette figure présente les opérations et informations principales de la visualisation : l'interprétation, la connaissance, l'exploration, la spécification.

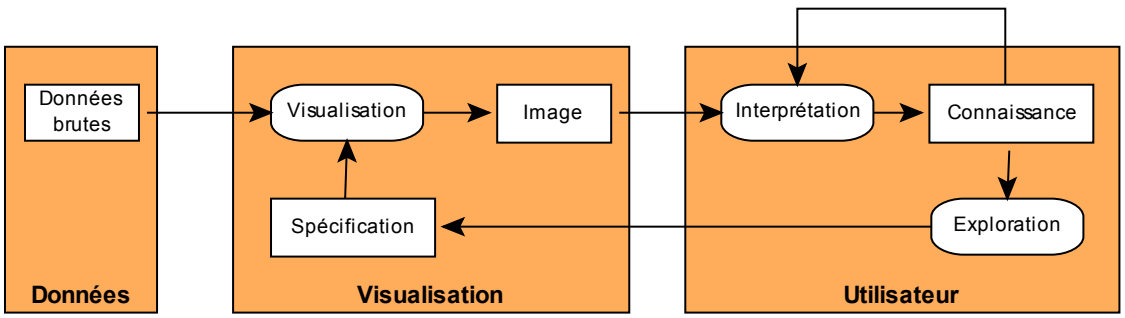

Figure 1.1 : Processus de découverte en visualisation

1.4 Ergonomie

L'**ergonomie** est une discipline qui vise l'adaptation d'un système à son utilisateur, permettant à ce dernier de travailler avec un maximum d'efficacité, de satisfaction et de bien-être, après un effort d'apprentissage faible. En informatique, l'ergonomie a pour objectif principal l'amélioration de l'interaction homme-ordinateur.

1.4.1 Utilisabilité et ergonomie

L'**utilisabilité** (*usability* ou facilité d'utilisation) est une notion proche de celle d'ergonomie. La norme ISO 9241 [48] définit l'utilisabilité comme « le degré selon lequel un système peut être utilisé par des utilisateurs identifiés pour atteindre des buts définis avec **effectivité** (*effectiveness*), **efficacité** et **satisfaction**, dans un contexte d'utilisation spécifié ». Un système est **effectif** s'il permet d'atteindre le résultat prévu. Il est **efficace** s'il permet de l'atteindre avec le moindre effort ou dans un temps minimal. La **satisfaction** finalement a trait au confort et à l'évaluation subjective de l'interaction par l'utilisateur.

1.4.2 Caractéristiques de l'utilisabilité

Pour Nielsen [71], l'utilisabilité est une composante de l'acceptabilité d'un système (figure 1.2). Il se base sur une hiérarchie où l'acceptabilité d'un système se divise en acceptabilité pratique et acceptabilité sociale.

L'acceptabilité sociale englobe les dimensions culturelles, sociales et éthiques. Des systèmes inacceptables socialement seraient, par exemple, des systèmes posant des questions indiscrètes aux utilisateurs ou visant à identifier dans un but de licenciement des opérateurs insuffisamment productifs.

L'acceptabilité pratique englobe à son tour un certain nombre de caractéristiques dont le **coût**, la **compatibilité** et **l'utilité pratique** (*usefulness*). A la suite de Grudin [36], Nielsen divise ensuite cette **utilité pratique** (c'est-à-dire le but que le système permet effectivement d'atteindre) en **utilité théorique** (c'est-à-dire le but que le système est censé permettre d'atteindre) et en **utilisabilité**. Cette hiérarchie met en évidence l'importance de l'utilisabilité pour l'acceptation (et donc le succès) d'un système tout en soulignant que ce n'est qu'une composante de cette acceptabilité.

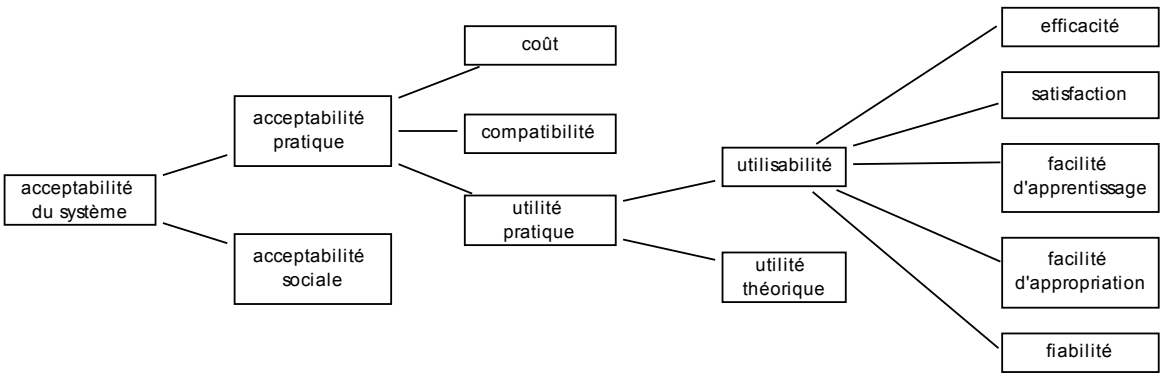

Figure 1.2 : Acceptabilité d'un système selon Nielsen

Un système peut satisfaire tous les critères d'utilisabilité mais être inutile (au sens de l'utilité théorique). L'analyse d'une activité permet de cerner quelles fonctionnalités sont utiles et quelles sont celles qui doivent être offertes par un système.

Nielsen décompose le concept d'utilisabilité en cinq caractéristiques majeures :

- l'efficacité (*efficient to use*),
- la satisfaction (*subjective satisfaction*),
- la facilité d'apprentissage (*easy to learn*),
- la facilité d'appropriation (*easy to remember*),
- la fiabilité *(few errors)*.

L'**efficacité** et la **satisfaction** se retrouvent dans la norme ISO 9241. La **facilité d'apprentissage**, **la facilité d'appropriation** et la **fiabilité** peuvent être considérées comme des composantes de **l'effectivité**. Selon Nielsen [71] et Mayhew [63], il est nécessaire, dans une démarche d'amélioration de l'utilisabilité, de déterminer l'importance de ces critères. Par exemple, la facilité d'apprentissage et l'efficacité peuvent être des objectifs contradictoires. L'expérience des utilisateurs influence l'importance de ces critères : par exemple des utilisateurs novices ou des experts n'auront pas la même perception de l'effectivité.

1.4.3 Utilisabilité des systèmes de VSC

Pour qu'un système soit adopté par les utilisateurs, il doit être **ergonomique**. En VSC, les données à visualiser peuvent être volumineuses ; en outre les calculs nécessaires pour la production des représentations sont souvent assez complexes. Un système de VSC doit donc aussi être efficace pour être accepté. Par exemple, si un participant doit attendre trop longtemps pour qu'un résultat s'affiche à l'écran, il risque fort de ne pas utiliser ce système. En particulier, un système de VSC ne doit pas conduire à des temps d'attente nettement plus grands qu'un système de visualisation non collaboratif comparable.

Pour un public d'utilisateurs novices et occasionnels, comme c'est souvent le cas pour des personnes suivant des présentations, il n'est pas possible de consacrer des heures pour maîtriser les fonctionnalités collaboratives du système. Bien évidemment, l'enseignant doit posséder une certaine maîtrise du système afin de pouvoir faire sa présentation. Le système doit être intuitif ; en particulier les interactions avec le système doivent être aussi naturelles que possible. Bien sûr, les personnes expertes profitent aussi des avantages d'un système intuitif.

Nous avons décidé de nous concentrer sur les deux caractéristiques suivantes qui sont jugées les plus importantes dans le cadre de nos travaux :

- **Efficacité** : il faut un temps minimal pour atteindre un résultat. Cette caractéristique est appelée efficacité technique.
- **Satisfaction** : un confort dans l'utilisation et dans les interactions doit être assuré par le système. Le système doit être simple et intuitif afin que les utilisateurs l'adoptent. Cette caractéristique est appelée satisfaction cognitive.

Bien évidemment, les trois autres caractéristiques de l'utilisabilité ne peuvent pas être ignorées ; nous en avons aussi tenu compte tout au long de nos travaux.

1.5 Esquisse des problèmes en VSC

Nous avons défini auparavant le terme « visualisation scientifique ». Comme des auteurs donnent différents sens au terme « visualisation scientifique collaboratif », nous précisons notre point de vue et donnons des exemples d'utilisation de la VSC pour identifier les principales difficultés.

1.5.1 Visualisation collaborative répartie

Dans le cadre de collaboration en VS à des lieux différents, il est utile de faire la distinction entre les trois termes suivants [9] :

1) **Visualisation scientifique répartie :** Ce terme désigne une coopération au niveau du système entre les processeurs. Une grande partie des systèmes actuels possède une telle fonctionnalité ; il est en effet possible d'effectuer des calculs sur différents ordinateurs. Bien évidemment, le potentiel de ce genre d'architecture ressort, par exemple, lorsqu'une tâche de visualisation requiert de longs calculs et que certains calculs sont effectués sur un ordinateur possédant une grande puissance de calcul alors les autres sont effectués sur les ordinateurs de moindre puissance. Même si plusieurs ordinateurs effectuent des calculs, un tel système est encore prévu pour une seule personne ; travailler dans un tel environnement n'implique donc pas un travail en groupe.

2) **Visualisation scientifique partagée :** Ce terme désigne une collaboration au niveau humain. La visualisation scientifique implique souvent que plusieurs participants travaillent ensemble pour créer ou interpréter des représentations graphiques. La plupart des systèmes actuels ne possèdent pas de telles fonctionnalités [9] et ce n'est que récemment que certains systèmes proposent des fonctionnalités collaboratives de base.

3) **Visualisation scientifique collaborative répartie :** Ce terme regroupe les deux termes, permettant ainsi une collaboration tant au niveau du système qu'au niveau humain. Un tel système permet à des utilisateurs, dispersés géographiquement, non seulement de partager leurs ressources, mais aussi d'interagir et de collaborer à travers un réseau pour produire une visualisation.

Par abus de langage, le terme « **visualisation scientifique collaborative** » est utilisé pour désigner cette dernière catégorie.

1.5.2 Exemples d'utilisation de la VSC

Nous allons considérer deux exemples particuliers d'utilisation de la VSC. Ces deux exemples démontrent les avantages de la collaboration dans le domaine de la VS. Mais ils permettent aussi d'identifier des besoins à satisfaire sans toutefois en constituer un inventaire complet.

1.5.2.1 Scénario : Télémédecine

Un médecin *A*, après avoir obtenu des images IRM du cerveau d'un patient, désire avoir l'avis d'un collègue *B* situé sur un autre site. Les données brutes permettant de représenter le cerveau sont disponibles sur l'ordinateur de *A*. *A* prend contact avec *B* pour obtenir son avis ; à cet effet, ils vont utiliser un système de VSC. Les deux médecins sont ainsi connectés et peuvent collaborer. *A* explique son problème et crée des objets graphiques (par exemple différentes coupes du cerveau) que *B* voit apparaître sur son écran. *A* indique des endroits intéressants. *B*, ayant plus d'expérience que *A*, repère un endroit étrange ; il crée à son tour un objet graphique représentant le phénomène (par exemple une coupe sous un angle différent). Les deux médecins communiquent, échangent des informations et parviennent ensemble à un diagnostic.

Les médecins doivent pouvoir communiquer, créer des objets graphiques partagés ainsi que pouvoir interagir avec ces derniers.

1.5.2.2 Scénario : Enseignement à distance

Un professeur en océanographie veut montrer à des étudiants situés dans une autre ville des phénomènes se produisant dans le Golfe du Lion. Pour cela, il lance un système de VSC auquel chaque étudiant se connecte. Le professeur construit des objets graphiques et explique ce qui se passe. Un étudiant pose une question telle que : « Que se passe-il à dix mètres sous la surface si le mistral souffle pendant une semaine ? ». Dans ce cas, soit le professeur crée lui-même un plan de coupe à l'aide des indications de l'étudiant (ce qui sera le cas si l'étudiant ne connaît pas bien le système), soit il donne à l'étudiant la possibilité de créer l'objet en question (si l'étudiant en est capable et que le professeur estime que l'intervention ne créera pas de dommage).

Il s'ensuit que le professeur doit pouvoir conserver le contrôle de la session de visualisation tout en ayant la possibilité d'attribuer des droits d'accès de manière très ciblée. Dans ce scénario, la communication est aussi un besoin essentiel.

1.5.3 Problèmes

De nos jours, la majorité des systèmes de visualisation scientifique sont des systèmes de VS mono-utilisateurs. Certains systèmes de VSC sont spécifiques à un domaine d'application particulier (par exemple la médecine) et donc peu adaptables à d'autres domaines car il n'y a pas la possibilité de leurs ajouter de nouvelles fonctionnalités.

Nous allons recenser les problèmes qui peuvent se poser et présenter des esquisses de solution :

- La VS demande beaucoup de ressources de calcul et implique un volume important de données. Un système de VSC doit donc gérer efficacement les calculs de la visualisation et la transmission des données.

- Il peut s'avérer difficile de faire accepter un système de VSC par des participants provenant d'horizons très divers. Un système collaboratif doit être adopté par tous les participants potentiels. Un tel système doit posséder une bonne utilité théorique et une bonne utilisabilité.

- D'un point de vue technique, la réalisation d'un système collaboratif est plus complexe qu'un système mono-utilisateur. Il faut s'assurer que les informations partagées soient identiques pour tous les participants. Il faut tenir compte de l'hétérogénéité du matériel à disposition des participants.

- Il est difficile de démontrer qu'un système est confortable et facile d'utilisation. La satisfaction étant un concept subjectif, il est malaisé de définir une métrique traduisant fidèlement le sentiment de satisfaction d'un utilisateur. Par contre, il est plus aisé de définir une métrique pour l'efficacité, par exemple en utilisant le temps d'attente.

1.6 Organisation de la thèse

Dans un premier temps, nous allons présenter les concepts liés aux systèmes collaboratifs ainsi qu'aux systèmes de VS. Ensuite, nous allons affiner les **besoins** des systèmes de VSC, cela permettant de définir encore plus précisément nos **objectifs**. Nous présentons des **solutions**, dont nous discutons les avantages et les inconvénients, afin de retenir les plus adéquates d'entre elles en vue de leurs réalisations dans le **système**.

Le chapitre 2 présente l'état de l'art dans les différents domaines qui nous concernent. Les principaux problèmes qui se posent sont développés dans le chapitre 3. Une discussion sur les différents modèles et la solution technique à l'efficacité du système sont présentés au chapitre 4. Le chapitre 5 présente nos réponses aux défis liés à la satisfaction. Les difficultés techniques ainsi que les détails concernant la conception du système se trouvent au chapitre 6. Le chapitre 7 présente nos conclusions et discute les développements futurs envisageables.

Chapitre 2 Visualisation et collaboration

Dans un premier temps, nous allons nous intéresser aux principaux modèles et systèmes de visualisation scientifique mono-utilisateurs. Dans un deuxième temps, nous présentons et discutons les notions de base qui sont liés aux systèmes collaboratifs dans leur ensemble.

Finalement, nous décrivons les principaux systèmes de visualisation scientifique collaborative.

2.1 Modèles et systèmes de VS mono-utilisateurs

Cette section présente les principaux modèles et systèmes de visualisation scientifique (VS) mono-utilisateurs. Cela nous servira de base pour discuter des systèmes de visualisation scientifique collaborative (VSC).

Selon Duce [26] un modèle de référence peut servir de base :

- pour comparer des systèmes et en réaliser d'autres ;
- pour concevoir des nouveaux systèmes, qui comportent des avantages par rapport à d'autres approches ;
- pour expliquer les défauts dans des systèmes existants et montrer des pistes pour les corriger.

Comme exemple nous citons le modèle de référence CGRM (*Computer Graphics Reference Model*) [24, 49] développé par l'ISO/IEC, qui est utilisé pour définir les standards dans le domaine du graphisme informatique.

CGRM définit des interfaces entre les représentations graphiques et des entités externes telles que l'application et l'utilisateur. Pour réaliser cela, le modèle utilise une suite de transformations entre l'application et l'utilisateur. Ce modèle aide à établir des liens entre les standards graphiques et des standards dans d'autres domaines comme les réseaux.

Il est important de noter que, dans cette section, nous décrivons des modèles et non des architectures ; un modèle décrit essentiellement des fonctionnalités, mais il peut être utilisé ensuite pour décrire une architecture.

2.1.1 Modèle du flux de données

Les **MVEs** (*Modular Visualization Environment* – Environnement Modulaire de Visualisation) utilisent les flux de données (*dataflows*) et une programmation visuelle. Le modèle du flux de données en VS, proposé par Haber [42], est un modèle de référence parmi les plus anciens et les plus répandus de la VS.

Un processus de visualisation basé sur les flux de données est décrit comme un ensemble de **trois transformations** (appelés parfois modules) principales chargées de transformer les données brutes en une image. La figure 2.1 représente un tel enchaînement de transformations.

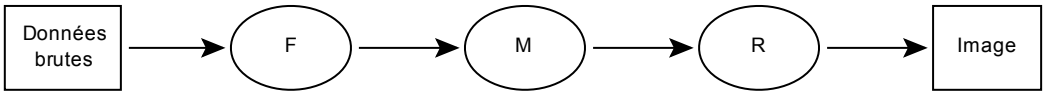

Figure 2.1 : Modèle du flux de données

La première transformation ***Filter* (F)** prend les données brutes (*raw data*) et les transforme en données dérivées. La transformation suivante, ***Map* (M)** convertit les données dérivées en une description géométrique. La dernière transformation ***Render* (R)** produit une image correspondant à cette description géométrique. Chaque transformation possède une entrée et une sortie qui doivent être connectées par l'utilisateur à une autre transformation. Les données brutes peuvent être interpolées par une transformation F ; cela va générer des données traitées qui vont servir à construire un objet graphique par une transformation M. Des rotations, des agrandissements, du *clipping*, des suppressions de surfaces cachées ou d'anti-aliasing sont des exemples typiques de transformations R.

Les systèmes basés sur le modèle du flux de données proposent un ensemble prédéfini de transformations. Un des avantages de tels systèmes est leur extensibilité. En effet les utilisateurs ont la possibilité de rajouter leurs propres transformations au flux de données ; ces transformations pouvant, par exemple, être programmées en C++, Fortran ou Java.

2.1.2 Systèmes de VS mono-utilisateurs

Nous distinguons deux types d'utilisateurs : **l'utilisateur final** (la personne utilisant le système pour visualiser ses données brutes) et **l'utilisateur-développeur** (la personne qui ajoute des fonctionnalités au système, par exemple en intégrant de nouvelles données à visualiser ou en développant des outils de visualisation ; cette personne possède la plupart du temps des connaissances en programmation). L'utilisateur final et l'utilisateur-développeur peuvent être une seule et même personne. Cette distinction est utile car les besoins de ces deux types d'utilisateurs sont différents.

Le besoin **d'extensibilité** est habituel dans la VS, parce que le format des données à visualiser ainsi que les techniques de visualisation varient fortement d'un domaine d'application à un autre. Il doit donc être possible d'étendre les systèmes en fonction des besoins spécifiques de l'utilisateur, que ces derniers concernent l'intégration de types de données, de techniques de visualisation ou d'interfaces utilisateurs. Des *wizards* ou des langages de script peuvent être utiles pour simplifier le développement d'extensions.

Dans les grandes lignes, on peut regrouper les systèmes de visualisation scientifique en quatre grandes catégories :

1) Les systèmes dédiés à un seul domaine : systèmes dédiés à un seul domaine de recherche, par exemple les systèmes commerciaux FieldView [59] et Fluent [31] pour la dynamique des fluides ou ITK [47] pour la médecine. La plupart de tels systèmes sont peu extensibles, ils ne sont efficaces que pour un domaine précis et il est difficile d'ajouter de nouvelles fonctionnalités. Par contre, comparé aux systèmes des autres catégories, ils sont faciles d'utilisation, faciles d'appropriation et simples d'apprentissage (comme expliqué à la section 1.4.2).

2) Les systèmes basés sur un MVE : on peut citer des systèmes commerciaux fréquemment utilisés comme AVS/Express [2], Iris Explorer [61, 105] ou Amira [97]. Les problèmes principaux auxquels sont confrontés les utilisateurs de MVE sont :

   - Les problèmes techniques à résoudre afin de permettre au système d'accéder aux données brutes. Ces systèmes n'acceptent qu'un nombre restreint de formats de données brutes, l'utilisateur doit donc s'adapter à un de ces formats.

   - L'effort d'apprentissage est assez important et il est souvent laborieux de créer des suites de transformations. Par exemple, la construction d'une isosurface par un MVE peut demander une suite de sept transformations.

3) Les librairies de développement : il s'agit de librairies spécialisées conçues pour être étendues à différents domaines d'application, par exemple la librairie VTK [89]. Destinées à des utilisateurs-développeurs, ces librairies demandent un effort assez grand pour obtenir un système. En contrepartie, elles offrent une grande flexibilité.

4) Autres systèmes : par exemple, Spray [78] se base sur la métaphore d'un spray qui envoie des particules qui réagissent aux données liées aux régions traversées. Un système basé sur cette métaphore rend difficile certaines extensions, comme une représentation d'un champ vectoriel par des textures LIC [12]. ZoomIn [86] utilise un concept basé sur les flux de données, fournit un mécanisme pour ajouter de nouvelles fonctionnalités et de nouveaux formats de données brutes.

A la vue de cette classification et des différents systèmes décrits, nous observons deux types de système, d'une part les systèmes faciles d'utilisation mais difficilement extensibles, d'autre part les systèmes extensibles (au prix d'un effort non négligeable) mais présentant une utilisation moins aisée.

Le choix d'un système par un utilisateur sera fortement influencé par les données qu'il désire visualiser, la manière dont il veut les visualiser et sa motivation à intégrer de nouvelles fonctionnalités. Un bon système de visualisation, susceptible d'être adopté par de nombreux utilisateurs, doit donc constituer un compromis entre la facilité d'utilisation et l'extensibilité.

2.2 Systèmes collaboratifs

Nous allons examiner les systèmes collaboratifs en général, établir les définitions de bases et tenter d'identifier les problèmes qui se posent lors de la conception d'un système collaboratif.

2.2.1 Mécanismes de collaboration

Dans cette section, nous allons aborder différents points de vue théoriques qui nous aideront à mieux comprendre certains mécanismes de la collaboration.

Le fait que, dans le cadre de la collaboration assistée par ordinateur, les participants ont accès aux mêmes informations, au même écran, ne signifie pas qu'ils partagent la même compréhension du problème. D'après Roschelle [84], trois conditions doivent être remplies pour que les participants parviennent à une représentation commune du problème :

1) Ils doivent pouvoir introduire et accepter de nouvelles connaissances à la représentation commune du problème.

2) Tout au long de leur activité, ils doivent rester attentifs à d'éventuelles divergences des représentations.

3) Ils doivent pouvoir réparer les divergences qui font obstacle à la progression de la collaboration.

Dans la section intitulée ***Grounding***, nous verrons de quelle manière des participants s'assurent qu'ils partagent bien une base commune de connaissances qu'ils peuvent exploiter pour construire leur interaction. La section ***Awareness*** traite de l'importance pour les participants de construire et maintenir une connaissance de l'espace de travail et des moyens pour y parvenir. La section **Média de communication** présente une catégorisation de différents moyens permettant aux participants de communiquer. Les sections suivantes traitent des ***Groupwares*** et des ***Frameworks*** **collaboratifs**.

2.2.2 Grounding

Les participants communiquent entre eux à travers l'envoi d'informations (messages). **L'émetteur** est l'initiateur d'un message, alors que le **récepteur** est la personne à qui est adressé ce message.

La collaboration repose largement sur ce que l'on appelle une **base de connaissances communes**. En particulier, cette base de connaissances communes permet à l'émetteur de formuler des messages qu'il pense être compréhensibles par le récepteur.

Par exemple, dans une conversation, il est important de donner le même sens aux mots sans quoi deux personnes ne pourraient pas se comprendre. En s'assurant de cette compréhension, ils peuvent alors considérer le message comme une contribution à leur base commune. Le ***grounding*** est le processus collectif par lequel les partenaires tentent d'enrichir leur base de connaissances communes [19].

L'effort collaboratif est le travail que fournissent deux participants du début de leurs interactions jusqu'à une acceptation mutuelle, c'est-à-dire l'énergie dépensée par les interlocuteurs pour effectuer un *grounding*. La quantité d'effort collaboratif n'est pas forcément liée à la réussite du *grounding*. Par contre, pour des raisons d'efficacité de la communication, il semble préférable de réduire l'effort nécessaire au *grounding*.

2.2.3 Awareness

En interagissant avec son environnement une personne produit une multitude de signaux. Ces signaux donnent aux autres participants une certaine connaissance de ses actions et des ses intentions. Cette connaissance des autres qui résulte des interactions avec l'environnement est souvent désignée par le terme « ***awareness*** » (qui peut être traduit par « être conscient de ce qui se passe » ou « être au courant de ce qui se passe »).

Il est cependant très difficile de définir l'*awareness* de manière précise [58]. Nous allons le constater en examinant les diverses définitions de l'*awareness* qui ont été proposées dans la littérature.

2.2.3.1 Définitions proposées dans la littérature

Une des premières définitions de l'*awareness* fut donnée par Dourish [23] : « l'*awareness* est une compréhension de l'activité des autres, qui fournit un contexte pour sa propre activité ».

Cette définition restreint donc l'*awareness* aux actions des utilisateurs. Une définition différente est donnée par Sohlenkamp [91] qui définit l'*awareness* comme « une compréhension de l'état global d'un système » ; dans ce cas, l'*awareness* ne tient pas forcément compte des actions individuelles des participants.

Gutwin, quant à lui, présente l'*awareness* comme « une connaissance créée à travers une interaction entre un agent et son environnement » [38]. Il se base sur les travaux de Endsley [29] pour qui l'*awareness* consiste à « savoir ce qu'il se passe » et qui voit quatre caractéristiques principales de l'*awareness* :

1) L'*awareness* est la connaissance de l'état d'un d'environnement délimité dans l'espace et le temps.

2) L'*awareness* est constitué de connaissances qui doivent être mises à jour en fonction des transformations de l'environnement.

3) L'*awareness* est entretenu par l'interaction des personnes avec l'environnement.

4) L'*awareness* est habituellement un objectif secondaire, qui sert à la résolution d'une tâche.

Ces définitions de l'*awareness* restent très générales. Il est nécessaire de décomposer l'*awareness* pour en donner des définitions mieux adaptées au domaine de la VSC. Gutwin distingue quatre types d'*awareness* : informel, social, structure de groupe et espace de travail. Ces catégories ne sont pas mutuellement exclusives et sont combinables.

1) ***Awareness* informel.** L'*awareness* informel d'une communauté de travail est la conscience générale de qui se trouve alentour, et de ce que chacun fait. C'est le genre d'informations dont on dispose lorsque des personnes travaillent dans le même bureau. L'*awareness* informel est acquis au hasard d'une rencontre à la machine à café ou à l'imprimante et facilite les interactions informelles.

2) ***Awareness* social**. L'*awareness* social regroupe les informations concernant les autres dont une personne dispose dans un contexte social ou conversationnel : l'attention que porte un interlocuteur, son état émotionnel, ou son niveau d'intérêt. Ces informations sont obtenues grâce aux réactions des interlocuteurs et aux indices non verbaux comme le regard, l'expression faciale, ou encore le langage du corps. Dans une conversation en face-à-face, les sources d'information sont l'expression du visage, les signes et des onomatopées telles que « hum, hum ».

3) ***Awareness* de la structure de groupe**. L'*awareness* de la structure de groupe est la connaissance de caractéristiques telles que le rôle et les responsabilités, la position par rapport à un but, et le statut de chacun. L'*awareness* de la structure de groupe concerne le protocole et les structures utilisés pour formaliser la collaboration. Les questions concernant les rôles et les responsabilités, les droits d'accès à une source d'information, le déroulement d'une rencontre formelle font partie intégrante de la notion d'*awareness* de la structure de groupe. Liechti [58] le caractérise par son but qui est de fournir des informations sur l'état et l'activité des gens au sein d'une équipe. L'*awareness* de groupe facilite la coordination entre les personnes et fournit des indicateurs utiles pour amorcer une communication ou une collaboration.

4) ***Awareness* de l'espace de travail**. Le quatrième type est l'*awareness* de l'espace de travail (*Workspace Awareness* – WA), sur lequel le plus de travaux ont été réalisés. C'est sans doute l'aspect le plus intuitif de l'*awareness*. En effet, c'est la conscience de l'activité du groupe (c'est-à-dire de chacun des individus du groupe) sur l'espace de travail. Les points majeurs du WA concernent l'identité des personnes se trouvant dans l'espace de travail, leurs positions, leurs activités ainsi que les changements immédiats qu'ils provoquent sur l'espace de travail.

Dans nos travaux, nous allons utiliser le terme « *awareness* » au sens du WA comme défini par Gutwin. En effet, en VSC, les interactions se produisant entre les participants et l'espace de travail sont les informations principales de la collaboration. L'interaction de plusieurs participants dans un espace de travail génère une multitude d'informations ; il est possible de diviser ces informations en différents composants du WA. Ces composants ont pour objectif de couvrir tous les besoins en *awareness*, même ceux appartenant aux autres types d'*awareness*. Gutwin propose trois catégories qui regroupent les différents types d'informations en rapport avec un événement :

1) **Qui**, c.-à-d. qui sont les autres participants ?

2) **Quoi**, c.-à-d. que font les autres participants ?

3) **Où**, c.-à-d. où travaillent les autres participants ?

Dans ces trois catégories, Gutwin a identifié des **éléments** spécifiques de connaissances qui constituent l'essentiel du WA. Le tableau 2.1 présente de manière détaillé ces éléments et les questions auxquelles chaque élément peut répondre.

Cette décomposition permet aux concepteurs de définir le niveau d'*awareness* qui doit exister dans leurs systèmes.

Tableau 2.1 : Eléments de l'*awareness*

Catégorie	Eléments	Questions
Qui	Présence Identité Identification de l'auteur	Y-a-t-il quelqu'un dans l'espace de travail? Qui participe? Qui est-ce? Qui fait cela?
Quoi	Action Intention *Artefact*	Que font-ils? Quel est le but de cette action? Sur quel objet sont-ils en train de travailler?
Où	Localisation Regard Vue Portée	Où sont-ils en train de travailler? Où sont-ils en train de regarder? Que peuvent-ils voir? Quel est leur champ d'action?

On peut identifier trois différentes sources d'informations :

1) **Les *artefacts***. Les objets dans un espace de travail (« *artefacts* ») peuvent fournir des informations telles que la relation spatiale à d'autres objets, des états visuellement distincts de l'objet en fonction de son état, etc. Les informations recueillies par le récepteur sont les indicateurs d'un changement de l'état d'un objet consécutif à une manipulation par une autre personne ; ces informations sont appelées « *feedthrough* ».

2) **Le corps dans l'espace de travail**. Le corps d'un partenaire en train d'interagir avec l'espace de travail fournit entre autres des informations tels que la situation dans l'espace, la posture, le mouvement de la tête, des bras, des yeux et des mains. Il s'agit d'une forme de communication non-verbale, surtout visuelle, mais avant tout non intentionnelle. Les informations recueillies par le récepteur sont les conséquences d'une communication non explicitement voulue par l'émetteur, c'est-à-dire une communication implicite ; appelée « *consequential communication* ».

3) **Conversation et gestuelle**. Lors d'une conversation, les informations peuvent être recueillies de trois manières. Premièrement, dans une discussion des partenaires peuvent s'échanger des informations concernant des éléments d'*awareness*, par exemple en disant où ils sont ou ce qu'ils font. Deuxièmement, lorsque deux personnes conversent directement il se peut qu'une troisième personne puisse entendre les informations qu'elles échangent. Troisièmement, une personne peut se parler à elle-même lorsqu'elle accomplit une tâche, « *shadowing* », ce qui peut fournir indirectement des informations à une autre personne. La conversation et la gestuelle sont deux formes de communication intentionnelle ; appelée « *intentional communication* ».

2.2.3.2 Informations statiques ou dynamiques

Une des nécessités dans un travail collaboratif est de trouver les bonnes personnes avec qui collaborer. Danis [21] propose pour cela de fournir un certain nombre d'informations sur les personnes afin de permettre aux participants de choisir leurs collaborateurs. Pour cela, l'auteur fait une distinction importante entre les données concernant une personne qui sont statiques et celles qui sont dynamiques.

Les sources de données statiques sur la personne sont en premier lieu les informations qui renseignent un participant avant qu'il se connecte à un système (par exemple son profil, ses centres d'intérêts ou ses compétences dans tel ou tel domaine). Cependant, il est difficile de prévoir à l'avance quelles sources de données statiques doivent être collectées.

Les données dynamiques dérivent du comportement de la personne. Elles sont collectées au travers de l'observation de l'activité de la personne.

Gutwin définit l'*awareness* comme de la connaissance sur un environnement dynamique mais nous pensons que les informations statiques doivent aussi être prises en compte. En effet, elles sont également utiles pour comprendre ce qui se passe.

2.2.3.3 Situation de collaboration synchrone ou asynchrone

Il parait clair qu'au niveau applicatif, l'*awareness* va être différent suivant que la situation de collaboration est synchrone ou asynchrone. En effet, pour un système synchrone, le support de l'*awareness* va se faire, par exemple grâce à des outils tels que les vues par des avatars. Alors que dans un système asynchrone, la tâche peut se dérouler sur plusieurs jours ou même sur plusieurs années. Dans ces conditions, les participants ne se trouvent pas forcément dans l'espace de travail au même moment et des outils tels que des vues sont moins utiles. Dans ce cas, par exemple, des historiques détaillant tous les changements ayant eu lieu entre deux connexions à l'espace de travail peuvent être beaucoup plus adéquats.

On remarque que les solutions sont différentes pour une situation synchrone ou une situation asynchrone. Malgré cela, les informations présentées au participant dans ces deux situations sont semblables. En effet, les deux solutions citées ci-dessus permettent de renseigner les participants sur les actions des autres participants. Il convient donc pour la suite de garder à l'esprit que le synchronisme d'un système influence les solutions à utiliser.

2.2.3.4 Informations privées sur les participants

Dans le domaine de l'*awareness*, nous distinguons deux types de problèmes liés au respect des informations privées sur les participants :

1) Il ne faut pas perturber le travail des participants, ni agir de manière trop intrusive dans le travail des participants, par exemple en affichant trop souvent des messages. Il faut un compromis entre le fait d'informer correctement quelqu'un et le fait de le distraire [37].

2) Un des problèmes récurrents en informatique est le respect de la sphère privée. Il ne faut pas collecter et présenter trop d'informations sur tous les faits et gestes des participants. Les modifications effectuées par quelqu'un ne devraient être rendues visibles que lorsque celui-ci le décide.

Il convient donc de permettre aux participants de choisir dans quelle mesure ils veulent de l'*awareness* et ne pas donner des informations non désirées. En effet, certains participants peuvent être gênés d'être « observés » et peuvent même refuser d'utiliser un tel système.

2.2.4 Média de communication

Le terme « **média de communication** » désigne un moyen de communication tel que la parole ou l'écriture. Clark [19] identifie huit contraintes qu'un média de communication peut imposer à la communication entre deux personnes :

1) La coprésence : présence des interlocuteurs dans un même environnement.

2) La visibilité : les interlocuteurs se voient mutuellement.

3) L'audibilité : les interlocuteurs peuvent s'entendre mutuellement.

4) La cotemporalité : un énoncé est perçu par le récepteur au moment où il est produit par l'émetteur.

5) La simultanéité : les interlocuteurs peuvent simultanément émettre et recevoir.

6) La séquentialité : les tours de parole se succèdent les uns les autres.

7) La re-consultabilité (*reviewability*) : possibilité de consulter les précédents messages de son interlocuteur.

8) La révisabilité (*revisability*) : possibilité de réviser un message avant de l'envoyer.

Il est possible de donner une description d'un média en fonction des contraintes qui lui sont associées. Par exemple, les contraintes associées à la communication en face-à-face sont la coprésence, la visibilité, l'audibilité, la cotemporalité, la simultanéité et enfin la séquentialité.

Lors de la conception d'un environnement collaboratif, on doit tenir compte des contraintes spécifiques des médias de communication et du profil de coûts qu'elles impliquent, et réaliser en conséquence l'interface de communication adéquate.

2.2.5 *Groupwares*

Le terme « *groupware* » (appelé aussi *Computer-Supported Cooperative Work systems-CSCW, collaborative computing, multi-user systems* ou *workgroup computing*) est un terme générique qui désigne des systèmes utilisés par un groupe, par exemple des systèmes d'emails, de vidéoconférences ou de calendriers partagés.

Ellis [28] donne la définition suivante du terme « *groupware* » :

Computer-based system that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment.

Grudin [35] énonce huit défis que doivent affronter les concepteurs de *groupware* pour que leur système soit utilisé dans la pratique :

1) *Disparity in work & benefit* : Le bénéfice ainsi que le temps d'apprentissage doivent être les mêmes pour tous.

2) *Critical mass & Prisoner's dilema* : Il faut un certain nombre de personnes avant d'obtenir un gain lié à l'utilisation du CSCW. Il faut que les participants apprennent à collaborer.

3) *Social, political & motivational factor* : Il ne faut pas briser de facteur social.

4) *Exception handling* : Il faut prévoir des erreurs d'utilisation.

5) *Designing for infrequently used features* : Il faut conserver les aspects d'un système mono-utilisateur et faciliter l'accès aux fonctions peu souvent utilisées.

6) *Difficulty of evaluation* : En raison de la nature distribuée de la collaboration, il est difficile d'évaluer le système.

7) *Failure of intuitivity* : Un système doit être intuitif afin d'être accepté.

8) *Adoption procedure* : La majorité des personnes doit pouvoir utiliser facilement le système.

L'email est l'exemple d'un *groupware* qui a réussi à relever avec succès ces huit défis. Soulignons le fait qu'il faut se placer dans la perspective de l'utilisateur plutôt que dans une perspective technique pour affronter ces huit défis.

2.2.6 *Framework* collaboratif

Au niveau technique, il existe des *frameworks* collaboratifs qui prétendent pouvoir être aisément appliqués à de nombreux domaines, tel que la visualisation scientifique. Habanero [18], GroupKit [85] ou DOVE [54] sont des exemples de *frameworks* collaboratifs extensibles à différents domaines. Ils proposent des fonctionnalités de collaboration, que ce soit pour la communication (par exemple en offrant des *chats*, des systèmes audio-vidéo ou

des tableaux noirs partagés), pour la gestion des objets partagés (par exemple en réalisant une gestion client-serveur ou une gestion répartie) ou au niveau des fonctionnalités de base (par exemple en gérant des connexions au système ou l'ajout/départ de participants).

D'après Lopez [34], la majorité des *frameworks* collaboratifs ont des défauts conceptuels ou techniques qui font que réaliser un système de visualisation scientifique efficace se révèle être une tâche très ardue. Néanmoins, certains *frameworks* collaboratif conçus spécialement pour la visualisation scientifique, comme par exemple Shastra [3], peuvent être utilisés dans le cadre de la VSC.

2.3 Modèles de la VSC

Nous allons brièvement examiner quelques modèles de la VSC. Nous allons nous servir d'un modèle pour décrire différentes architectures, plus ou moins adaptées à différentes situations.

2.3.1 Modèle de Wood

Le modèle de Wood [108] est une extension collaborative du modèle du flux de données [42] ; il est décrit par la figure 2.2.

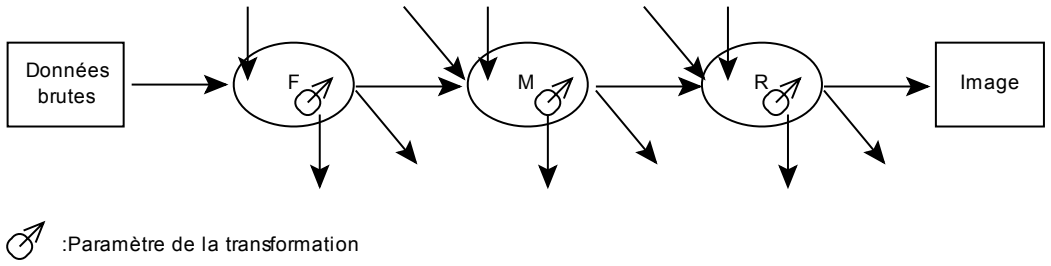

Figure 2.2 : Modèle de Wood

Comme dans le modèle du flux de données (expliqué à la section 2.1.1), **F** représente le filtre de transformation, **M** (*mapping*) le calcul de l'objet de visualisation et **R** le rendu. Les flèches horizontales indiquent la transformation des données brutes à travers les différentes transformations pour aboutir à une image. A chaque étape, l'information de contrôle peut être importée ou exportée d'un autre flux de données ; elle est représentée par une flèche verticale issue du symbole « paramètre de la transformation » ou y aboutissant. D'une manière analogue, les données peuvent être, à chaque étape, importées ou exportées d'une autre transformation ; dans la figure, cela est indiqué par les flèches diagonales.

Une approche pour pouvoir collaborer consiste à distribuer chez les participants des parties du système : les transformations du modèle peuvent être exécutées à différents endroits, en un ou plusieurs exemplaires. Par exemple, la figure 2.3 montre un système dans lequel chaque participant peut avoir sa propre transformation de rendu R, alors qu'il n'y a qu'une seule transformation de filtre F, localisée chez un des participants.

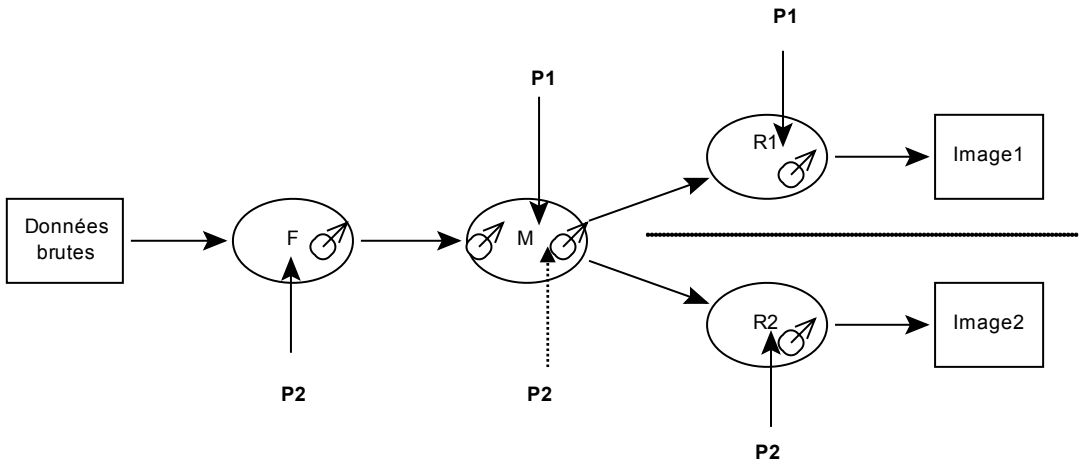

Figure 2.3 : Exemple de partage selon le modèle de Wood ; les participants ont leur propre vue

Ici, le participant P2 possède le contrôle des paramètres de la transformation filtre F. Le contrôle des paramètres de la transformation M est partagé entre P1 et P2. Chaque participant peut agir sur sa propre transformation de rendu (R1 et R2). Concernant la transformation M, P1 est le maître (flèche pleine) et P2 l'esclave (flèche en traitillé). Le maître peut modifier les paramètres de la transformation tandis que l'esclave ne peut que les lire. Précisons que la notation n'indique pas explicitement lequel des deux paramètres de M est contrôlé.

Cette notation permet de représenter les caractéristiques suivantes :

- Une visualisation peut être décrite par un processus comprenant trois étapes : F, R et M. Cela correspond au modèle du flux de données (se référer à la figure 2.1).

- Chaque étape du processus est contrôlée par un ensemble de paramètres.

- Un système de visualisation collaborative est représenté par un ensemble de transformations qui peuvent être contrôlées par un participant. Cet ensemble peut être complet (c'est-à-dire contenir toutes les étapes) ou partiel (c'est-à-dire contenir quelques étapes, par exemple les étapes F et M comme illustré par la figure 2.4 pour le participant P2).

- L'information de contrôle peut être exportée d'une transformation aux autres transformations de façon à synchroniser les valeurs des paramètres entre les différentes étapes. Cette notation permet de décrire les détails à un moment précis, comme dans l'exemple illustré par la figure 2.4, où les deux participants possèdent la même vue contrôlée par le participant P1.

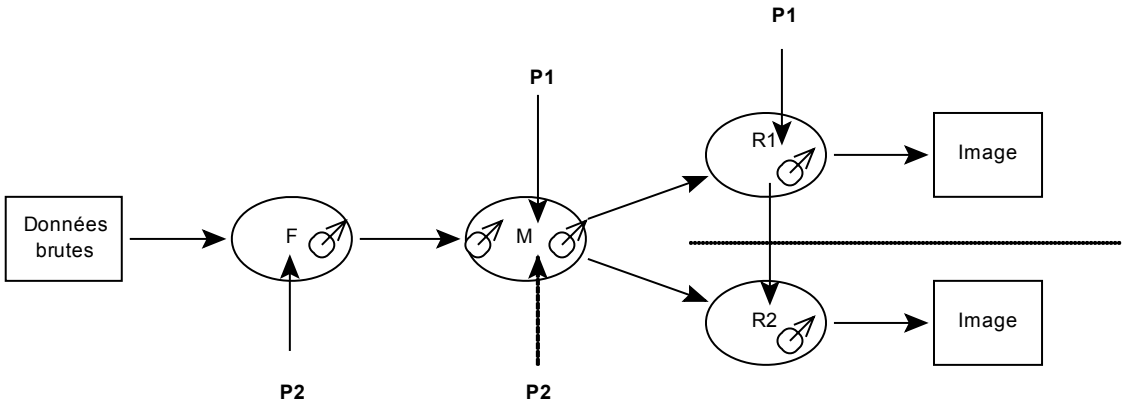

Figure 2.4 : Exemple de partage selon le modèle de Wood ; les participants ont la même vue

2.3.2 Modèle de Duce

D'autres modèles ont été proposés : citons celui de Duce [26] qui étend le modèle de Wood en tenant compte des participants et en distinguant explicitement chaque étape du processus. Il ne se restreint pas seulement à trois transformations, on peut ajouter des transformations particulières, telle que des transformations de contrôle ou de synchronisation.

Les ellipses représentent les transformations comme décrit dans le modèle du flux de données, les rectangles représentent des données et les flèches allant d'une transformation à un nom de participant (Pi) indique le participant qui contrôle l'état de cette transformation. On peut distinguer deux types de contrôle : **actif** ou **passif**. Un contrôle est actif lorsqu'un participant peut modifier la valeur des paramètres de ce contrôle. Il est passif lorsqu'un participant peut uniquement voir la valeur des paramètres. Cela est indiqué par la direction des flèches allant d'un participant à une transformation. De plus, un contrôle actif peut recevoir des mises à jour (*feedback*). Cela est illustré par la figure 2.5.

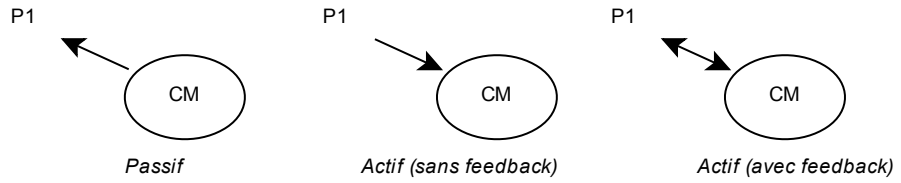

Figure 2.5 : Type de contrôle selon le modèle de Duce

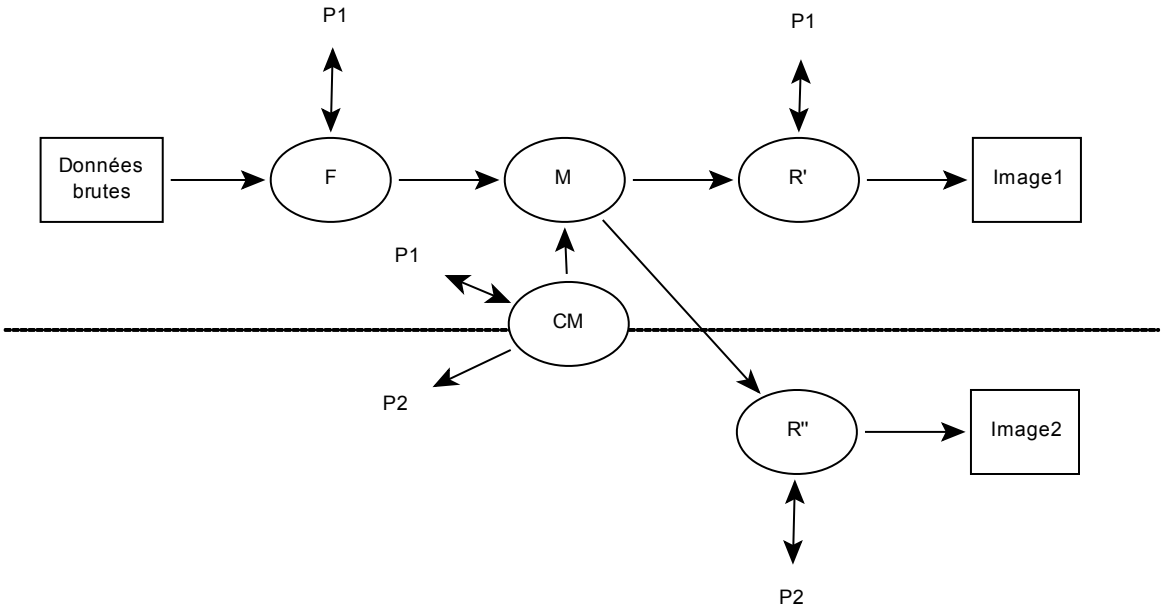

Figure 2.6 : Exemple de partage selon le modèle de Duce

Un exemple de partage tiré du modèle de Duce est illustré dans la figure 2.6. Dans cette figure, la transformation CM (*Control Module*) permet de contrôler l'accès à la transformation M. Selon Duce, les avantages de ce modèle comparé au modèle de Wood sont les suivants :

- La granularité des processus de transformation n'est pas limitée aux transformations filtre, *mapping* et rendu du modèle du flux de données. Cela permet de représenter plus de détails, par exemple pour décrire et comparer entre eux des systèmes MVEs.

- La représentation des entrées-sorties des transformations est plus explicite (par exemple, la transformation CM de la figure 2.6). Cela facilite un partage personnalisé des données, surtout lorsque plus de deux participants collaborent.

- Ce modèle permet d'identifier le participant qui contrôle une transformation d'une manière plus explicite.

2.4 Systèmes de VSC

Nous discutons quelques systèmes, sans vouloir présenter une liste exhaustive, en nous limitant à ceux qui introduisent des concepts novateurs, qui sont le plus utilisés en pratique ou qui possèdent des objectifs semblables aux nôtres.

Nous pouvons classifier ces systèmes en trois catégories :

1) Les systèmes de **VSC simplifiés**, qui sont destinés à être utilisés par un petit nombre de participants (un seul participant à la fois peut interagir avec le système) et qui ne possèdent pas de support de l'*awareness*.

2) Les systèmes de **VSC complets** qui permettent à un plus grand nombre de participants de collaborer et qui possèdent un support de l'*awareness*.

3) Les systèmes basés sur un **environnement de réalité virtuelle**. Même si la VSC n'est pas l'objectif principal de tels systèmes, il est possible de les utiliser grâce à leur support de l'*awareness*.

2.4.1 Systèmes de VSC simplifiés

Certains systèmes ne possèdent que des **fonctionnalités élémentaires de collaboration**. On peut distinguer ceux qui sont basés sur une approche client-serveur directement intégré au système de visualisation scientifique (VS) et ceux dont les fonctionnalités collaboratives sont basés sur un système annexe.

2.4.1.1 Systèmes utilisant une approche client-serveur

Une architecture client-serveur est souvent utilisée pour combiner les avantages de deux stations à caractéristiques différentes. En VS, un exemple classique consiste à faire calculer les objets graphiques par un processeur puissant (le **serveur**) et à faire afficher les objets graphiques produits par une station graphique (le **client**), un protocole adéquat assurant la communication entre client et serveur.

Comme exemple de tel système, on peut citer Semotus Visum [62]. Engels [30] présente un système qui combine une station de travail standard avec une station graphique afin que le participant utilisant la station graphique puisse profiter des performances de la station standard.

2.4.1.2 Systèmes utilisant le partage d'écran

Une autre manière de collaborer est d'utiliser un système de VS mono-utilisateur et de le combiner avec un système de partage d'écran : on reproduit sur les écrans des **clients** ce qui s'affiche sur l'écran du **serveur**, qui effectue ainsi tous les calculs. Il s'agit d'une des plus anciennes et des plus simples méthodes pour créer un système collaboratif à partir d'un système mono-utilisateur.

Les systèmes X Server sous Unix, MS NetMetting [64] pour Windows ou le système multi-plateforme VNC (*Virtual Network Computing*) sont des exemples de tels systèmes. Le système VNC a été développé par AT&T et est actuellement utilisé par deux projets *open source* : RealVNC [82] et TightVNC [99], ce dernier offrant différents algorithmes de compression.

Il est évident que ces systèmes ne permettent qu'une collaboration très partielle. En effet, dans ce cas, seul un participant à la fois peut interagir avec le système ; les participants ne disposent évidemment pas d'un espace de travail privé. De plus, le réseau reliant les différentes stations doit posséder une importante bande passante : chaque modification du contenu de l'écran provoque l'envoi d'une copie d'écran. Pour ces raisons, il n'est guère praticable d'utiliser cette méthode dans le cadre de la visualisation scientifique [87]. Néanmoins, cette méthode est utilisée pour la visualisation dans le cadre de grilles informatiques (*grid*) [10].

2.4.2 Systèmes de VSC complets

Nous examinons quelques exemples de systèmes de VSC en nous concentrant sur l'aspect collaboratif ; certains sont issus de projets de recherche et d'autres de systèmes commerciaux.

Ses systèmes peuvent être divisés en deux sous-catégories suivant qu'ils sont basés ou non sur une approche MVE.

2.4.2.1 Systèmes de VSC basés sur un MVE

Du point de vue de la collaboration, les systèmes basés sur une approche MVE ont comme principal avantage une approche visuelle, qui permet d'affecter des droits d'accès sur les transformations et de partager sa visualisation avec les autres participants.

Comme défaut de tels systèmes, nous notons une difficulté pour les participants à gérer les différentes transformations ou un manque d'*awareness* (par exemple la difficulté à situer les autres participants dans l'environnement de travail).

2.4.2.1.1 VisAD

VisAD (*Visualization for Algorithm Development*) [45] est une librairie conçue pour l'analyse et la visualisation interactive de données. Elle est disponible gratuitement [104]. Cette librairie se base sur un modèle de données qui peut être adapté à tous les types de données numériques et qui permet le partage de données. Ce système est portable, car il est écrit en Java et utilise Java RMI [96] pour la partie réseau. Cette librairie comprend en outre une extension destinée à la collaboration. Le partage de données et de transformations (au sens MVE) est possible, de même que la possibilité de répartir les calculs sur les différents ordinateurs à disposition.

VisAD peut fonctionner en deux modes : **autonome** et **client-serveur**. Le mode **autonome** est le mode standard sans utilisation du réseau. En mode **client-serveur**, le **serveur** se différencie du mode autonome par le fait que, une fois que les données ont été chargées, le système peut répondre aux requêtes des clients. Le **client** permet de se connecter à un serveur et de recevoir toutes les informations nécessaires pour se connecter à une **vue** (une vue est un objet de l'architecture de VisAD qui contient une fenêtre dans laquelle s'affichent des objets et des contrôles). Le client et le serveur se synchronisent sur cette vue. Les informations proviennent du serveur ; si le serveur s'arrête, la vue du client ne fonctionnera plus. Tous les participants peuvent donc créer des objets et interagir avec.

Comme exemple d'application collaborative basé sur VisAD, citons « *Milky Way Galaxy Designer* » (disponible sur le site web de VisAD) qui permet de créer une carte du ciel en fonction de données provenant d'observations faites depuis la Terre. Tous les participants voient les paramètres et peuvent les faire varier.

2.4.2.1.2 IRIS Explorer - COVISA

IRIS Explorer [61, 105] est un système MVE basé sur le modèle du flux de données. Ce système propose une sélection de transformations (au sens MVE) et offre la possibilité aux utilisateurs de créer leurs propres transformations, que ce soit pour produire un objet graphique par un algorithme de visualisation ou pour intégrer de nouvelles données.

La partie collaborative est fournie grâce à un ensemble de transformations produites dans le cadre du projet COVISA (*Co-Operative working in VIsualization and Scientifique Analysis*) [109]. Ces transformations permettent de se connecter à une session collaborative et fournissent des outils permettant de partager les transformations. Les données et les contrôles peuvent être partagés entre les instances d'IRIS Explorer des différents participants. Les

différentes transformations doivent être reliées manuellement, il en résulte que l'on ne peut donc pas optimiser automatiquement les transmissions de données. La figure 2.7 montre le fonctionnement de COVISA : chaque participant à son point de vue pour visualiser les objets graphiques ; les participants ont la possibilité d'affecter des droits d'accès en lecture/écriture sur des objets graphiques, par exemple un participant peut créer un plan de coupe qu'il est le seul à voir. Un serveur s'occupe de gérer les informations partagées.

Le projet gViz [41] a étendu IRIS Explorer à la collaboration afin de pouvoir accéder à des grilles informatiques (*grid*) [8].

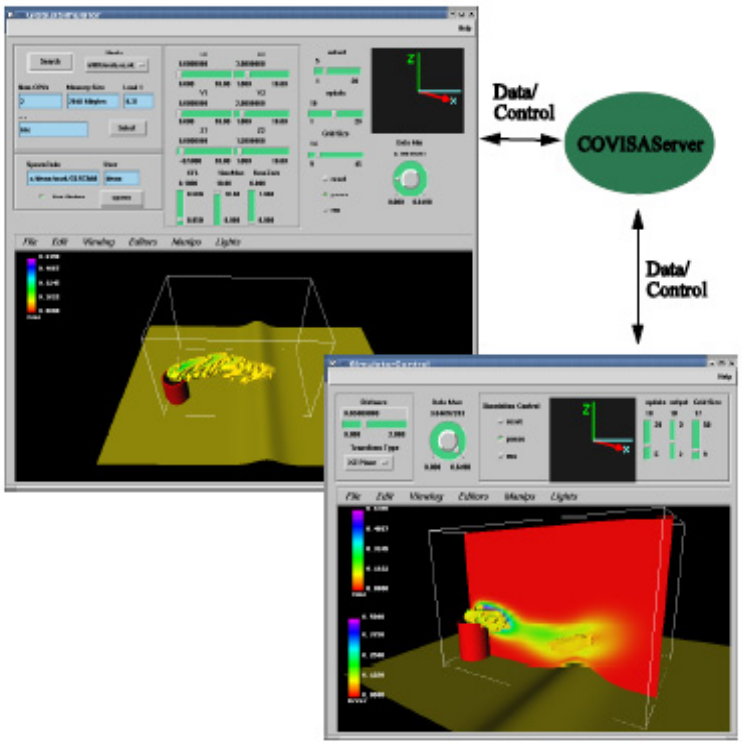

Figure 2.7 : Deux participants travaillant avec COVISA

2.4.2.1.3 AVS - MANICORAL DCV

AVS/Express, créé en 1989, est le premier système utilisant une approche basée sur un MVE. Il est actuellement l'un des systèmes les plus utilisé.

AVS, sans autre qualificatif, désigne les premières versions de ce système, tandis que les nouvelles versions encore en cours de développement sont appelées AVS/Express. Le projet Collaborative AVS [52] utilise AVS afin d'obtenir un système collaboratif.

MANICORAL DCV [25] est réalisé comme un ajout au système AVS/Express. Chaque participant doit lancer sa propre version d'AVS/Express. Un serveur central gère les communications entre les différents participants et le partage des transformations. Des droits d'accès permettent de protéger ses données. Il n'y a pas de support spécifique pour l'*awareness*.

2.4.2.2 Systèmes de VSC non basés sur un MVE

Les systèmes suivants ne sont **pas** basés sur **l'approche MVE**. Ils sont issus de projets de recherche.

2.4.2.2.1 CSpray

CSpray [79] est un système de visualisation collaborative basé sur le modèle de Spray [78], qui utilise la métaphore d'un spray : les participants utilisent un spray pour gicler des particules colorées qui sont envoyées dans l'espace et qui réagissent en fonction des données traversées pour créer des objets de visualisation abstraits (*Abstract Visualization Object* – AVO).

En particulier, CSpray possède les fonctionnalités collaboratives suivantes. Les sprays et les AVO peuvent être privés, c'est-à-dire utilisés uniquement par leur créateur, ou publiques, c'est-à-dire utilisable par tous. Ce système utilise des avatars pour représenter la position et le point de vue des participants dans un monde 3D virtuel. Des pointeurs permettent d'indiquer les endroits intéressants. Il est conçu pour un nombre assez petit de participants, c'est-à-dire moins de cinq.

La figure 2.8 montre une copie d'écran de ce système : la fenêtre principale est la vue du participant B et on aperçoit la position de l'œil du participant A. La fenêtre publique en bas à droite contient la vue de A ; ici A peut voir l'œil de B. La fenêtre publique permet à B de se connecter à la vue de A et ainsi les deux participants peuvent avoir une vue identique.

Ce système possède l'avantage de mettre à disposition des participants un espace public, un espace privé, des avatars et un contrôle des droits d'accès.

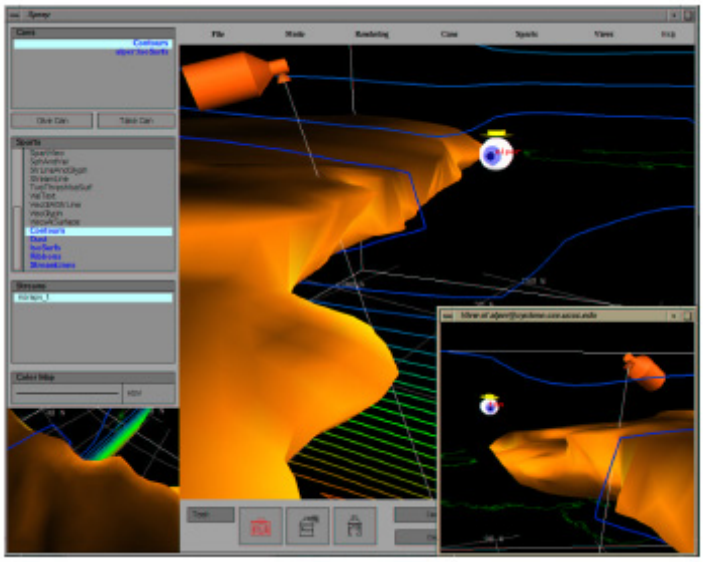

Figure 2.8 : Espace de travail de CSpray avec deux participants

2.4.2.2.2 FASTexpeditions

FASTexpeditions [67] est une extension collaborative d'un système d'analyse de flux, FAST [66], développé par la NASA. FAST est un ensemble de programmes qui interagissent pour visualiser le résultat de simulations numériques, dans le domaine de la dynamique des fluides.

Dans la version collaborative, la visualisation produite par un participant (le **pilote** selon la terminologie de FAST) est envoyée aux autres participants (les **passagers**) ; le pilote contrôle et dirige la visualisation, ce qui lui permet de guider les passagers à travers le processus de visualisation. Ce système utilise une technique de jeton pour gérer le contrôle ; le pilote peut le passer pour donner le contrôle à un passager.

2.4.2.2.3 CEV

CEV [7] est un système de visualisation scientifique qui permet à de multiples personnes de collaborer grâce à une interface basée sur une page web, en séparant l'image finale des composants 3D et en utilisant un serveur. Le serveur s'occupe d'effectuer tous les calculs et produit une image qui est envoyée au client ; le client peut ensuite la visualiser localement, par exemple directement dans un navigateur web, sans avoir besoin d'installer un programme. L'ordinateur sur lequel fonctionne le serveur doit être puissant, alors que les clients ont besoin de très peu de capacité de calcul. Chaque participant possède sa propre vue et une vue globale. Les participants ont la possibilité de prendre le contrôle de la visualisation afin de pouvoir effectuer une action sans que les autres participants puissent interagir.

2.4.3 Environnements de réalité virtuelle

Nous donnons quelques exemples de visualisation dans le cadre d'environnements de réalité virtuelle. Un descriptif plus complet est donné par van Dam [100].

Plusieurs projets ont été développés dans le cadre de la VR (***Virtual Reality*** – Réalité Virtuelle) coopérative. Ils permettent à des participants situés en différents endroits d'avoir une représentation d'eux-mêmes (avatars), d'explorer le même monde 3D et de communiquer.

Comme exemples d'environnement de réalité virtuelle, CAVE [20] ou DIVE [11] permettent à plusieurs participants d'interagir dans un monde 3D virtuel. Cette technique peut être appliquée dans le cadre de la VSC, en effet il suffit que les objets géométriques produits par un système de visualisation soient situés dans un monde 3D virtuel. Les résultats du projet VIVRE [98] montrent que les participants résolvant des problèmes complexes ont besoin d'avoir un contrôle sur le processus de visualisation [6], ce qui n'est pas le cas pour la majorité des systèmes de VR. Steed [94] utilise un environnement de réalité virtuelle collaboratif pour visualiser des images produites par un scanner IRM ; l'environnement utilise un système similaire à celui de CAVE.

Les techniques d'AR (*Augmented Reality* – Réalité Augmentée) peuvent aussi s'appliquer dans le cadre de la VSC. Le projet Studierstube [95] [33] permet à plusieurs participants de collaborer sur des visualisations en 3D, par exemple pour étudier des phénomènes physiques. Chaque participant porte un casque HMD (*Head Mounted Display*) avec des lunettes 3D et possède donc son propre point de vue. De nouveaux objets graphiques peuvent être ajoutés dans l'environnement par tous les participants.

2.5 Conclusion

Nous avons présenté les différentes notions jouant un rôle dans le domaine de la collaboration ainsi que divers systèmes de visualisation scientifique. Il en ressort que les systèmes de VSC ne satisfont pas à toutes les exigences du travail collaboratif. Certains sont mieux adaptés à la collaboration (par exemple grâce à un support de l'*awareness* ou à une gestion des droits d'accès) alors que d'autres attachent plus d'importance à l'efficacité (par exemple grâce à des

possibilités de coopération pour répartir les calculs). Dans le chapitre suivant, nous détaillons les besoins que de tels systèmes doivent satisfaire afin d'être adoptés le plus aisément possible ainsi que les défis à relever afin de satisfaire ces besoins.

Chapitre 3 Défis à relever

Nous allons décomposer le problème en deux aspects fondamentaux : **visualisation** et **collaboration**. Les principaux défis se trouvent dans le second aspect, le premier étant moins intéressant, car de nombreux problèmes ont déjà été résolus [101]. Mais la visualisation sera traitée dans cette section car les deux aspects sont fortement liés.

Chacun de ces aspects sera décomposé en quatre parties : **les besoins fondamentaux** pour les utilisateurs, les **objectifs** à atteindre, les **problèmes** que posent ces objectifs et les **approches de solutions.**

3.1 Aspect visualisation scientifique mono-utilisateur

Comme indiqué à la section 1.4, pour qu'un système satisfasse les besoins des utilisateurs, il doit être **effectif**, **fiable** et **efficace.** Pour rappel, nous ne désirons pas obtenir un système dédié à un seul domaine mais un système **extensible**. Comme chaque utilisateur est différent et comme chaque domaine ayant recourt à la visualisation scientifique (par exemple l'océanographie, la médecine ou la science des matériaux) a des besoins en visualisation particuliers, il faut prévoir un système adaptable aux besoins des différents utilisateurs et aux différents domaines d'application. Cependant, comme une partie des besoins est commune, une base qui peut être étendue en fonction des besoins spécifiques des utilisateurs se révèle utile.

En outre, en tenant compte des exigences ergonomiques, il ressort qu'un système de VS doit posséder les fonctionnalités suivantes :

- Effectivité : visualisation pertinente des données et extensibilité. Concernant l'**extensibilité**, l'ajout des fonctionnalités suivantes doit être possible :
    - Ajout de nouveaux outils de visualisation : un système doit posséder des outils efficaces pour visualiser les données ; des outils de base doivent être fournis et de nouveaux doivent pouvoir être créer.
    - Ajout de nouveaux types de données : chaque utilisateur pouvant produire des données dans son propre format, le système doit pouvoir aisément accéder à de nouveaux types de données.
- Satisfaction : le système doit offrir un confort d'utilisation et les interactions avec le système doivent être simples et intuitives
- Efficacité : l'accès aux données doit se faire simplement et dans un temps minimal.
- Facile d'apprentissage et d'appropriation.
- Fiabilité.

L'utilisabilité doit être présente malgré le fait que la facilité d'apprentissage et l'extensibilité sont souvent des objectifs contradictoires. Une analyse des systèmes existants est utile pour trouver un compromis satisfaisant entre ces deux objectifs. Il en ressort, que dans la majorité

des cas, la maîtrise d'un système de visualisation extensible demande un temps d'apprentissage non négligeable pour un utilisateur-développeur. Bien évidemment, l'apprentissage de certains systèmes de VS est facile s'il ne s'agit que de les utiliser sans y apporter d'extensions, mais il faut plus de temps pour apprendre à y ajouter des fonctionnalités.

3.2 Aspect collaboratif

L'aspect collaboratif est divisé en deux parties, l'une traitant l'efficacité et l'autre le confort d'utilisation.

3.2.1 Efficacité et temps d'attente

Besoins et objectifs

En visualisation, les informations peuvent être très volumineuses, le temps de calcul pour accéder aux données et produire leur représentation graphique peut être grand. En plus, les performances du réseau peuvent être très variables selon les participants. Tout cela implique qu'il faut transmettre les données de façon à ce que les participants attendent le moins possible avant que le résultat d'une action soit visible. Autrement dit, il faut minimiser le **temps d'attente**, à savoir le délai entre le moment où un participant décide de créer un objet graphique et son apparition sur l'écran des différents participants.

Pour minimiser ce temps d'attente, il convient avant tout d'optimiser les transmissions de données volumineuses : dans un tel cas, le temps de transmission peut dépasser le temps nécessaire au calcul d'un objet graphique.

Problèmes et approches de solutions

Plusieurs problèmes se posent, tel que le choix des **données de visualisation** à transmettre ; à titre d'exemple, on peut transmettre les données brutes, les données traitées ou les objets graphiques. Il faut exploiter le mieux possible toutes les informations à disposition et essayer d'en dégager une méthode pour choisir automatiquement le meilleur type de données de visualisation à transmettre. Il faut choisir le mécanisme de transmission afin que ces dernières se fassent le plus efficacement possible.

Afin de représenter les différentes configurations, nous utiliserons un modèle. Il permet de décrire les liens possibles entre les différentes entités du système et les données de visualisation qu'elles traitent et celles qu'elles produisent, mais aussi d'établir les différentes possibilités de transmissions. Une comparaison entre les temps d'attente dans ces différentes configurations peut ensuite être faite et une heuristique pourra en être déduite.

3.2.2 Confort d'utilisation

Besoins et objectifs

Afin d'avoir un système ergonomique, la partie collaborative doit être simple à utiliser et le temps d'apprentissage du système collaboratif doit être faible pour celui qui connaît déjà un système non collaboratif similaire. Comme indiqué à la section 1.3.1, l'interaction doit être **synchrone** et le système doit être optimisé pour moins de cinq participants, mais il devra aussi fonctionner pour un nombre plus élevé de participants. Le système doit aussi être

flexible, il doit par exemple pouvoir être utilisé tant dans le cadre de l'enseignement que pour la recherche.

Les participants doivent pouvoir faire fonctionner le système sur leur propre ordinateur. Le système doit donc être portable sur plusieurs systèmes d'exploitation ; il doit en outre s'adapter à des matériels et à des accès au réseau présentant des performances différentes. Un mécanisme de contrôle d'accès doit permettre aux participants de limiter l'accès aux ressources partagées du système.

Problèmes et approches de solutions

Afin de pouvoir contrôler l'accès aux ressources, des modes d'interaction peuvent être mis à disposition, par exemple un mode « présentation », où seul un participant peut créer et modifier les informations, les autres participants pouvant uniquement les voir ou bien un mode « libre », où tous les participants peuvent interagir librement. Le choix d'un de ces modes dépend évidemment des besoins des participants.

Le système doit être confortable à utiliser ; pour cela les participants doivent pouvoir accéder facilement aux informations liées à l'*awareness* ainsi qu'au *grounding*. Afin de faciliter l'apprentissage, l'établissement de la communication initiale doit être simple et intuitive.

Pour démontrer que le système est simple à utiliser, il faut des *feedbacks* des participants. Par exemple, il faut recueillir les appréciations des participants sur les points positifs ou négatifs du système.

3.3 Fonctionnalités souhaitées pour un système de VSC

Voici une liste des différents problèmes qui se posent pour un système de VSC, ainsi que des approches de solution.

Système de visualisation de base

Idéalement, il faudrait que chaque participant puisse travailler et collaborer en utilisant son système de visualisation préféré. Un tel but est pratiquement impossible à atteindre ; en effet il n'y a pas de fonctionnalités standards ou d'architecture commune aux différents systèmes existants. De plus, chaque système gère de sa propre manière l'accès aux données brutes et leur traitement ; il en résulte qu'il n'est pas possible d'échanger des données entre les systèmes aux différentes étapes du traitement des données.

Ce problème ne sera pas résolu ici, nous nous contenterons donc de l'hypothèse que tous les participants travaillent sur le même système.

Portabilité sur différents systèmes d'exploitation – Support de matériels différents

Avec l'ajout à un système de visualisation de fonctionnalités permettant la collaboration, il est tout aussi naturel que chaque participant travaille sur son propre ordinateur. En effet, un des avantages de la collaboration est de ne pas avoir à se déplacer, que se soit dans un autre pays ou dans le bâtiment d'à côté pour pouvoir profiter d'un ordinateur plus puissant que le sien. Il y a donc un réel besoin de portabilité que ce soit au niveau du système d'exploitation ou du matériel.

Contrairement au problème précédent de la compatibilité entre des systèmes de visualisation différents, ce problème de portabilité peut être résolu.

La technique choisie doit pouvoir tirer profit du matériel disponible, par exemple pouvoir utiliser l'accélération matérielle de la carte graphique. Une solution possible consiste à utiliser Java3D [93]. Cette solution offre un bon compromis entre les performances pour la visualisation en 3D et la portabilité. On peut ainsi lancer le système depuis une page web.

Portabilité sur différents réseaux - Support de réseaux rapides/lents

Comme chaque participant travaille sur son propre ordinateur, chacun possède sa propre connexion au réseau ; les vitesses de transmission entre les différents participants peuvent donc passablement différer. Le système doit donc pouvoir travailler tant avec des connexions rapides qu'avec des connexions lentes.

Si un participant est relié aux autres par une connexion lente (par exemple une connexion WIFI), il doit pouvoir participer à une session collaborative si certaines conditions sont remplies ; son ordinateur doit par exemple posséder dès le début de la session une copie des données brutes. En effet, la transmission des informations de contrôle nécessaires au maintien de la cohérence de l'espace de travail ne requiert pas une connexion rapide. Par contre, comme le participant en question ne dispose pas de suffisamment de bande passante, la transmission des données de visualisation demanderait beaucoup de temps.

Pour la portabilité au niveau du réseau, une solution technique possible est d'utiliser un *middleware* tel que RMI [96] ou CORBA [76]. Un problème non résolu se pose au niveau de la sécurité, par exemple en raison des systèmes de pare-feu, qui peuvent interdire les échanges entre les participants.

Flexibilité des rôles

Chaque participant doit pouvoir configurer son ordinateur comme il le souhaite. Comme la visualisation demande parfois beaucoup de ressources, il est judicieux de pouvoir, dans le cadre de la session collaborative, dédier certains ordinateurs à un travail donné. Par exemple, on peut dédier un ordinateur dont la puissance de calcul dépasse celle des autres au rôle de **serveur de données** ; c'est-à-dire qu'il produit des données traitées à partir des données brutes. Cela permet de décharger les autres ordinateurs, qui ne s'occupent plus que de construire des objets graphiques à partir de ces données traitées. Dans une autre situation, on peut avoir un **serveur d'objet graphique** ; ce serveur s'occupe de créer les objets graphiques à partir des données traitées. Dans ce cas, l'affichage final se fait sur un **client léger** dont la tâche principale consiste à afficher à l'écran les objets graphiques. Ce client peut être un ordinateur peu puissant ou ne possédant pas assez d'espace pour stocker les données. Idéalement, les participants doivent pouvoir fixer dynamiquement le rôle de leur ordinateur.

Simplicité de la connexion à la session

Les connexions au système doivent être simples à mettre en œuvre :

- Connexion initiale : La connexion à une session collaborative doit demander le moins d'effort que possible. Un administrateur, qui est le premier participant à créer la session, doit pouvoir restreindre l'accès et effectuer la configuration initiale.

Idéalement, les participants doivent pouvoir se connecter simplement à partir d'une page web.

- Rejoindre/quitter la session : Tous les participants ne travaillent pas exactement en même temps. Un participant doit donc pouvoir rejoindre ou quitter une session quand il le désire.

Système intuitif

Un système collaboratif doit ressembler autant que possible à sa version non collaborative. Cela permet à un nouvel utilisateur d'utiliser le système en réduisant la phase d'apprentissage du système collaboratif. Il est évident que l'ajout de la collaboration ne devra pas briser les fonctionnalités du système non collaboratif.

Contrôle d'accès

Le contrôle d'accès (*floor control*) permet de limiter l'accès aux ressources partagées du système par les participants.

Le mode de travail dépend des relations entre les participants. Il s'agit d'identifier ces différents modes. Chaque situation particulière appelle **un** certain mode de travail parmi ceux qui ont été prévus. Par exemple, on peut considérer les deux situations suivantes : la situation où trois scientifiques ayant des connaissances diverses désirent collaborer et la situation d'une classe virtuelle où un maitre veut montrer un phénomène particulier à ses étudiants. Ces deux situations impliquent des droits d'accès différents ; par exemple dans la première situation les participants devront pouvoir interagir librement alors que dans la seconde les étudiants ne doivent pas pouvoir modifier ce que le maître a créé.

Il en résulte deux modes principaux, un mode « **libre** » et un mode « **présentation** ». Ce sont ces deux modes qui seront les plus utilisés parmi les modes mis à disposition. Pour plus de flexibilité, on peut adapter l'un ou l'autre de ces modes ; dans le mode présentation, on peut par exemple prévoir deux maîtres ou autoriser temporairement un étudiant à créer des objets graphiques. Cela peut être réalisé en utilisant des permissions ; dans ce cas les modes sont définis par un ensemble de permissions et les variantes sont réalisées en modifiant les permissions originales d'un des modes.

La gestion des permissions constitue une solution technique simple et efficace. Ces permissions sont affectées aux différentes entités qui doivent être protégées. Une permission donne à un ensemble de participants le droit d'effectuer une action, par exemple de créer un *artefact*. Pour plus de détails, se référer à la section 6.3.4.

Protection des données de visualisation

Comme la production des données brutes demande un effort important et qu'elles sont parfois confidentielles, les participants souhaitent parfois les protéger. Il convient donc de prévoir un mécanisme de protection simple à utiliser.

Les participants travaillent dans un **monde 3D partagé**, où chacun peut créer des objets graphiques et interagir avec eux. Les participants doivent avoir la possibilité de protéger leurs données ainsi que les objets graphiques de leur monde 3D. Un **monde partagé** permet aux participants de collaborer. En même temps, un participant doit pouvoir travailler seul dans un **monde privé**, par exemple pour faire des essais sans salir le monde partagé. Les objets

graphiques créés dans un monde privé doivent ensuite pouvoir aisément être exportés dans le monde partagé afin que tous puissent y accéder.

La différence entre le monde partagé et le monde privé réside uniquement dans les permissions dont dispose un participant : par défaut tous les participants peuvent accéder au monde partagé, alors que seul le participant concerné peut accéder à son monde privé. Si le système est conçu de manière flexible, on peut imaginer donner accès à son monde privé à certains participants, par exemple pour demander un avis à un collègue ou pour former des sous-groupes de travail.

Comme décrit précédemment, l'utilisation des permissions constitue aussi une solution technique à ce problème.

Système efficace

L'ajout de la collaboration à un système non collaboratif doit avoir le moins d'impact possible sur les performances. Idéalement, un système collaboratif utilisé par une seule personne doit avoir les mêmes performances que le système non collaboratif correspondant. En utilisation collaborative, le temps de transmission de l'information constitue la principale entrave à un fonctionnement satisfaisant du système. Donc dans la plupart des cas, le fait de collaborer augmentera le temps d'attente. Le temps d'attente devrait augmenter le moins possible. Il faut que les différents transmissions d'information entre participants, qu'il s'agisse de données ou d'objets graphiques, se fassent le plus efficacement possible. C'est un de nos objectifs ; la solution proposée est traitée en détail dans le chapitre 4.

Système robuste

Le système doit pouvoir continuer à fonctionner en cas de départ soudain d'un participant ou d'une panne d'un ordinateur. Il faut veiller à garder la cohérence entre tous les participants.

Système *scalable*

Lorsque le système est utilisé en mode présentation, l'ajout de nouveaux élèves doit avoir le moins d'impact possible sur les performances, en particulier sur le temps d'attente. Par contre, en mode libre, cette exigence est moins importante. En effet dans ce cas, le problème le plus important est le confort d'utilisation cognitif ; il s'agit avant tout d'assurer un support efficace de l'*awareness*.

Support de l'*awareness* et du *grounding*

Afin d'améliorer l'*awareness*, un participant doit savoir qui se trouve dans le monde 3D, où les autres participants sont situés et ce qu'ils font. Cela peut être réalisé par exemple par une représentation virtuelle des participants dans l'espace de travail ou par des messages sous forme de textes donnant des informations sur les activités de chaque participant.

L'utilisation d'un média de communication, comme un *chat,* un téléphone ou une visioconférence permet d'améliorer le *grounding*. Souvent les participants sont habitués à leur propre système de communication, il faut donc être flexible quant au choix de ce dernier. Au minimum, un *chat* doit être réalisé dans le système. Le *chat* a l'avantage d'être disponible pour tous les participants et d'utiliser peu de bande passante. Un système de visioconférence alourdit l'interface graphique et requiert de la bande passante.

Le support de l'*awareness* et du *grounding* est traité plus en détail dans le chapitre 5.

3.4 Conclusion

Parmi les différents systèmes existants, tels que AVS et ses extensions collaboratives [2], CSpray [79], VisAD [46] ou gViz [8], la plupart sont conçus soit pour optimiser les transmissions de données (efficacité technique), soit pour améliorer la satisfaction perçue par les participants, mais la majorité d'entre eux ne prêtent pas assez attention aux deux aspects simultanément [9].

Pour cette raison, nous cherchons à obtenir une excellente satisfaction cognitive tout en minimisant le temps d'attente pour les participants. Cela nous conduit à présenter nos travaux en deux parties : le chapitre 4 discute de **l'efficacité technique** et le chapitre 5 de **la satisfaction cognitive**.

Chapitre 4 Efficacité technique

Un de nos objectifs consiste à améliorer l'efficacité technique. Ce chapitre présente les différents moyens mis en œuvre pour minimiser le temps d'attente. Nous présentons d'abord l'**architecture générale**, puis nous indiquons les différentes configurations possibles pour transmettre l'information entre les participants. Finalement, nous discutons les résultats des tests effectués.

4.1 Architecture

Afin de présenter nos solutions aux problèmes techniques qui se posent, nous nous appuyons sur un modèle qui sert de base à notre architecture. Ces problèmes concernent principalement la réduction du temps d'attente tout en conservant l'intuitivité du système.

4.1.1 Modèle utilisé

La figure 4.1 décrit notre architecture. Dans celle-ci, ainsi que dans toutes les figures de cette section, un rectangle représente de l'information, une ellipse un processus de traitement et les flèches représentent les interactions entre ces entités. Le modèle utilisé pour décrire notre architecture est basé sur les modèles collaboratifs de Wood [108] et de Duce [26]. Les notations et termes décrits dans leurs modèles ne sont pas tous repris.

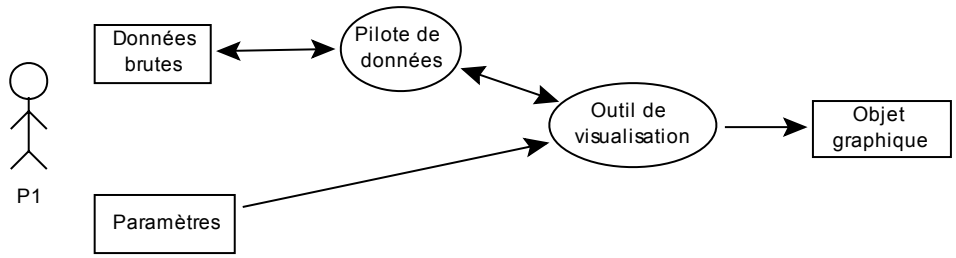

Figure 4.1 : Architecture de notre système

Dans la figure 4.1, le participant P1 fournit les paramètres nécessaires à la création d'un objet graphique. Ces paramètres sont constitués de l'information nécessaire à la création d'un objet graphique, donc la désignation de l'**outil de visualisation** et ses arguments. A titre d'exemple, considérons l'outil de visualisation Isosurface : ses arguments sont le champ scalaire considéré, la région d'intérêt, l'*isovalue* et la *colormap* (palette de couleurs).

L'outil de visualisation interroge les données brutes à travers le pilote de données. Le pilote de données s'occupe donc de transformer les données brutes dans un format que tous les outils de visualisation peuvent interpréter. Une telle architecture implique que les données brutes sont indissociables du pilote de données. Cela rend les outils de visualisation indépendants des types de données brutes, ce qui rend le système extensible au prix de l'écriture d'un pilote de données pour chaque nouveau type de données brutes.

La transformation *Filter* du modèle du flux de données [42], décrit à la section 2.1.1, peut être comparé au pilote de données, les transformations *Map* et *Render* à l'outil de visualisation, même si une partie du travail de la transformation *Render* est effectué par le système graphique.

Comparé au modèle de Wood, notre modèle est donc plus simple et mieux adapté à nos besoins : nous pouvons représenter directement les principaux flux ainsi que les différentes transmissions d'informations possibles. Deux entités peuvent être liées dans les deux directions, ce qui permet de représenter de multiples interactions entre l'outil et le pilote de données pour construire l'objet graphique.

La figure 4.1 se rapporte au cas d'un seul participant. Nous allons voir les différentes difficultés qui se posent lorsque plusieurs participants interviennent.

4.1.2 Informations à transmettre

La transmission de l'information entre les ordinateurs des participants peut se faire par les paramètres, les informations fournies par le pilote de données ou le résultat des calculs effectués par l'outil de visualisation (c'est-à-dire l'objet graphique). Pour des raisons de protection des données et d'extensibilité (comme décrit à la section 3.1), les données brutes ne sont jamais directement transmises par le système. Seules sont transmises (par l'intermédiaire des pilotes de données) les données traitées. Cela permet, par exemple, de mettre en place des permissions d'accès et de ne pas avoir à écrire un mécanisme de transmission pour chacun des formats de données brutes. Toutefois cela n'implique pas qu'un seul participant possède les données brutes. En effet, les données brutes peuvent être transmises par des moyens annexes avant le lancement du système et être présentes chez les ordinateurs de plusieurs participants.

4.1.3 Les différentes configurations possibles

Nous distinguons cinq **configurations** possibles pour transmettre l'information entre les participants :

Configuration a) Transmission des paramètres, les données brutes étant disponibles localement

La figure 4.2 représente un scénario de création d'un objet graphique sur l'écran de tous les participants, les mêmes données brutes étant disponibles localement chez chacun. Chaque ordinateur d'un participant construit localement un objet graphique grâce à ses données brutes et aux paramètres fournis par le participant ayant fixé les paramètres. Les objets graphiques résultants seront identiques.

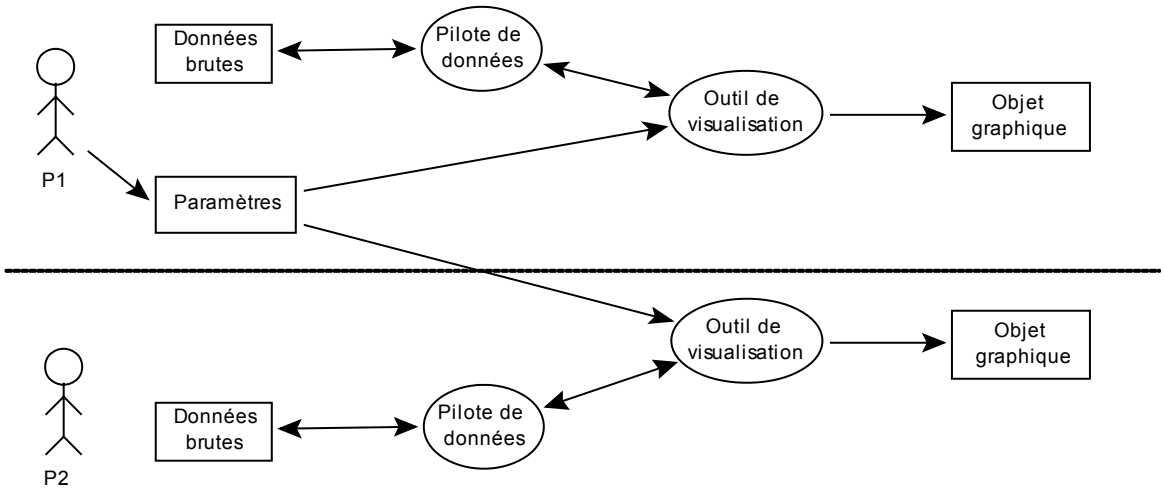

Figure 4.2 : Création d'un objet graphique, les données brutes étant disponibles localement chez chacun

Dans cette configuration, les données brutes sont présentes chez chacun dès le début de la session de visualisation ; elles ont donc été transmises préalablement. Cette configuration est clairement celle qui charge le moins le réseau. En effet, on transmet uniquement les paramètres requis pour construire l'objet graphique, qui occupent peu de volume. Il en résulte que, dans la majorité des situations, c'est la configuration où le temps d'attente est le plus court. Malgré tout, il existe une situation (voir configuration e) où ce n'est pas le cas.

Cependant les données brutes ne sont souvent pas disponibles chez tous les participants. Dans les domaines d'application considérés, la production des données brutes demande en effet un grand travail ; en outre des considérations de protection des données, comme dans le domaine médical, peuvent s'opposer à leur diffusion.

Nous allons donc devoir examiner les configurations qui se rapportent à la situation où les données brutes ne sont pas disponibles chez chacun.

Un ordinateur possédant les données brutes localement est appelé **serveur de données** alors qu'un ordinateur construisant un objet graphique localement est appelé **serveur de calcul**.

Configuration b) Transmission des paramètres et des données

La figure 4.3 représente la configuration dans laquelle les données brutes n'existent que chez un seul ordinateur, le **serveur de données**. Chaque ordinateur va construire son objet graphique localement en demandant les données traitées nécessaires au serveur de données.

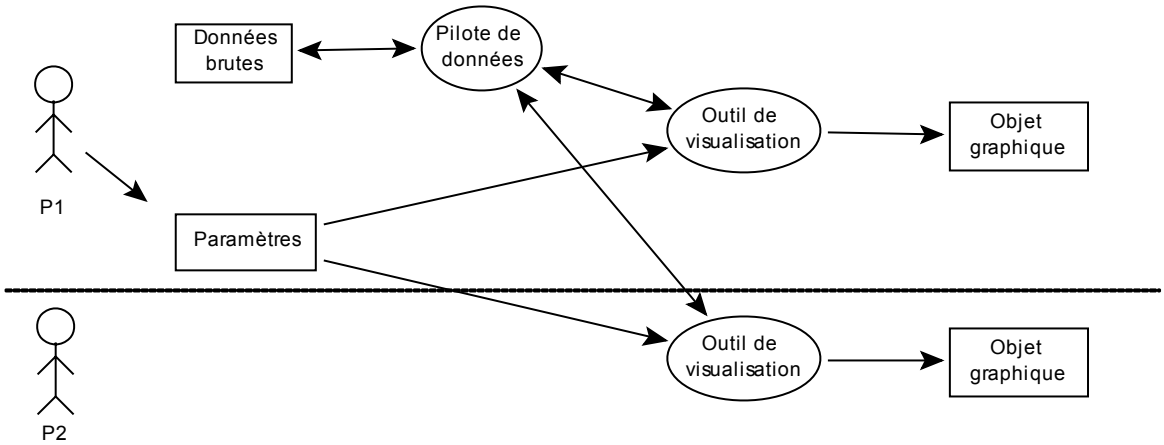

Figure 4.3 : Création d'un objet graphique, les données brutes étant disponibles chez un seul participant qui les met à disposition par son pilote de données

Seuls les paramètres et les données traitées transitent donc par le réseau, seules ces dernières étant peut-être volumineuses. Par rapport à la configuration a), cette situation conduit à une charge de calcul supplémentaire pour le serveur de données, dont le pilote de données, en plus des données demandées par **son** outil de visualisation, doit produire les données traitées demandées par les autres participants. Les outils de visualisation de chacun ayant besoin des mêmes données traitées pratiquement simultanément, ce surcroît de travail peut être néanmoins évité par une mémoire-cache réalisée à l'intérieur du pilote de données.

Configuration c) Transmission des paramètres et de l'objet graphique

La figure 4.4 représente la configuration dans laquelle les données brutes n'existent que chez un seul participant, qui produit l'objet graphique pour les autres et le leur envoie.

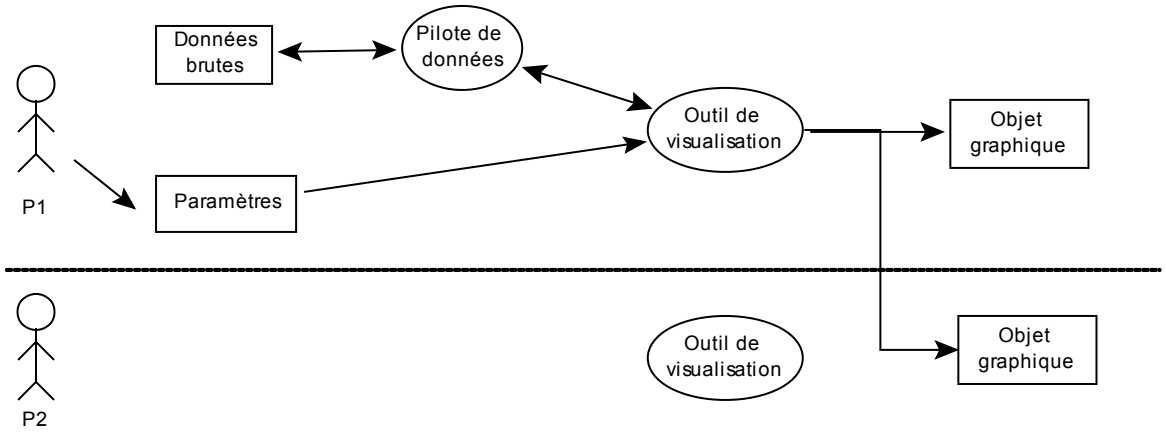

Figure 4.4 : Création d'un objet graphique, les données brutes étant disponibles chez un seul participant, l'objet graphique est partagé

Ainsi seul l'objet graphique transite par le réseau ; son volume peut être grand. Par rapport à la configuration a), l'ordinateur qui le produit ne doit effectuer aucun calcul supplémentaire, si ce n'est celui qui est lié à la diffusion de l'objet graphique.

Configuration d) Transmission des paramètres, des données et de l'objet graphique

La figure 4.5 représente la configuration dans laquelle les données brutes n'existent que chez un **serveur de données**, l'objet graphique étant construit par un autre ordinateur, le **serveur de calcul**, qui diffuse ensuite ce résultat à chacun.

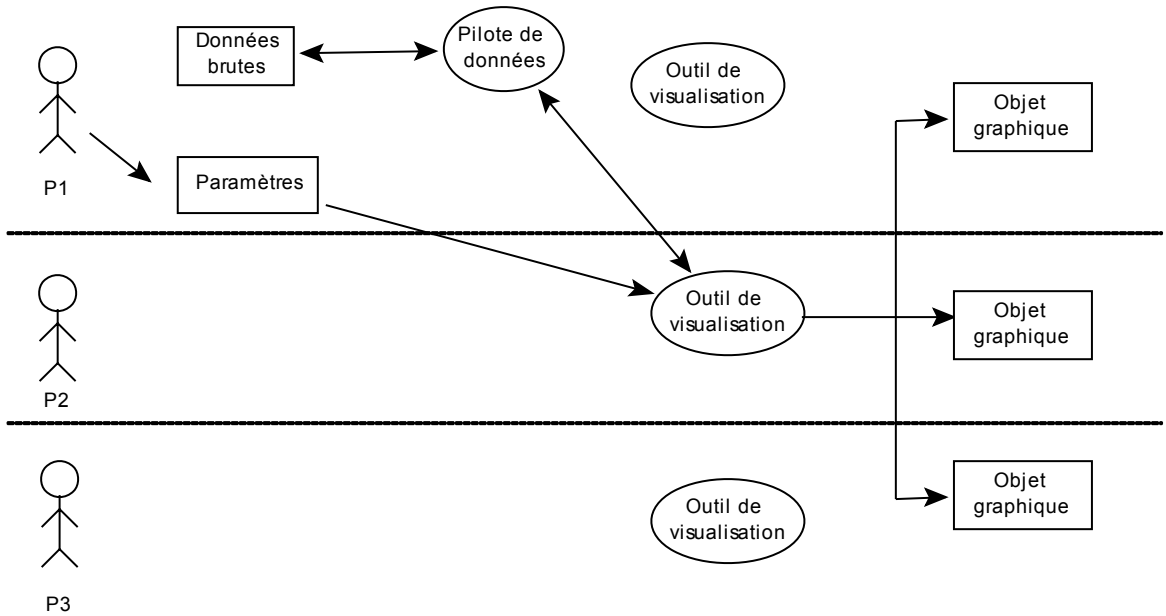

Figure 4.5 : Création d'un objet graphique, les données brutes étant disponibles chez un seul participant, l'objet graphique est construit par un ordinateur ne possédant pas les données brutes

Pour les différents participants, cette configuration est plus efficace que la configuration c) uniquement si la construction de l'objet graphique est plus rapide chez le serveur de calcul que chez le serveur de données et si ce gain n'est pas anéanti par le temps supplémentaire requis par les transmissions.

Configuration e) Transmission des paramètres et de l'objet graphique avec des données brutes disponibles localement

La figure 4.6 représente la configuration dans laquelle, même si les données brutes existent chez chacun, un participant produit pour les autres l'objet graphique et leur envoie cet objet graphique.

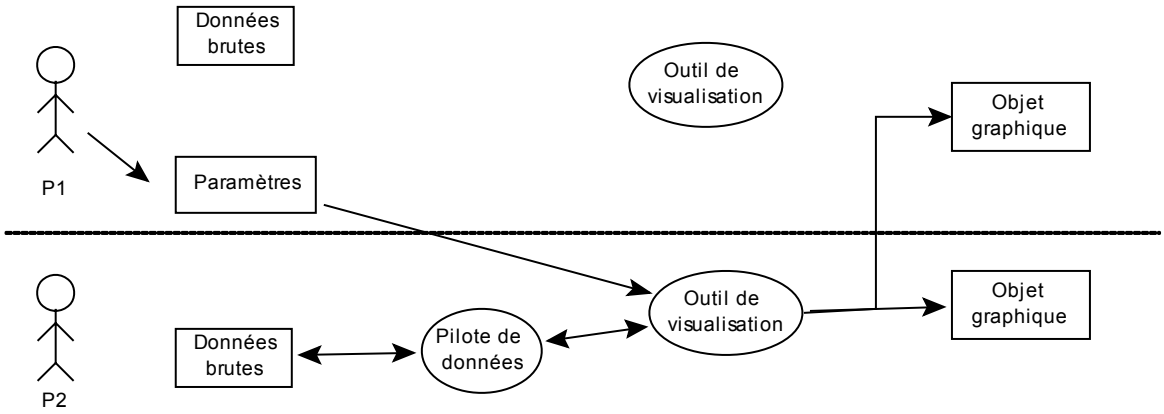

Figure 4.6 : Création d'un objet graphique, les données brutes étant disponibles localement chez chacun et construction de l'objet graphique chez un participant

Comme énoncé auparavant, la configuration a) est en principe celle où le temps d'attente est le plus court. Néanmoins la configuration e) peut se révéler plus efficace que la configuration a) si un ordinateur est nettement plus performant que les autres et si le temps de transmission de l'objet graphique n'est pas trop grand. Dans ce cas, le participant possédant l'ordinateur le plus performant va construire l'objet graphique et ensuite le transmettre aux autres participants.

4.2 Discussions des configurations

Au regard des défis énoncés dans la section 3.3, on peut maintenant présenter les avantages et les désavantages de ces configurations, ainsi que les possibilités qu'offrent ces configurations et notre modèle :

- **Portabilité sur différents réseaux - Support de réseaux rapides/lents** : Notre modèle permet de représenter simplement les différents types de transmissions possibles. Si un participant possède une connexion lente au réseau et qu'il dispose localement des données brutes, il pourra participer à la session collaborative, par exemple en utilisant la configuration a). En effet, la transmission des informations de contrôle nécessaires au maintien de la cohérence de l'espace de travail requiert un petit volume. Si un participant dispose d'un réseau rapide, il a le choix entre plusieurs configurations afin de transmettre efficacement les informations. Nous avons ainsi un système flexible et pouvant s'adapter à différents type de réseau.

- **Flexibilité des rôles** : Les différents types de rôles sont parfaitement identifiables dans nos configurations : le rôle **serveur de données** correspond au participant envoyant les données de la configuration b), le **rôle serveur d'objets graphiques** au participant envoyant les données de la configuration c) et le rôle du **client léger** correspond au participant recevant ces informations. En fonction de ses besoins et des caractéristiques de son matériel, un participant peut donc configurer son ordinateur

comme il le souhaite. Par exemple, un ordinateur dont la puissance de calcul dépasse celle des autres, peut facilement tenir le rôle de serveur de données.

- **Système intuitif** : Vu que le système n'est pas basé sur un MVE, les paramètres d'un outil de visualisation sont tous regroupés, il est donc inutile d'avoir plusieurs instances (comme les sept transformations dans l'exemple de l'isosurface de la section 2.1.2), ce qui constitue une simplification par rapport au modèle de Wood. Ce modèle permet d'identifier le participant qui contrôle une transformation d'une manière explicite ainsi que d'indiquer où sont situés les informations de visualisation. L'intuitivité de la version mono-utilisateur est conservée ; en effet la collaboration requiert peu d'ajouts.

4.3 Heuristiques

Nous voulons minimiser le **temps d'attente** qui s'écoule entre le moment où un participant donne l'ordre de création d'un objet graphique et le moment où il apparaît sur tous les écrans. Pour cela, il faut tenir compte du temps de transmission et du temps de calcul. Malheureusement, une diminution de l'un de ces deux facteurs entraîne le plus souvent une augmentation de l'autre. Le principal problème provient du fait qu'il est ardu de prévoir à l'avance (c'est-à-dire avant que l'objet graphique soit créé) le temps de transmission ainsi que le temps de calcul nécessaires.

Pour rappel, l'**outil de visualisation** génère un objet graphique à partir des paramètres et des données brutes (section 4.1) ; c'est le **système** qui gère ces outils de visualisation. Un utilisateur-développeur peut rajouter de nouveaux outils de visualisation dans le système.

Afin d'obtenir un système simple d'utilisation, les participants ne doivent pas devoir s'occuper de choisir la configuration : c'est à l'outil de visualisation ou au système que doit être dévolue cette tâche. Ce choix de la configuration se fait lors de chaque création d'objets graphiques.

Il s'agit donc d'élaborer une **heuristique** qui choisit entre les différentes configurations. Une telle heuristique doit tenir compte de plusieurs facteurs, tels que l'outil de visualisation utilisé, la valeur des arguments passés à ce dernier, la région d'intérêt (en fait le nombre de points pour lesquels il faut demander des données traitées), les caractéristiques des connexions réseau entre les participants, la puissance de calcul des ordinateurs des participants et les caractéristiques des données brutes utilisées [16].

Le facteur qui joue le rôle le plus important est l'outil de visualisation. En fonction de celui-ci, le choix de la configuration devra être fait en tenant compte des autres facteurs.

Cela conduit à proposer deux types d'approches :

1) implémentation de l'heuristique dans l'outil de visualisation,

2) implémentation de l'heuristique dans le système.

Nous allons décrire ces deux approches plus en détail et ensuite les comparer.

4.3.1 Heuristique intégrée à l'outil de visualisation

Lorsque l'heuristique est implémentée dans l'outil de visualisation, un utilisateur-développeur qui crée un nouvel outil de visualisation doit aussi réaliser une heuristique adéquate. Cela

n'empêche pas un système de mettre à disposition une heuristique par défaut, qui sera certainement très grossière parce qu'elle ne dispose que de peu d'informations utiles.

A titre d'illustration, nous présentons quelques outils de visualisation courants, ainsi que des heuristiques envisageables pour chacun d'entre eux. Plus précisément, nous donnons pour chaque outil de visualisation une courte description et nous comparons le volume de données traitées et le volume de l'objet graphique produits. Puis nous discutons l'heuristique fixée par l'outil de visualisation. Les outils se basant sur des algorithmes de visualisation similaires ont été regroupés.

- **Plan de coupe**. Cet outil représente par des couleurs les valeurs d'un champ scalaire pour les points d'un plan de l'espace choisi par l'utilisateur. Le volume des données transmises pour les configurations b) et c) est plus ou moins le même. Cela est dû au fait que pour les *m*n* points d'un plan de coupe, on demandera *m*n* valeurs au pilote de données et que le résultat graphique sera composé de *m*n* valeurs. Le volume du résultat graphique varie bien évidemment en fonction de la conception de l'algorithme. Nous considérons le cas où une texture est utilisée pour stocker ces *m*n* valeurs. Notre heuristique choisit donc toujours la configuration c).

- ***Arrows*, Sonde**. Ces outils représentent les valeurs d'un champ vectoriel pour les points d'un domaine de l'espace (souvent un plan) par des flèches, respectivement par des sondes [102]. Comparé à l'outil *Arrows*, l'outil Sonde permet de représenter plus d'informations, par exemple la vitesse, l'accélération et la torsion d'un liquide en mouvement. Pour construire leurs représentations, *Arrows* et Sonde demandent au pilote de données les composantes du champ aux endroits en question. L'objet graphique résultant est composé d'un grand nombre de triangles alors que le pilote de données ne doit fournir qu'un petit nombre de valeurs du champ vectoriel. C'est pourquoi, notre heuristique choisit toujours la configuration b). La figure 4.8 représente une Sonde et la figure 4.9 montre un exemple d'utilisation de l'outil *Arrows*.

- ***Streamline*, *Streakline*, *Pathline*** [70], **Ruban** [75]**.** Ces outils représentent un champ vectoriel pour les points d'un domaine de l'espace. Le volume des données traitées nécessaires dans les configurations b) et c) est plus ou moins le même, néanmoins pour calculer une *streamline* (les trois autres outils sont dans la même situation), un point après l'autre doit être demandé. En effet, pour une *streamline* de longueur *n* composée de p_n points, on calcule la position p_1 en demandant une donnée, puis la position p_2 en demandant une nouvelle fois une donnée, etc. Cela conduit à demander *n* fois une donnée au pilote de données. Il est donc plus judicieux de la calculer en une fois et de transmettre le résultat. Notre heuristique choisit donc toujours la configuration c). La figure 4.7 montre un exemple d'utilisation des outils *Streamline* et Ruban.

- **Isosurface, Isoligne.** Ces outils représentent le lieu géométrique des points d'une partie de l'espace, respectivement d'un plan, pour lesquels un certain champ scalaire a une valeur donnée. Lors de la construction d'une isosurface ou d'une isoligne, le volume des données brutes est connu, par contre le volume de données graphiques est impossible à connaître à l'avance. Ce n'est qu'après l'exécution de l'algorithme calculant l'isosurface ou l'isoligne (dans notre cas l'algorithme « *Marching cube* » [60] ou une version améliorée [74]) que le volume des données graphiques est

connu. Notre heuristique choisit toujours la configuration b) mais ce n'est évidemment pas toujours le meilleur choix.

- ***Textured vector field***. Cet outil représente les valeurs d'un champ vectoriel pour les points d'un domaine de l'espace par une texture [12]. Le volume des données ainsi que la taille de la texture résultant sont connus à l'avance. On peut donc prédire quelle configuration conduit au plus petit volume de données à transmettre. Toutefois, cet outil requiert passablement de temps de calcul, il faut donc en tenir compte et privilégier la configuration b) uniquement quand celle-ci conduit à un volume à transmettre nettement moindre que la configuration c). L'heuristique doit donc estimer à l'avance le volume de données traitées et le volume de données graphiques produites ; si ce dernier est plus grand que le volume de données traitées, elle choisit la configuration b), sinon la configuration c). La figure 4.9 montre un exemple d'utilisation de cet outil.

- **Objet géométrique**. Cet outil représente un objet graphique décrit avec des données brutes. Par exemple, dans le domaine de l'océanographie, cet outil est utilisé pour représenter la topographie de la mer. Le choix de la configuration dépend des données traitées et de sa représentation graphique. Vu que le volume de la représentation graphique est inconnu, notre heuristique prend toujours la configuration b), mais ce choix n'est évidemment pas toujours le meilleur. Par exemple, si les données traitées sont plus volumineuses que l'objet graphique, le choix de la configuration c) serait plus judicieux.

Si un outil de visualisation ne fournit pas d'heuristique, le système choisit par défaut toujours la configuration b). Cette heuristique est très simple.

Le tableau 4.1 résume nos heuristiques.

Tableau 4.1 : Outils de visualisation et heuristique

Outil de visualisation	Choix de l'heuristique
Plan de coupe	b)
Arrow, Sonde	b)
Streamline, Streakline, Pathline, Rubans	c)
Isosurface, Isoligne	Volume des données inconnues→ b)
Textured vector field	Volume des données connues → choix dynamique
Objet géométrique	Volume des données inconnues→ b)
Par défaut	b)

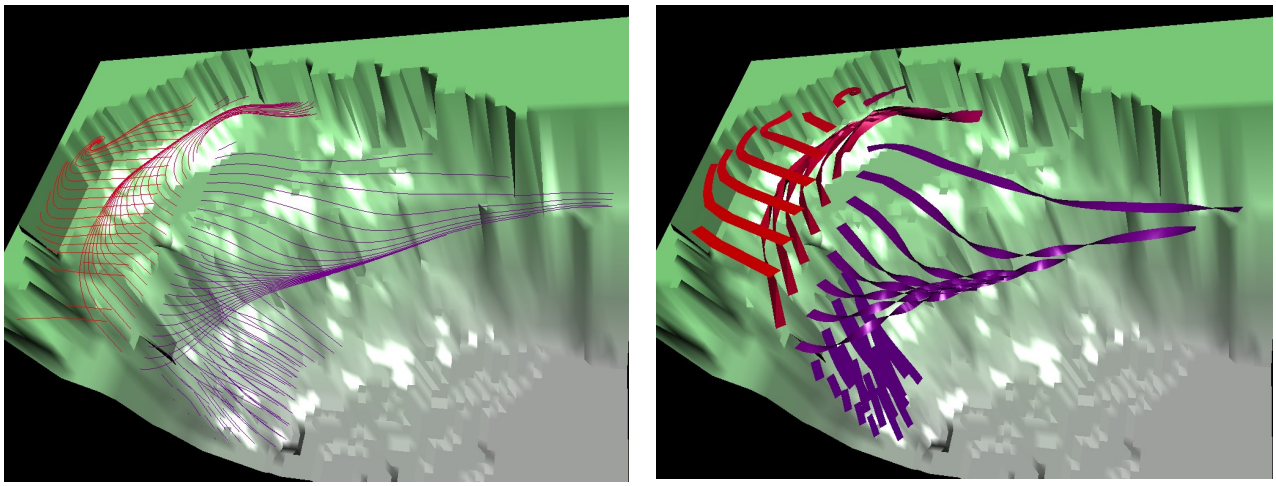

Figure 4.7 : Représentation des courants dans le golfe du Lion par des *Streamlines* (gauche) et par des Rubans (droite)

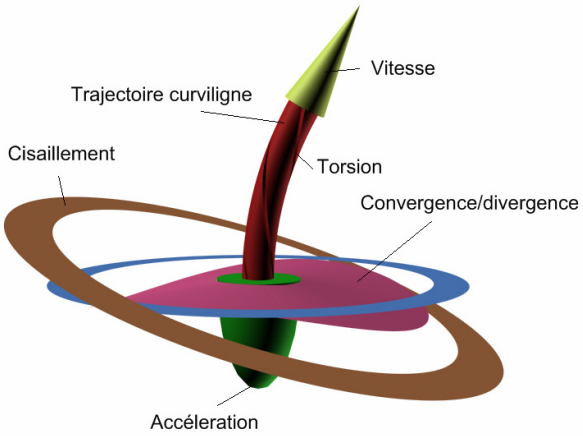

Figure 4.8 : Sonde

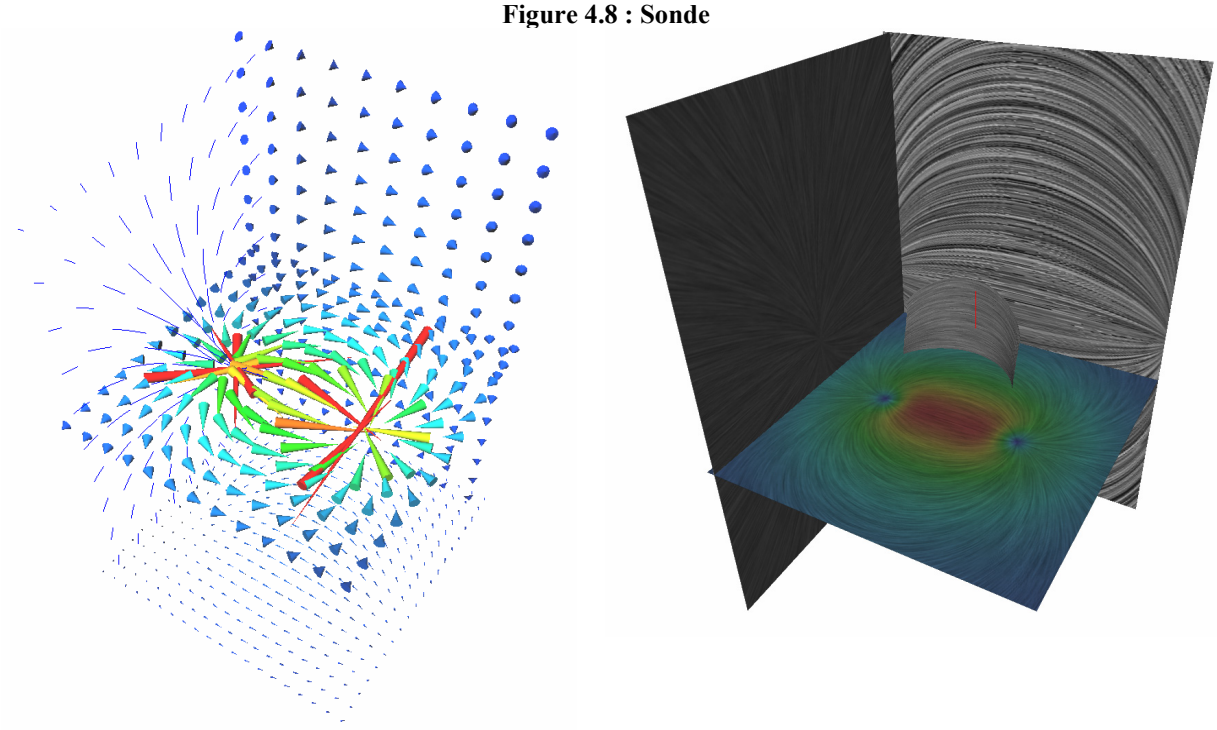

Figure 4.9 : Représentation du champ magnétique d'un dipôle par des *Arrows* (gauche) et par des textures LIC (droite)

4.3.2 Heuristique intégrée dans le système

Dans cette seconde approche, ce n'est pas l'outil de visualisation qui choisit la configuration mais notre système. L'heuristique se base sur les facteurs suivants, décrits par la figure 4.10, qui sont classées en deux catégories :

- Les facteurs dépendant de l'outil de visualisation. En fonction des arguments fournis par le participant, un outil de visualisation doit pouvoir évaluer les facteurs suivants :
    - le volume des données traitées ainsi que le nombre d'envois nécessaires à sa transmission,
    - le volume de l'objet graphique,
    - le temps requis pour effectuer les calculs produisant l'objet graphique.
- Les facteurs dépendant du matériel :
    - la performance du réseau à disposition,
    - la puissance de calcul des ordinateurs à disposition.

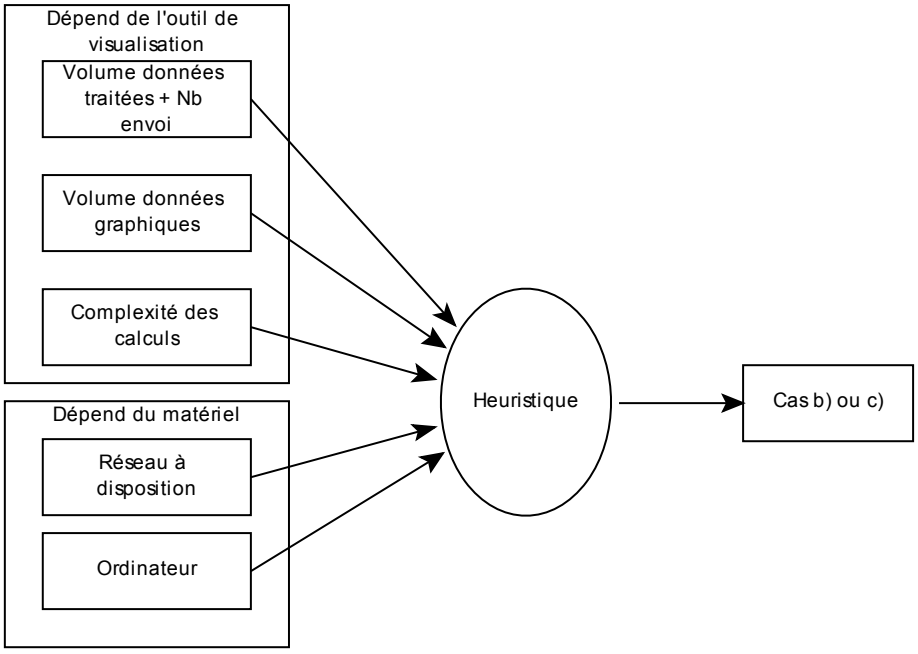

Figure 4.10 : Choix par le système de la configuration en fonction des facteurs

Le matériel ne changeant pas au cours d'une session de visualisation, une seule évaluation des facteurs concernant le matériel est suffisante. Pour obtenir un système efficace, les facteurs dépendant du matériel doivent être connus : chaque ordinateur doit connaître la puissance de calcul des différents ordinateurs et les performances des réseaux qui les relient. L'évaluation de la puissance de calcul peut être produite en fonction du temps mis pour effectuer un même calcul complexe sur les ordinateurs. L'évaluation de la performance réseau peut être produite en fonction du temps mis pour transmettre un certain volume de données entre les ordinateurs.

L'estimation des facteurs dépendants de l'outil de visualisation se fait en fonction des facteurs dépendants du matériel. Certains des facteurs étant impossibles à prédire, l'heuristique doit pouvoir s'accommoder de facteurs déclarés « **inconnus** ».

Lorsqu'un participant désire construire un objet graphique, le système va demander à l'outil de visualisation, en fonction des arguments entrés, quelles seront les valeurs des différents facteurs. Ensuite l'heuristique détermine la configuration à utiliser.

L'algorithme 4.1 décrit notre choix de l'heuristique utilisé.

```
TCalcul         = temps estimé pour les calculs
TDonnée         = temps estimé pour la transmission des données
TObjetGraphique = temps estimé pour la transmission de l'objet
                  graphique
IF (TCalcul disponible)
{
      IF (TDonnée disponible AND TObjetGraphique disponible)
      {
            IF (TDonnée + TCalcul > TObjetGraphique)
                  RETURN C;
            ELSE RETURN B;
      }
      ELSE RETURN RANDOM (B; C); // Au hasard b) ou c) (*)
}
ELSE
{
      IF (TDonnée disponible AND TObjetGraphique disponible)
      {
            IF (TDonnée > TObjetGraphique)
                  RETURN C;
            ELSE RETURN B;
      }
      ELSE RETURN RANDOM (B; C); // Au hasard b) ou c) (*)
}
```

Algorithme 4.1 : Choix de l'heuristique

Comparé à des choix déterministes, les choix aléatoires de l'algorithme 4.1 marqués par un astérisque (*) permettent, dans de nombreuses situations, d'obtenir de meilleures performances.

Soit p_{ik} la probabilité (inconnue) pour que, dans le cas i, la configuration k soit la meilleure. On a bien sûr $p_{i1}+p_{i2}=1$. En choisissant toujours la configuration k, la probabilité de tomber juste est $(\Sigma_i p_{ik})/(\Sigma_i 1)$ qui a une certaine valeur (inconnue) entre 0 et 1. Si on a la malchance de faire le mauvais choix, cette probabilité peut être inférieure à 0.5. En prenant l'algorithme probabiliste, la valeur espérée de la probabilité de tomber juste est de 0.5.

A titre d'exemple, cette situation se produit lorsqu'un participant cherche à obtenir une isosurface et qu'il doit plusieurs fois varier légèrement la valeur de l'*isovalue* avant d'obtenir un résultat qui lui convient. Il est fort probable que la meilleure configuration soit toujours la même. Ce choix conduit à trouver la meilleure configuration dans 50% des cas. Par contre, avec un choix déterministe, la meilleure configuration est trouvée soit dans 100% ou soit dans 0% des cas. Il est évident que de nombreux autres choix de l'heuristique peuvent être faits.

4.3.3 Comparaison entre ces deux approches

L'approche 1 est évidemment la plus simple à réaliser dans le système. En outre, elle permet d'exploiter au mieux les informations disponibles car chaque outil peut définir le choix du cas en tenant compte des arguments pertinents. En revanche, elle reporte le fardeau de l'implémentation de l'heuristique sur l'utilisateur-développeur et le système n'a aucun contrôle sur l'heuristique.

L'approche 2 permet de centraliser l'heuristique dans le système, mais l'utilisateur-développeur doit, pour chaque nouvel outil de visualisation, estimer les facteurs déterminants. Elle peut se révéler moins extensible que l'approche 1 si de nouveaux facteurs particuliers à un outil se révélaient utiles pour déterminer le choix de la configuration.

Dans notre prototype, afin de conserver l'extensibilité du système et pour sa simplicité, nous avons réalisé l'approche 1. La section suivante présente quelques résultats expérimentaux, ainsi que diverses évaluations.

4.4 Mesures et évaluations

4.4.1 Tests

Les mesures ont été faites en utilisant les données brutes « *Lobus* », qui représentent un cerveau de mouche. Dans ces tests, il y a deux participants reliés par une connexion câble à 1000/200Kb/s. La puissance de calcul à disposition est la même pour les deux participants : dans les deux cas, il s'agit de PC Windows XP Pro, Pentium 4 2.6Ghz, RAM 512Mb, carte graphique NVidia GeForce FX 5200.

L'action envisagée consiste à créer des objets graphiques en variant les outils et les paramètres. Pour chaque création d'objet, nous faisons une comparaison entre trois configurations :

1) Chaque participant possède les données et chacun construit l'objet graphique demandé.

2) Un seul participant possède les données et son pilote de données fournit à l'autre ce dont celui-ci a besoin ; chacun construit l'objet graphique demandé.

3) Un seul participant possède les données ; il construit l'objet graphique et le transmet à l'autre.

Nous avons mesuré le temps Δt nécessaire à la création de l'objet graphique :

t_{start} = le moment où le participant demande la création de l'objet

t_{end} = le moment où tous les participants voient apparaître le résultat (objet graphique) sur leur écran.

$\Delta t = t_{end} - t_{start}$

Tableau 4.2 : Δt et volume des données transmises pour créer un objet graphique

Action	Configuration a)	Configuration b)	Données traitées transmises	Configuration c)	Données graphiques transmises
Isosurface I	3.4 s	93.2 s	1272 Kb	152.3 s	2090 Kb
Isosurface II	3.3 s	93.4 s	1278 Kb	4.3 s	27 Kb
Plan de coupe	1.2 s	11.8 s	194 Kb	11.1 s	174 Kb
Arrows	1.9 s	13.4 s	203 Kb	34.5 s	454 Kb

Le tableau 4.2 contient les temps Δt nécessaires à la création de l'objet graphique par trois outils de visualisation différents. Il donne également le volume des données transmises, c'est-à-dire le volume de données traitées pour la configuration b) et le volume de données graphiques pour la configuration c).

Pour l'isosurface I, la région d'intérêt est constituée d'une boîte dans laquelle les valeurs relatives à 70 * 70 * 50 (=245'000 points) doivent être fournies par le pilote de données. L'*isovalue* vaut 90 et l'objet graphique produit est constitué de 101'852 triangles. La figure 4.11 montre l'image produite sur l'écran.

Tous les paramètres de l'isosurface II sont les mêmes que ceux de l'isosurface I à l'exception de l'*isovalue* valant ici 50. L'objet graphique produit est constitué de 1'035 triangles. La figure 4.12 montre l'image produite sur l'écran.

Le Plan de coupe demande au pilote de données les valeurs du même champ scalaire que pour les deux isosurfaces ; ces valeurs sont relatives à 200 * 200 (=40'000) points situés sur un plan. La figure 4.13 montre l'image produite sur l'écran.

L'outil *Arrows* demande au pilote de données les valeurs d'un champ vectoriel représentant les courants en un plan vertical dans la mer Méditerranée ; ces valeurs sont relatives à 200 * 200 (=40'000) points. La figure 4.14 montre un exemple d'image produite par cet outil sur l'écran.

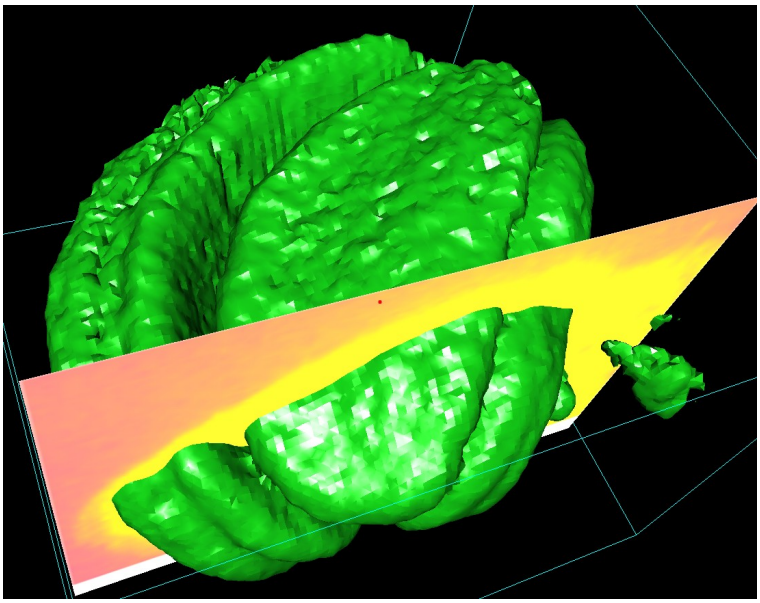

Figure 4.11 : Isosurface I

Figure 4.12 : Isosurface II

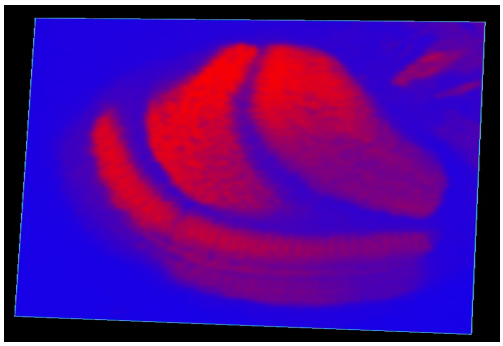

Figure 4.13 : Plan de coupe

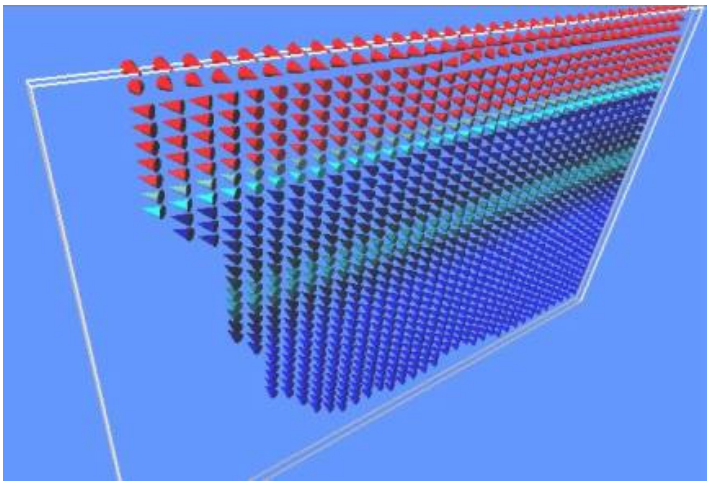

Figure 4.14 : Arrows

4.4.2 Discussion des résultats

La configuration le plus rapide est toujours a) ; en effet c'est elle qui conduit à la transmission du plus petit volume de données entre participants. Comme il a été relevé plus haut, en pratique les participants ne disposent malheureusement pas toujours de toutes les données

brutes sur leur ordinateur. Comparons maintenant les différences de performance entre les configurations b) et c).

Dans le cas de l'Isosurface, on se rend compte que le volume des données graphiques peut varier fortement suivant la valeur de l'*isovalue* (90 ou 50 dans notre cas). Il est cependant impossible de prévoir le volume de données graphiques produites en se basant uniquement sur les paramètres. Il n'est donc pas possible de définir une heuristique qui dans tous les cas choisit la meilleure configuration.

Pour l'outil Plan de coupe, le volume des données transmis pour les configurations b) et c) est à peu près le même. Comme il a été relevé dans la section 4.3.1, cela est dû au fait que, pour un plan de coupe comprenant m*n points, on demandera m*n valeurs au pilote de données et que le résultat graphique sera lui aussi composé de m*n valeurs. Pour cet outil, les deux configurations se révèlent équivalentes.

Pour l'outil *Arrows*, le volume de données de visualisation transmis dans la configuration b) est plus petit que dans la configuration c). C'est dû au fait que pour construire une des flèches de l'objet graphique, une seule valeur des données traitées est nécessaire, tandis qu'une flèche est composée d'un grand nombre de triangles.

En utilisant l'heuristique réalisée dans le système pour l'Isosurface, la configuration choisie est b) ou c) avec la même probabilité de 50%, $t_{ObjetGraphique}$ étant inconnu. Pour l'outil Plan de coupe, la configuration c) est choisie ($t_{Donnée} \approx t_{ObjetGraphique}$ et $t_{Calcul} < t_{Donnée}$). Pour l'outil *Arrows*, la configuration b) est choisie ($t_{Donnée} < t_{ObjetGraphique}$ et $t_{Calcul} << t_{Donnée}$). Donc pour ces deux derniers outils, cette heuristique trouve la meilleure configuration et pour l'outil Isosurface, elle a 50% de probabilité d'obtenir la meilleure configuration.

4.5 Extensions

Cette section décrit d'autres possibilités de réduire le temps d'attente. Les deux premières ont été réalisées.

4.5.1 Cache

Un cache permet d'éviter que le pilote de données ne calcule plusieurs fois les mêmes données, donc de réduire le temps d'attente pour les participants. Cette situation se présente lorsqu'un objet graphique est créé dans le monde partagé, les mêmes calculs étant alors effectués par chacun des participants, qui demandent presque simultanément les mêmes données. Un cache est également utile dans d'autres situations, en particulier lorsqu'un participant met à jour un objet graphique précédemment créé en ne modifiant que certains de ses arguments. Les données utilisées étant les mêmes, un cache permet là aussi de n'effectuer qu'une seule fois le calcul. C'est pourquoi notre système comprend un tel cache [81].

Les principales difficultés pratiques dans la réalisation du cache sont liées au choix des informations de visualisation à stocker dans le cache ainsi qu'au choix de l'information à effacer lorsque le volume occupé par le cache devient trop grand. Les résultats obtenus montrent des gains de performance substantiels (pouvant aller jusqu'à 60%) apportés grâce à l'utilisation du cache.

4.5.2 Amélioration de la transmission dans le cas de plus de deux participants

Si plus de deux participants collaborent à une session de visualisation, plusieurs améliorations sont possibles pour augmenter l'efficacité :

- Il faut éviter la centralisation des calculs sur un seul serveur. En effet, cela peut constituer un goulet d'étranglement et rendre le système peu tolérant aux pannes. A titre d'exemple, si six participants collaborent et que seuls deux participants possèdent les mêmes données brutes, les serveurs de données doivent se repartir le travail de manière à servir chacun deux participants.

- En plus de cette dernière mesure, il est possible de tenir compte de la répartition physique des participants. Il est plus efficace de transmettre des informations entre deux participants situés très près l'un de l'autre qu'entre deux participants très éloignés. En VSC, il est fréquent que les participants soient situés sur le même site ; des groupes de participants proches peuvent ainsi se former. Cette répartition doit être exploitée afin de ne pas surcharger inutilement le réseau par des émissions redondantes de données. Une solution technique est de concevoir un réseau s'adaptant dynamiquement et de manière totalement transparente à l'emplacement géographique des participants. A chaque arrivée d'un nouveau participant, le système regroupe les participants proches.

Notre système comprend ces deux mécanismes afin d'améliorer l'efficacité [1].

4.5.3 Autres extensions possibles

Pour réduire le temps de transmission des données, il est envisageable de comprimer ces données. Bien évidemment, une telle compression entraîne un surcroît de calculs tant pour l'ordinateur du participant envoyant les données, qui doit les comprimer, que pour l'ordinateur du participant recevant les données, qui doit les décomprimer. Pour que cette extension réduise le temps d'attente, il faut donc que le temps gagné dans la transmission soit plus grand que le temps perdu dans le calcul. Il faut aussi pouvoir estimer l'impact de la compression sur le temps de transmission.

4.6 Conclusion

Nous avons présenté les différentes configurations possibles pour transmettre les informations de visualisation entre les participants. Suivant de nombreux facteurs (comme l'outil utilisé, les données brutes, le matériel à disposition), les configurations conduisent à des temps d'attente très différents. Nous avons comparé ces configurations et essayé d'en tirer une heuristique capable de choisir chaque fois une bonne configuration. Il en ressort qu'il est impossible d'obtenir une heuristique choisissant dans toutes les situations la meilleure configuration. Afin d'obtenir de bons résultats malgré les nombreuses inconnues, nous avons utilisé, dans les cas où une configuration ne s'avère pas être clairement la meilleure, une heuristique probabiliste.

Le temps d'attente peut aussi être réduit par d'autres moyens, que ce soit en utilisant un cache, en tenant compte de la répartition physique des participants ou en comprimant les données.

Tout cela permet d'améliorer notablement les performances. En revanche il ne sert à rien de disposer d'un système performant s'il n'est pas facile à utiliser. Le chapitre suivant présente donc les moyens mis en œuvre afin de faciliter la collaboration dans notre système ainsi qu'une évaluation des apports fournis par ces moyens.

Chapitre 5 Satisfaction cognitive

Rappelons que nous avons deux objectifs dans la réalisation de notre système : l'efficacité et la satisfaction des utilisateurs. Le premier objectif a été traité dans le chapitre 4. Dans ce chapitre, nous considérons le deuxième objectif. Plus précisément, nous allons discuter des besoins des utilisateurs et évaluer la satisfaction perçue par les utilisateurs de notre système.

5.1 Besoins en collaboration dans la VSC

Avant de procéder à l'évaluation du système, nous formulons les besoins de l'utilisateur et décrivons les moyens que nous avons mis en œuvre pour les satisfaire. Dans cette section, nous nous concentrons sur les aspects liés à la collaboration.

Pour collaborer, les participants ont besoin de communiquer. Une fois qu'un participant a formulé un message, il ne doit pas seulement se soucier de le faire parvenir aux autres participants, mais il doit en plus s'assurer qu'il a été compris correctement. Le *grounding* (section 2.2.2) est un élément essentiel pour construire des interactions entre les participants.

5.1.1 Médias de communication

Pour les participants, l'idéal serait certainement de pouvoir communiquer comme s'ils étaient l'un à coté de l'autre et en face d'un seul écran. Il a été montré dans la littérature [68] [73] que l'influence des médias de communication utilisées ne dépend guère du domaine d'application. Nous formulons l'hypothèse que les résultats obtenus dans le domaine de la visualisation scientifique collaborative (VSC) devraient être les mêmes que dans d'autres domaines où la collaboration intervient [56], c'est-à-dire que l'audio est plus efficace que le *chat* et la combinaison vidéo-audio n'apporte rien de plus que l'audio tout seul. En revanche, le *chat* offre des fonctionnalités que les autres médias de communication n'offrent pas ; il ne doit donc pas seulement être considéré comme une alternative lorsque aucun autre moyen n'est disponible [88].

Les médias de communication facilitent le *grounding* ainsi que, dans une moindre mesure, l'*awareness* [68] (section 2.2.3). Les contraintes définies par Clark [19], et précédemment décrites dans la section 2.2.4, sont présentées pour les cas particuliers de l'audio et du *chat* dans le tableau 5.1.

Tableau 5.1 : Contraintes pour l'audio et le *chat*

Contrainte	Audio	*Chat*
Coprésence		
Visibilité		
Audibilité	X	
Cotemporalité	X	X
Simultanéité	X	X
Séquentialité	X	X
Re-consultabilité		X
Révisabilité		X

5.1.2 Support de l'*awareness* dans les systèmes collaboratifs

Dans cette section, les principaux aspects de l'*awareness* sont traités de manière générale, alors que la section 5.1.3 traite le cas particulier de l'*awareness* dans le cadre de la VSC.

L'*awareness* est utile dans de nombreuses activités de collaboration telle que la coordination des actions ou la communication avec d'autres participants. Maintenir l'*awareness* est une condition nécessaire, évidemment non suffisante, pour que les participants puissent collaborer de manière naturelle et efficace dans un environnement de travail réparti.

5.1.2.1 Capteurs et indicateurs

D'un point de vue conceptuel et pratique, il convient de séparer clairement la collecte d'informations contribuant à l'*awareness* et la présentation de ces informations (après un éventuel traitement). Cette séparation se traduit dans les faits par deux types d'entités distinctes servant l'un à la collecte de l'information (les capteurs) et l'autre à la présentation de l'information (les indicateurs).

Pour contribuer à l'*awareness* des participants, il convient donc de répondre à deux questions : « Quelles informations doit-on fournir ? » (contenu) et « Comment présenter ces informations ? » (forme).

5.1.2.2 Contenu et forme

Contenu

Plusieurs questions se posent : quelles sont les informations contribuant à l'*awareness* ? Quels évènements doivent être saisis ? Quelles informations doivent être distribuées ? Gutwin propose un cadre conceptuel pour contribuer à l'*awareness* et il présente également un ensemble de questions qu'il convient de se poser, comme décrit à la section 2.2.3. Les réponses à ces questions précisent les besoins des participants en relation avec l'*awareness*.

La présentation du contenu dépend des informations dans le sens où toutes les informations n'ont pas la même importance (certaines informations sont cruciales pour accomplir une tâche alors que d'autres sont secondaires) et que les informations les plus importantes doivent être mises en valeur.

Dans cette optique, Weir et Cockburn [106] proposent, dans leur système DOME, six niveaux de support de l'*awareness* (d'un support nul à un support très élevé) ce qui permet aux participants de choisir un niveau élevé de support pour une activité qui se déroule en étroite collaboration et un niveau beaucoup plus faible, voire nul, pour un travail indépendant.

Forme

Pour réaliser des indicateurs de qualité, nous pouvons nous inspirer des enseignements tirés des travaux liés à l'*awareness*. Tout d'abord, il convient de ne donner ni trop ni trop peu d'informations : 4 à 5 indicateurs sont déjà trop [80]. Ensuite, pour la bonne compréhension des informations fournies, les indicateurs doivent posséder une sémantique simple.

Du point de vue de l'implémentation, il convient aussi de prendre en compte les capacités du système informatique et de chercher un compromis entre la richesse des indicateurs et les ressources informatiques nécessaires à leur élaboration. Dans cette optique, Lee et

Girgensohn [57] ont implanté un système vidéo qui fournit une vue générale de l'activité d'une communauté grâce à une matrice d'images fixes qui sont rafraîchies régulièrement. Ces images représentent les participants et leur taux d'activité. Un mécanisme permet de mettre en avant les dernières modifications effectuées.

Spécificité d'une solution

Dans les travaux rapportés dans la littérature, les solutions proposées sont souvent difficilement applicables à d'autres domaines [91]. Bien qu'il existe des besoins généraux communs à de nombreuses applications, comme décrit par Gutwin [39], une grande partie des solutions dépend fortement du domaine d'application. Il n'est donc pas possible de proposer un support de l'*awareness* qui soit totalement générique ; néanmoins certaines solutions sont réutilisables dans d'autres domaines.

5.1.3 Support de l'*awareness* dans la VSC

Après avoir examiné, dans la généralité, différents aspects de l'*awareness*, nous allons traiter les problèmes liés à l'*awareness* dans le cadre de la VSC. Les problèmes les plus importants consistent en l'identification des informations à fournir (contenu) et en leur présentation (forme).

Dans la plupart des systèmes actuels de VSC, seule une partie de l'information nécessaire au support de l'*awareness* est mise à disposition (section 2.4). Il est donc plausible qu'un support accru de l'*awareness* conduit à une meilleure facilité d'utilisation ; c'est une des hypothèses que nous allons essayer de démontrer. Cette hypothèse se vérifie dans d'autres domaines d'application [40], mais nous voulons examiner si elle se confirme dans le cas particulier de la VSC.

Rappelons que l'*awareness* est dynamique ; pour le maintenir, il faut collecter les informations actuelles provenant de l'environnement. Quand un participant a la réponse aux questions des trois catégories (« qui ? », « quoi ? », « où ? ») du tableau 5.2, les activités collaboratives sont facilitées. Autant il est aisé de répondre à ces questions lorsque les participants sont l'un à côté de l'autre, autant il est difficile d'y répondre dans un monde partagé.

Tableau 5.2 : Eléments de l'*awareness* dans la VSC

Catégorie	Eléments	Questions
Qui	Présence	Y-a-t-il quelqu'un dans l'espace de travail ?
	Identité	Qui participe ? Qui est-ce ?
	Identification de l'auteur	Qui effectue telle action ?
Quoi	Action	Que font les participants ?
	Intention	Quel est le but de telle action ?
	Artefact	Sur quel artefact travaillent les participants ?
Où	Localisation	Où travaillent les participants ?
	Regard	Où regardent les participants ?
	Vue	Que peuvent voir les participants ?
	Portée	Quel est le champ d'action des participants ?

Nous allons reprendre les éléments de Gutwin et nous allons répondre aux deux questions, « **Quelles informations doit-on fournir?** » et « **Comment présenter ces informations?** », pour tous les éléments. Pour cela, nous utilisons les indications données

dans son cadre conceptuel [37]. Nous nous rappelons qu'il s'agit de recueillir un maximum d'informations utiles tout en les présentant de manière pertinente (ni trop, ni pas assez).

Catégorie « Qui »

Y-a-t-il quelqu'un dans l'espace de travail ?

La liste indiquant les participants doit être disponible. Il s'agit d'une information dynamique, mais qui varie peu. L'arrivée et le départ d'un participant sont annoncés par des messages.

Une représentation visuelle, par exemple par un avatar, permet indirectement de savoir si une personne est présente ou non.

Qui participe ? Qui est-ce ?

Un participant doit fournir au minimum son nom. Des informations supplémentaires, comme ses compétences ou sa localisation peuvent être utiles. Ces informations statiques sont fournies par le participant au début d'une session de visualisation. La connaissance de ces informations facilite le *grounding*. En VS, il est fort probable que les participants se connaissent et ainsi se font une idée des compétences des autres.

La personnalisation des avatars grâce à des caractéristiques visuelles telles qu'une couleur ou une forme spécifique permet de reconnaître plus aisément les participants.

Qui effectue telle action ?

Chaque action effectuée sur un *artefact* doit pouvoir être identifiée. Cette information est utile pour améliorer le *feedthrough* (section 2.2.3). Le fait qu'un avatar soit proche de la position où une action se déroule est un indice que le participant en question effectue cette action. Il est possible de fournir en même temps de l'information supplémentaire, telle que la liste complète des modifications effectuées sur cet *artefact*.

Comme solution technique, on pourrait songer à colorier entièrement l'*artefact* mais vu qu'en VS les couleurs sont très utilisées, cette technique est peu judicieuse. Par contre, lors d'une sélection d'un *artefact*, le fait de l'entourer par une boîte colorée dans la couleur du participant qui l'a créé permet de reconnaître simplement son créateur.

Catégorie « Quoi »

Que font les participants ? Que se passe-t-il ?

Il faut connaître le **statut** des participants. Un participant peut être en train de travailler dans la session de visualisation (par exemple, en créant des objets graphiques ou en explorant le monde 3D) ou de communiquer ; mais il peut aussi s'être absenté (par exemple pour répondre au téléphone ou pour faire des recherches bibliographiques).

Lorsqu'un participant se déplace dans le monde 3D, il fait connaître implicitement son statut ; c'est un cas particulier de la *consequential communication*.

Il s'agit de collecter de la manière la plus automatisée possible ces informations. Par exemple, si un participant n'interagit pas avec le système pendant un laps de temps donné, son statut

passe à absent. Le participant peut aussi indiquer lui-même son statut, par exemple s'il est occupé par une activité externe.

Cette information peut être présentée sous forme visuelle, par exemple en indiquant par une icône qu'un participant est en train d'écrire un message sur le *chat* ou en donnant à un avatar une certaine couleur pour indiquer qu'il a bougé récemment.

Quel est le but de telle action ?

Un participant peut faire connaître ses intentions de manière volontaire ou involontaire.

Par un message émis à travers un média de communication, un participant peut faire connaître volontairement ses intentions. En se déplaçant, un participant dévoile involontairement ses intentions, notamment à ceux qui suivent sa position.

Sur quel artefact travaillent les participants ?

La position d'un participant permet de situer les *artefacts* sur lesquels il est susceptible de travailler. Dans la vie de tous les jours, nous agissons là où nous nous trouvons. Nous ramassons des objets proches de nous. En VSC, nous nous approchons d'un *artefact* pour l'explorer visuellement car notre acuité visuelle est limitée. La proximité permet de relier les personnes aux activités et les personnes entre elles. Les derniers *artefacts* sélectionnés par un participant sont aussi un indice pour connaître ses intentions. En VS il est nécessaire de pouvoir indiquer des endroits précis dans l'espace, comme sur la surface d'un *artefact*. Par exemple, un médecin veut pouvoir préciser la position d'une tumeur sur un organe humain.

Il s'agit d'une information utile localement. En effet, il n'est pas utile pour un participant de savoir qu'une personne travaille sur un *artefact* situé à l'opposé de son espace de travail. L'indication volontaire d'un *artefact* doit être possible ; cela peut se faire grâce à un pointeur.

Catégorie « Où »

Où travaillent les participants ?

Il faut indiquer la position du participant ainsi que la direction de son regard. Il s'agit d'une information fournie de manière involontaire.

Il semble judicieux d'utiliser un avatar pour représenter un participant dans le monde 3D. Par conséquence, sa position, son orientation et la direction de son regard sont connus.

Où regardent les participants ?

L'indication volontaire d'un endroit précis doit être possible. En effet en VS, les spécialistes se focalisent sur un endroit bien défini de la visualisation.

Un pointeur est un moyen d'indiquer un endroit précis. Les participants doivent avoir la possibilité de le cacher ou de l'afficher.

Que peuvent voir les participants ?

L'idéal est d'avoir une copie exacte de la vue d'un participant.

Cette information doit être obtenue de manière volontaire ; en effet afficher systématiquement la vue de tous les participants ne fait qu'alourdir le système. Pour respecter la sphère privée, seul le monde partagé est accessible. L'accès à la vue d'un autre participant peut se faire en sélectionnant l'avatar. En effet, en VSC, on va voir qui travaille à proximité des *artefacts* qui nous intéressent et par la suite se connecter à sa vue.

Quel est le champ d'action des participants ?

Le champ d'action d'un participant est défini en se connectant à sa vue.

5.2 Expérimentation

Dans la section précédente, nous avons présenté les mécanismes réalisés pour améliorer l'*awareness*. Dans notre système, le support de l'*awareness* se fait grâce à l'aide des composants suivants : avatars, pointeurs et monde partagé/privé. Cependant, ces fonctionnalités ne servent à rien si les participants ne parviennent pas à les utiliser correctement. Pour tester notre implémentation, nous procédons donc à une expérimentation du système.

5.2.1 Caractéristiques à évaluer

Afin d'évaluer la satisfaction, nous avons effectué une expérimentation. Dans cette section, nous allons en préciser les **méthodes**, puis interpréter les **résultats**, que nous comparerons avec nos **hypothèses** de travail.

Pour débuter, nous formulons la question principale à laquelle nous cherchons à répondre :

La partie collaborative de notre système répond-elle aux besoins des utilisateurs ?

Comme relevé par Neale [68], l'évaluation des systèmes collaboratifs est nettement plus difficile que l'évaluation des systèmes mono-utilisateurs. Les facteurs à considérer sont variés et complexes. Des facteurs cognitifs individuels doivent être pris en compte, de même que des facteurs collaboratifs et sociaux [69]. De nombreuses expérimentations entreprises dans le but d'évaluer des systèmes collaboratifs ont été effectuées dans le domaine de l'interaction homme-machine (*Human Computer Interaction* – HCI) ; elles ont porté, par exemple sur l'évaluation du système DOVE [54] [77] ou de la collaboration dans un jeu multi-utilisateur par Nova [72].

Nous posons comme acquis que les participants ont besoin de la collaboration, sinon ils n'utiliseraient pas un tel système. Nous nous intéressons aux effets des médias de communication et de nos principaux indicateurs sur la facilité d'utilisation du système. Nous allons voir leur influence sur le *grounding* et l'*awareness*.

5.2.2 Comparaisons à effectuer

Il s'agit d'évaluer l'utilité des fonctionnalités collaboratives de notre système dans une situation où les participants travaillent à distance. Cela est réalisé grâce à deux comparaisons :

1) Nous comparons deux médias de communication dans cette expérimentation : le ***chat*** (intégré dans le système) et la combinaison de l'**audio** (en utilisant le système Skype [90]) avec le *chat* intégré à notre système.

2) Nous comparons le système avec et sans certains **indicateurs** censés améliorer l'*awareness*. Pour que l'expérimentation ne demande pas une longue phase d'apprentissage de la part des participants, nous avons choisi de tester uniquement deux indicateurs : les **avatars** et les **pointeurs**.

Dans ce qui suit, le terme « indicateur » se réfère donc aux **avatars** et aux **pointeurs.**

5.2.3 Tâche à accomplir

Comme énoncé auparavant, l'évaluation de systèmes collaboratifs est complexe. En effet de nombreux facteurs subjectifs et difficilement mesurables jouent un rôle. De plus, nous ne disposions pas de suffisamment de participants capables d'accomplir des tâches complexes et réalistes. C'est pourquoi nous avons demandé aux participants d'accomplir une **tâche** simple mais qui exige d'effectuer des actions élémentaires propres à la VS. De plus, nous avons recherché un aspect ludique afin de stimuler les participants à s'impliquer réellement à accomplir cette tâche.

La tâche à accomplir s'apparente à un jeu. Une paire de participants se trouve face à un monde 3D. Ce monde est rempli d'**objets géométriques** possédant deux propriétés de base : une **forme** (cube, sphère, cône ou cylindre) et une **couleur** (rouge, rose, vert, jaune ou brun). En outre, en cliquant sur l'objet géométrique, deux autres propriétés s'affichent : une **lettre** et une référence à l'**objet géométrique suivant** (indiqué par sa forme et sa couleur). Les objets géométriques sont ainsi ordonnés.

Le but est de parcourir le plus rapidement possible six objets géométriques dans l'ordre induit par les références. Les six lettres ainsi obtenues, placées dans le bon ordre, forment une combinaison qui permet d'ouvrir un coffre et de gagner le jeu.

Afin de forcer la collaboration entre les deux participants, l'un d'entre eux n'a accès qu'à la couleur de l'objet géométrique suivant, tandis que l'autre participant n'a accès qu'à sa forme. La mise en commun de ces deux informations permet d'identifier l'objet géométrique suivant.

La forme et la couleur du premier objet géométrique sont communiquées par le système à travers le *chat*, ce qui marque le début du jeu. Ensuite, les participants n'ont droit qu'à une seule tentative pour ouvrir le coffre. En effet, que la combinaison soit exacte ou erronée, le jeu s'arrête. Un exemple de monde 3D avec ses objets géométriques est donné à la figure 5.1. Chaque tâche à accomplir possède sa propre **combinaison** ; en revanche le monde 3D avec ses objets géométriques reste le même pour toutes les tâches.

Pour que les participants puissent accomplir la tâche, ils doivent préalablement se familiariser avec les fonctionnalités du système (déplacement dans le monde 3D, utilisation des avatars, pointeurs, *chat*, etc.). Afin d'accomplir cette tâche, les participants vont élaborer, consciemment ou inconsciemment, une **stratégie**.

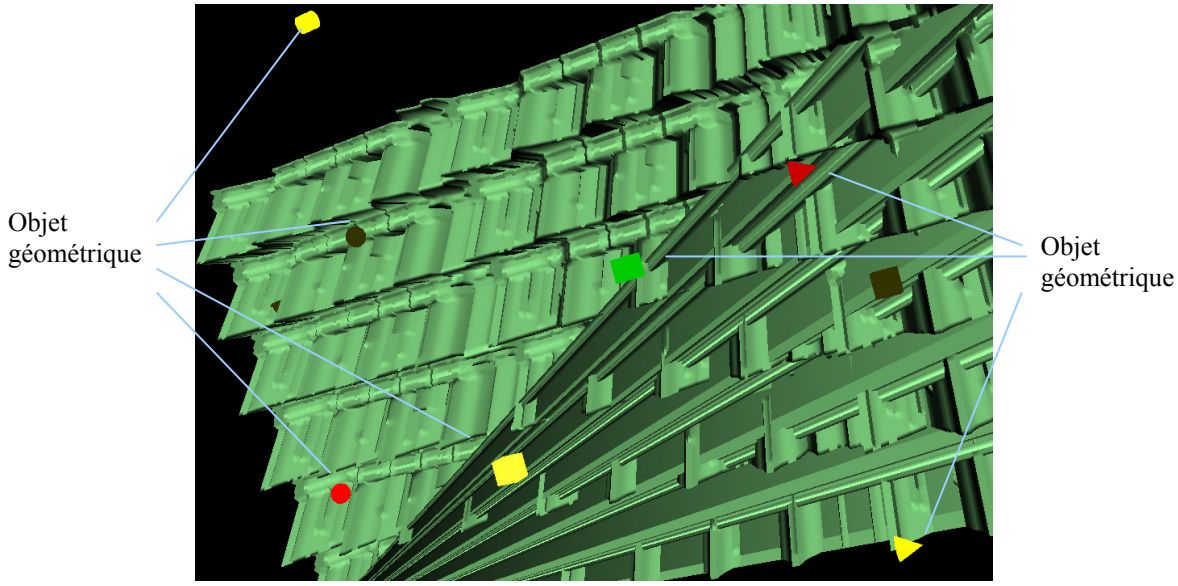

Figure 5.1 : Exemple du monde 3D avec les objets géométriques

5.2.4 Moyens à disposition pour accomplir les tâches

Les participants effectuent quatre **parties**, chacune consistant à accomplir une tâche avec des **moyens** différents. L'ordre des tâches est identique pour toutes les paires de participants (autrement dit la succession de combinaisons à découvrir est la même pour tous), seuls les moyens mis à disposition varient, comme montré dans le tableau 5.3. Pour éviter que l'expérience acquise n'influence les statistiques, l'ordre des moyens mis à disposition (**situation** A, B, C ou D) est différent d'une paire à l'autre.

Tableau 5.3 : Moyens mis à disposition pour accomplir la tâche

Média / Indicateur	***Chat***	***Chat* et audio**
Sans avatars, ni pointeurs	situation A	situation C
Avec avatars et pointeurs	situation B	situation D

Les actions et les conversations des participants sont enregistrées. Ainsi, nous pouvons déterminer les stratégies utilisées et les difficultés rencontrées par les participants.

5.2.5 Participants

Pour obtenir un résultat suffisamment fiable, **12 paires** de personnes ont participé à l'expérimentation. Comme précisé par Sonnerwald [92], il est très difficile d'estimer le nombre de participants nécessaire à une telle expérimentation. Nous nous sommes basés sur des expérimentations similaires [40, 43, 72], dans lesquelles interviennent entre 10 et 20 paires de participants.

Relevons que, selon l'enquête de Baker [4], une évaluation par trois à cinq personnes permet de déceler plus de 50% des défauts des systèmes collaboratifs. Cela nous sera utile car, même si ce n'est pas notre objectif principal, nous demandons aux participants de faire des commentaires sur le système pour en déceler les défauts.

Nous avons demandé à des étudiants de notre université de participer à l'expérimentation. En majorité, il s'agissait d'étudiants en informatique disposant déjà d'une certaine expérience du travail dans un monde 3D.

5.2.6 Critères d'évaluation

Pour comparer l'influence des divers médias de communication et indicateurs mis à disposition, il faut établir des **critères d'évaluation**. L'évaluation fait appel en partie à des critères subjectifs. Nous tenons compte de deux critères :

- **Temps *t*** mis pour accomplir la tâche. Il s'agit du temps s'écoulant entre le moment où les participants reçoivent la couleur et la forme du premier objet géométrique et le moment où un participant délivre la combinaison (qu'elle soit exacte ou erronée).

- **Intuitivité *i*** du système. Cette notion étant subjective, nous avons utilisé des questionnaires que les participants devaient remplir individuellement après avoir accompli une tâche. Comme indiqué dans le tableau 5.4, les deux questionnaires $q1$ et $q2$ mesurent six composantes de l'intuitivité.

Tableau 5.4 : Composante de l'intuitivité mesurée

Composante de l'intuitivité	Composante de l'intuitivité mesurée par q_1	Composante de l'intuitivité mesurée par q_2
Difficulté perçue	Difficulté perçue	
Effort	Effort absolu	Effort relatif
Concentration	Concentration	
Communication	Communication absolue	Communication relative
Awareness	*Awareness* absolu	*Awareness* relatif
Temps d'exécution perçu		Temps d'exécution perçu

Dans le questionnaire q_1, il s'agissait d'évaluer par un entier sur une échelle allant de 1 à 5, chacune des composantes suivantes : difficulté perçue, effort absolu, concentration, communication absolue et *awareness* absolu. Chaque participant remplissait ce questionnaire après chaque partie, puis il n'y avait plus accès. Il s'agit de comparaisons **absolues**.

Dans le questionnaire q_2, il s'agissait de comparer les quatre parties auxquelles avait participé une personne. Plus précisément, chaque participant devait comparer les composantes suivantes : effort relatif, communication relative, *awareness* relatif et temps d'exécution perçu ; il devait exprimer ce jugement par quatre entiers sur une échelle plus fine allant de 0 à 10. Le participant disposait du même questionnaire pour les quatre parties. Il s'agit de comparaisons relatives.

Les intitulés exacts des questions sont indiqués dans le tableau 5.5 et le tableau 5.6. Comme nous voulions éviter de devoir expliquer la notion d'*awareness* aux participants, nous interprétons la dernière question des questionnaires q_1 et q_2 comme une évaluation de la qualité du support de l'*awareness*.

Tableau 5.5 : Composantes de l'intuitivité du questionnaire q_1

Questionnaire q_1 :	**Composante de l'intuitivité mesurée :**
1 2 3 4 5 Comment évaluez-vous la difficulté pour accomplir la tâche ? Facile ☐ ☐ ☐ ☐ ☐ Difficile	Difficulté perçue
Quel effort avez-vous dû fournir pour accomplir la tâche ? Peu d'effort ☐ ☐ ☐ ☐ ☐ Grand effort	Effort absolu
Quel niveau de concentration avez-vous dû avoir pour accomplir la tâche ? Faible ☐ ☐ ☐ ☐ ☐ Elevé	Concentration
Comment évaluez-vous la difficulté pour communiquer avec votre partenaire ? Facile ☐ ☐ ☐ ☐ ☐ Difficile	Communication absolue
Comment évaluez-vous la facilité de se déplacer et d'indiquer des endroits précis dans le monde ? Facile ☐ ☐ ☐ ☐ ☐ Difficile	*Awareness* absolu

Tableau 5.6 : Composantes de l'intuitivité du questionnaire q_2

Questionnaire q_2 :	**Composante de l'intuitivité mesurée :**
Combien de temps avez-vous mis pour accomplir la tâche ? Peu Beaucoup 0 1 2 3 4 5 6 7 8 9 10	Temps d'exécution perçu
Quel effort avez-vous dû fournir pour accomplir la tâche ? Petit Grand 0 1 2 3 4 5 6 7 8 9 10	Effort relatif
Comment évaluez-vous la difficulté pour communiquer avec votre partenaire ? Facile Difficile 0 1 2 3 4 5 6 7 8 9 10	Communication relative
Comment évaluez-vous la facilité de se déplacer et d'indiquer des endroits précis dans le monde ? Facile Difficile 0 1 2 3 4 5 6 7 8 9 10	*Awareness* absolu

Enfin, après les quatre parties, chaque participant devait répondre au questionnaire q_3 et noter ses commentaires ; cela a permis de mettre en évidence les principales qualités et défauts du système. Dans le questionnaire q_3, présenté dans le tableau 5.7, il s'agissait d'évaluer par un entier de 1 à 5 différents aspects de la facilité d'utilisation du système.

Tableau 5.7 : Questionnaire q_3

	1	2	3	4	5	
J'utiliserais volontiers ce système						
Pas d'accord	☐	☐	☐	☐	☐	D'accord
Je trouve le système trop complexe						
Pas d'accord	☐	☐	☐	☐	☐	D'accord
Je trouve le système simple d'utilisation						
Pas d'accord	☐	☐	☐	☐	☐	D'accord
Les fonctionnalités du système sont facilement identifiables						
Pas d'accord	☐	☐	☐	☐	☐	D'accord
J'imagine que la plupart des gens arriveront facilement à apprendre ce système						
Pas d'accord	☐	☐	☐	☐	☐	D'accord
Je trouve le système très difficile à utiliser						
Pas d'accord	☐	☐	☐	☐	☐	D'accord
Je maîtrise le système						
Pas d'accord	☐	☐	☐	☐	☐	D'accord
Je dois encore apprendre beaucoup de choses avant de maîtriser le système						
Pas d'accord	☐	☐	☐	☐	☐	D'accord
J'imagine que la plupart des gens arriveront facilement à apprendre ce système						
Pas d'accord	☐	☐	☐	☐	☐	D'accord

5.2.7 Hypothèses

Les comparaisons sont faites entre les moyens mis à disposition pour accomplir les tâches. Nos hypothèses sont liées aux deux questions posées au début de ce chapitre. Nous formulons deux hypothèses :

Hypothèse H_1 : $t_{chat+\text{audio}} < t_{chat}$ et $i_{chat+\text{audio}} < i_{chat}$

Hypothèse H_2 : $t_{\text{avec indicateurs}} < t_{\text{sans indicateurs}}$ et $i_{\text{avec indicateurs}} < i_{\text{sans indicateurs}}$

Pour H_1, le fait de disposer de l'audio devrait permettre de communiquer aisément et rapidement. Il est plus rapide de parler que de taper du texte au clavier.

Pour H_2, le fait de mettre à disposition des indicateurs pour indiquer sa position ainsi que pour désigner des endroits du monde 3D devrait permettre de réduire le temps nécessaire à l'exécution de la tâche. Ces fonctionnalités devraient aussi améliorer l'intuitivité. Nous vérifierons si l'utilisation des indicateurs est affectée par l'apprentissage nécessaire pour maîtriser les indicateurs.

5.2.8 Déroulement de l'expérimentation

Avant de débuter l'expérimentation, les participants assistent à une présentation précisant les modalités du jeu ainsi que le fonctionnement du système. De plus, quelques minutes leur sont données pour tester le système et pour accomplir une tâche dans un monde 3D simplifié comprenant cinq d'objets géométriques. Puis, chaque participant est placé dans une salle différente. La première partie est lancée ; à la fin de celle-ci, les participants remplissent les questionnaires q_1 et q_2. Cette opération est répétée pour les trois autres parties. Après ces

quatre parties, les participants répondent au questionnaire q_3 et notent leurs commentaires généraux sur l'expérimentation.

5.3 Résultats

La figure 5.2 et la figure 5.3 montrent la moyenne des valeurs des composantes de l'intuitivité.

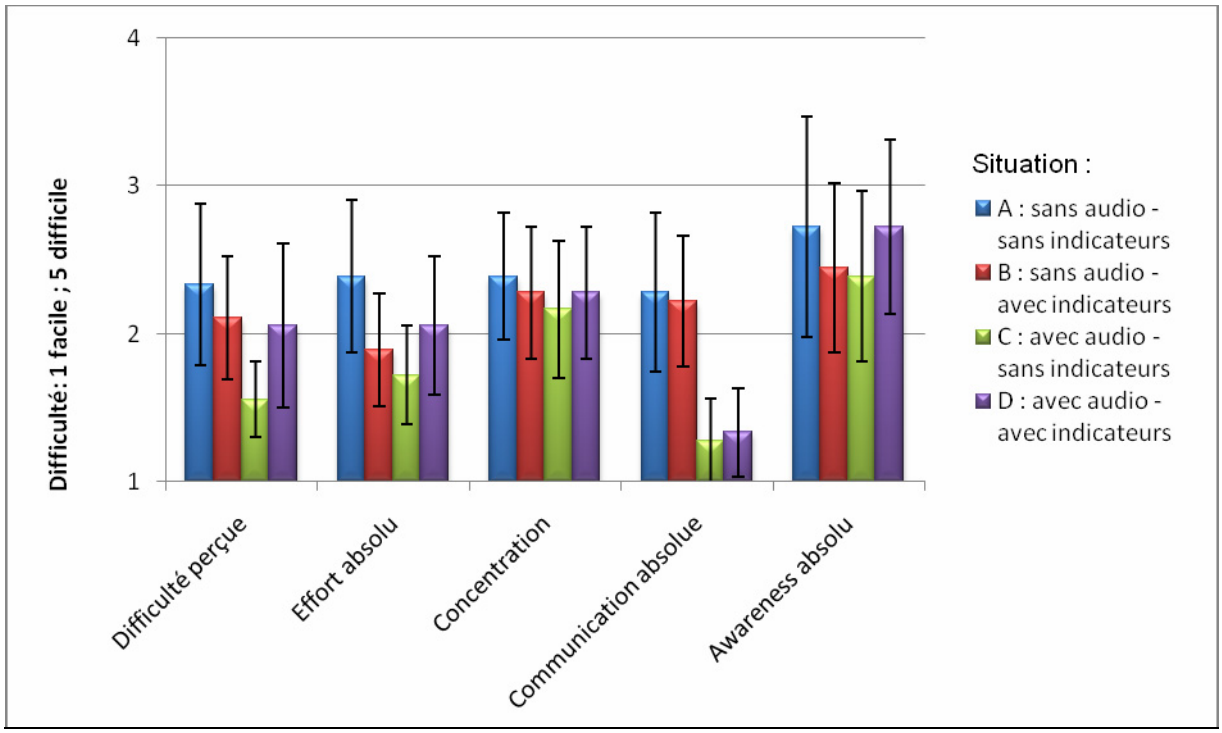

Figure 5.2 : Evaluation des composantes de l'intuitivité du questionnaire q_1 en fonction des situations

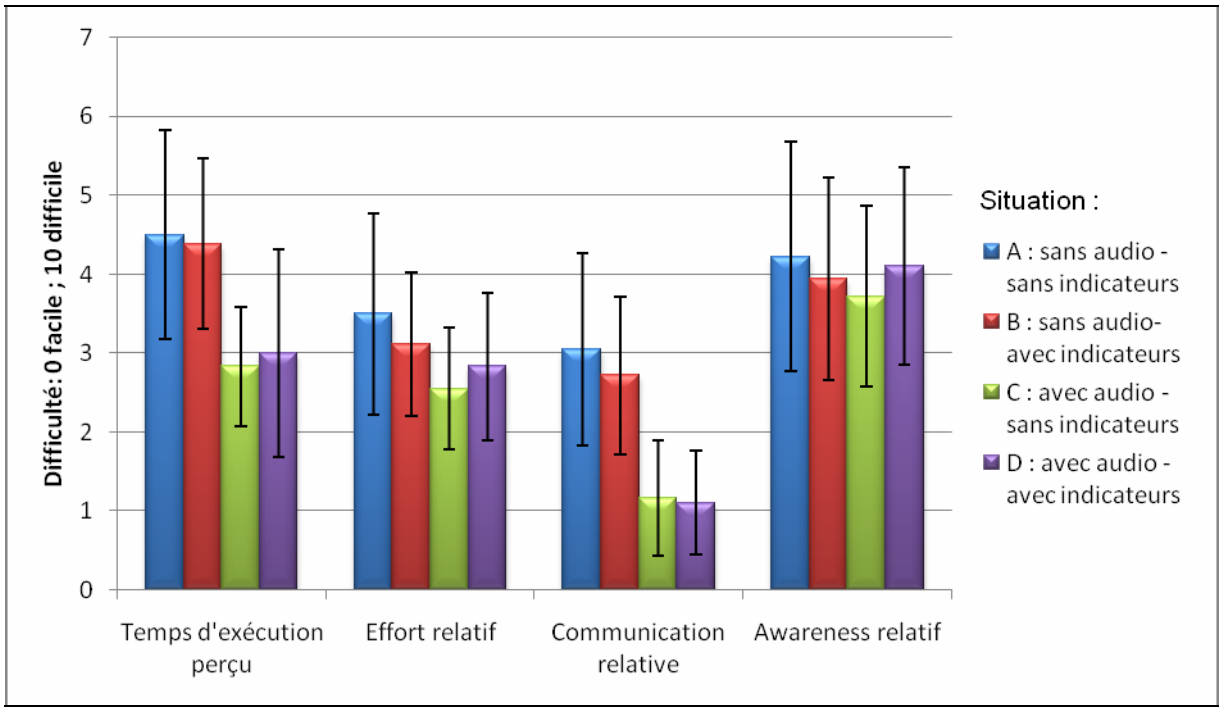

Figure 5.3 : Evaluation des composantes de l'intuitivité du questionnaire q_2 en fonction des situations

Nous discutons ces résultats dans les sections suivantes afin de contrôler si nos hypothèses se vérifient.

5.3.1 Comparaison des temps d'exécution

La figure 5.4 représente les temps mis par les différentes paires pour accomplir leurs tâches. La figure 5.5 indique la durée moyenne des parties. On se rend compte que les participants ont mis plus de temps pour accomplir la première tâche que pour les autres. Pour le confirmer, nous utilisons une analyse de la variance (*ANalysis Of Variance – ANOVA*) [53]. Il y a une différence significative entre la première partie et les autres (F=6.51, p=0.032 avec la deuxième partie, F=4.62, p=0.047 avec la troisième partie et F=10.68, p=0.005 avec la quatrième partie). En revanche, il n'y a pas de différence significative (p>0.50) entre les deuxième, troisième et quatrième parties.

Cela est dû au fait qu'au début le monde est inconnu et que les participants ne maîtrisent pas encore suffisamment le système. Pour éviter que les statistiques ne soient influencées par l'ordre dans lequel les situations sont présentées ; chacune des situations A, B, C et D du tableau 5.3 est utilisée autant de fois en première position que les autres.

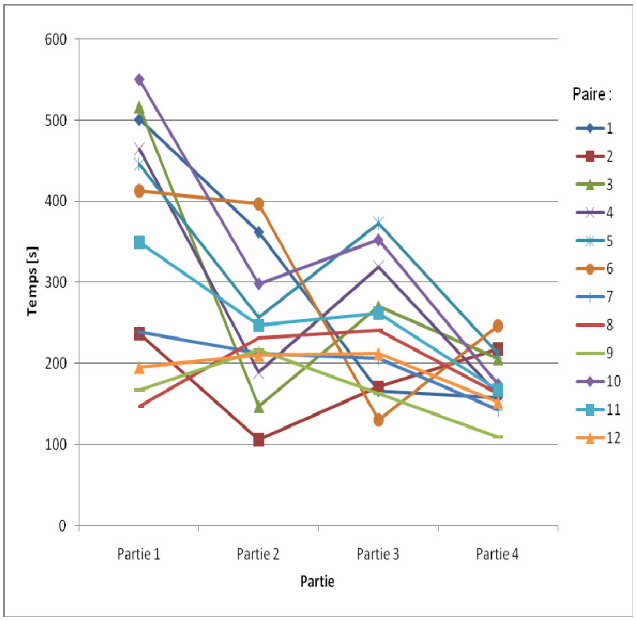

Figure 5.4 : Temps mis par les paires pour effectuer leur partie

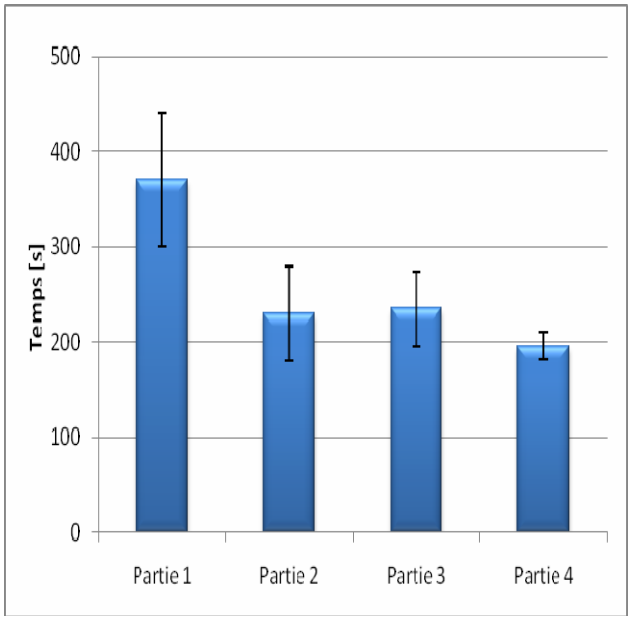

Figure 5.5 : Durée moyenne des parties

Le temps requis pour accomplir les différentes parties n'est pas identique. Il faut plus de temps pour les parties 1 et 3 que pour les parties 2 et 4. Les parties 1 et 3 demandent en effet plus de déplacements dans le monde 3D que les parties 2 et 4. Néanmoins, cette différence n'est plus perceptible pour les participants utilisant une stratégie habile.

5.3.2 Comparaison des performances en fonction des moyens à disposition

Pour mieux saisir les effets des moyens à disposition, nous isolons chacun des moyens : sans audio, avec audio, sans indicateurs, avec indicateurs. Cela est représenté par la figure 5.6, la figure 5.7 et la figure 5.8.

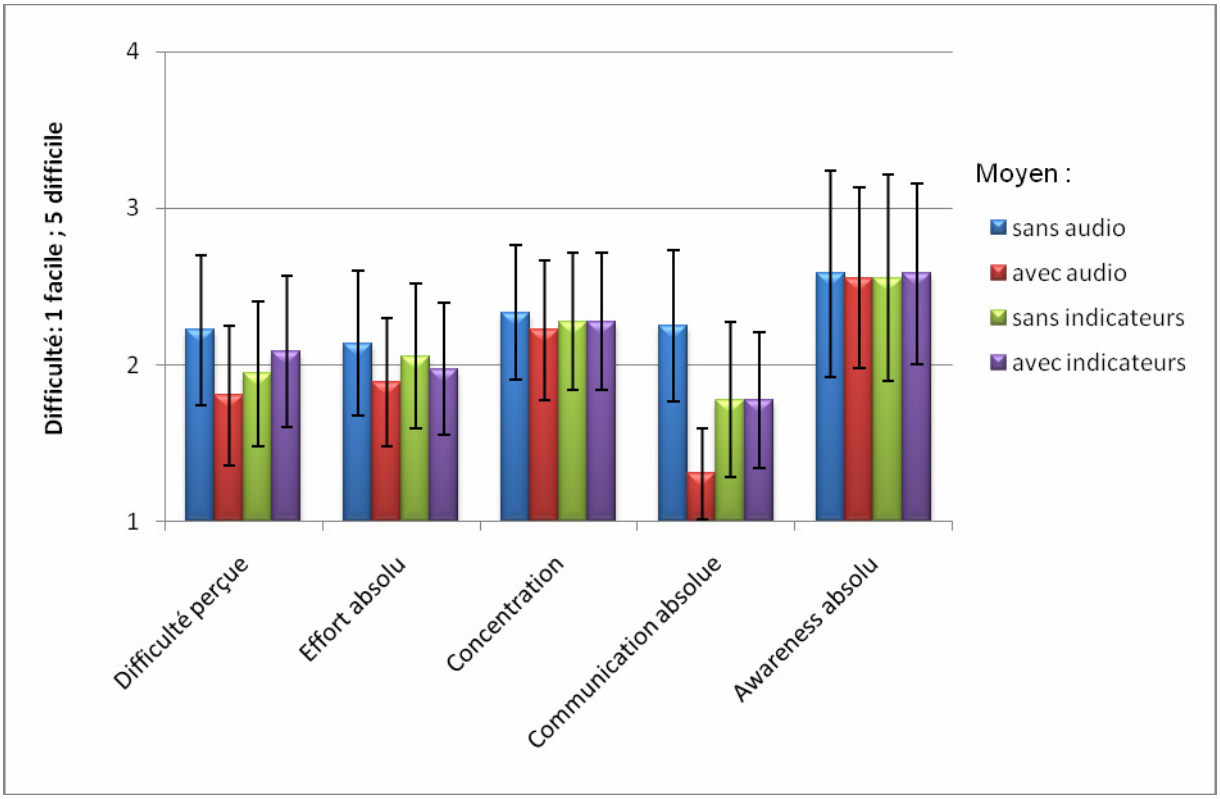

Figure 5.6 : Evaluation des composantes de l'intuitivité du questionnaire q_1 en fonction des moyens à disposition

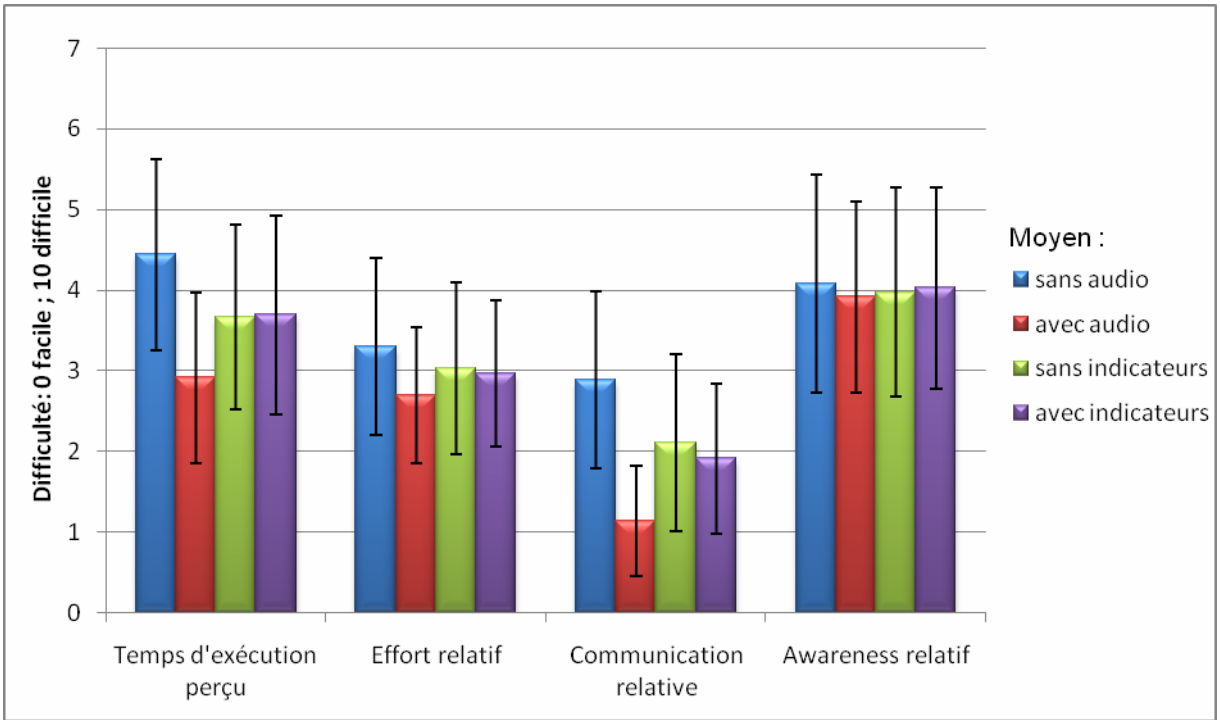

Figure 5.7 : Evaluation des composantes de l'intuitivité du questionnaire q_2 en fonction des moyens à disposition

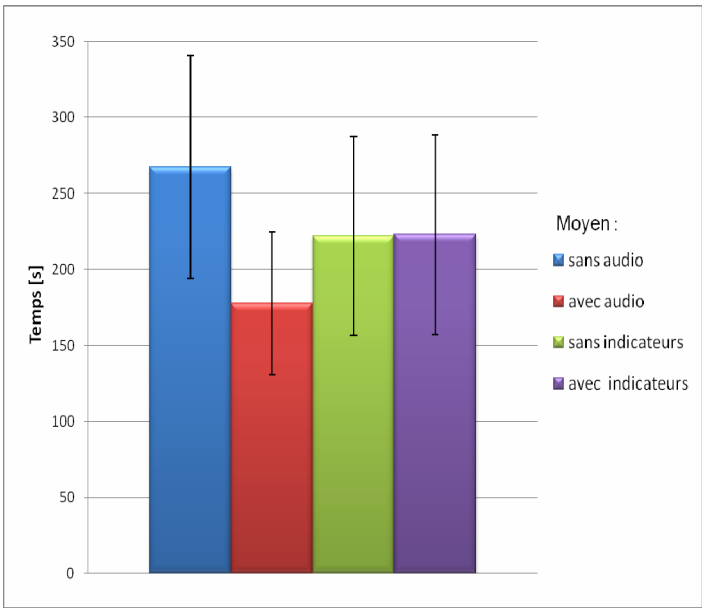

Figure 5.8 : Durée moyenne des parties en fonction des moyens à disposition

Le tableau 5.8 donne les résultats de l'analyse de la variance en fonction de la disponibilité de l'audio. Il en ressort que l'utilisation de l'audio a une influence significative sur le temps d'exécution de la partie (p=0.0066), sur la communication absolue (p<0.001) et sur la communication relative (p<0.001). Concernant la difficulté perçue et le temps d'exécution perçu, la différence n'est juste pas significative avec un p entre 0.05 et 0.06.

Le tableau 5.9 donne les résultats de l'analyse de la variance en fonction de la disponibilité des indicateurs. La disponibilité des indicateurs n'a pas d'influence significative sur la performance : la valeur de p la plus petite est en effet de 0.53.

Tableau 5.8 : Analyse de la variance en fonction de la disponibilité de l'audio

	Temps exécution partie	Difficulté perçue	Effort absolu	Concentration	Communication absolue	*Awareness* absolu	Temps d'exécut-ion perçu	Effort relatif	Communication relative	*Awareness* relatif
N	24	48	48	48	48	48	48	48	48	48
F	8.34	3.65	1.46	0.28	25.31	0.009	8.31	1.07	16.36	0.036
p	**0.0066**	0.06	0.23	0.59	**3.6E-06**	0.92	0.052	0.19	**0.000013**	0.85
Sign	**	°			***		°		***	

Tableau 5.9 : Analyse de la variance en fonction de la disponibilité des indicateurs

	Temps exécution partie	Difficulté perçue	Effort absolu	Concentration	Communication absolue	*Awareness* absolu	Temps d'exécut-ion perçu	Effort relatif	Communication relative	*Awareness* relatif
N	24	48	48	48	48	48	48	48	48	48
F	0.001	0.38	0.15	0.018	0.015	0.009	0.002	0.014	0.16	0.008
p	0.95	0.53	0.69	0.91	0.89	0.92	0.96	0.90	0.68	0.92

5.3.3 Analyse des résultats

Hypothèse 1 : l'audio réduit le temps d'exécution et améliore l'intuitivité

Notre première hypothèse H_1 énonce que l'audio avec le *chat* apporterait plus que le *chat* tout seul. Il s'avère que l'audio réduit effectivement le temps nécessaire pour accomplir la tâche (comme le montrent le tableau 5.8 et la figure 5.8, le temps d'exécution de la partie est réduit avec l'audio) et les participants trouvent que la communication se fait plus aisément s'ils disposent de l'audio (comme le montre le tableau 5.8 et la figure 5.7, les participants estiment que la communication relative et la communication absolue sont plus aisées avec l'audio). Par contre, les cinq autres composantes de l'intuitivité, c'est-à-dire la difficulté perçue, l'effort, la concentration, le temps d'exécution perçu et l'*awareness* ne sont pas influencées par la disponibilité de l'audio.

L'audio réduit donc le temps d'exécution et améliore l'intuitivité. Comme le *chat* est disponible avec l'audio, les participants profitent aussi des spécificités du *chat*, telles que la consultabilité et la révisabilité. On remarque que, même si le *chat* était disponible à côté de l'audio, la plupart des paires de participants l'ont utilisé uniquement pour transmettre les lettres découvertes et la combinaison finale.

Hypothèse 2 : les indicateurs réduisent le temps d'exécution et améliorent l'intuitivité

Nous avons formulé une deuxième hypothèse H_2 qui énonce que nos indicateurs permettent de réduire le temps d'exécution et d'améliorer l'intuitivité. Cette expérimentation ne montre pas d'effet significatif de la disponibilité des indicateurs (comme le précise le tableau 5.9). Néanmoins, nos observations ont montré que les participants ont utilisé les avatars pour se repérer dans le monde 3D et les pointeurs pour indiquer des endroits précis. Cela est vrai surtout pendant les premières parties. Par exemple, lorsqu'un participant peine à trouver un objet géométrique, son partenaire va lui indiquer une position proche de laquelle il doit chercher. Cela peut se faire en utilisant les médias de communications et les indicateurs. Les indicateurs sont donc réellement utiles dans certaines circonstances.

Participants

Le niveau des connaissances informatiques des participants était homogène et correspondait à celle des utilisateurs de la VSC. Ces derniers ont en effet déjà travaillé avec des ordinateurs, soit pour avoir fait des simulations et produit des données numériques, soit pour avoir utilisé l'un ou l'autre système de VS.

Plus les participants travaillent pour accomplir une tâche particulière et découvrir des *artefacts* particuliers, meilleurs vont être leurs résultats. Par contre, leur besoin d'avoir une présentation explicite des informations contribuant à l'*awareness* va diminuer. A titre d'exemple, certains participants utilisent, au fil de l'expérimentation, des descriptions de positions en mots de tous les jours : « Le cube rose se trouve proche des *montagnes*, vers l'origine ». Cela est très difficile à réaliser au début de l'expérimentation [55], mais facilite visiblement l'exécution de la tâche. Au bout de quelques parties, un participant connaît mieux le monde dans lequel il travaille et devient de plus en plus habile au jeu qui lui est proposé.

Le besoin d'utiliser des moyens contribuant à l'*awareness* est plus grand pour les participants qui ne sont pas experts du domaine ou qui ne sont pas familiarisés avec le monde 3D dans lequel ils travaillent. Par exemple, en VSC, un océanographe qui connaît bien la région dont il

s'occupe et sait où chercher les phénomènes intéressants expliquera : « regarde ici, à l'embouchure du Rhône ». En généralisant cela, l'importance de l'*awareness* est inversement proportionnelle à la connaissance.

Tâche proposée

Pour accomplir la tâche qui leur est proposée, les participants doivent explorer le monde 3D, se créer une représentation spatiale et acquérir de nouvelles connaissances. Ensuite, les participants cherchent à échanger les connaissances fraîchement acquises.

La tâche proposée est élémentaire et a peu de chance de se reproduire aussi simplement dans la VSC. Pour accomplir cette tâche, les participants doivent cependant faire appel à des interactions élémentaires qui, elles, sont représentatives de ce que l'on peut observer dans de nombreuses situations plus réalistes. En outre les différentes actions sont, elles aussi, représentatives : se déplacer pour se rapprocher d'un participant, indiquer où se situe un objet ou préciser sa propre position.

5.3.4 Commentaires des participants

Les commentaires, exprimés librement par les participants à la fin de l'expérimentation, permettent de mettre en évidence les qualités et les défauts du système.

Une remarque, faite par plus d'un quart des participants, concerne le manque d'intuitivité de la navigation dans le monde 3D. Le monde dans lequel se déroule la tâche ne possède pas d'orientation (haut-bas, droite-gauche) et il est donc difficile de se faire une représentation spatiale de ce monde.

Il manque un mécanisme permettant de se connecter directement à la vue d'un participant sans devoir chercher son avatar.

Les pointeurs sont peu utilisés pour accomplir la tâche proposée. Cela est dû à une mauvaise utilisation des pointeurs, que les participants ne placent pas correctement dans le monde 3D, mais quelque part entre l'endroit à désigner et leur œil, sans prendre la peine de vérifier si la profondeur est correcte.

Les participants préfèrent avoir à disposition l'audio que les indicateurs. Cela est confirmé par les résultats de notre expérimentation ; en effet la mise à disposition de l'audio permet d'améliorer l'intuitivité alors que les indicateurs ne l'influencent pas.

La tâche comporte un aspect ludique que de nombreux participants ont manifestement apprécié. Cela s'illustre par les messages de félicitations échangés à la fin d'une partie victorieuse ou par les discussions sur la stratégie à adopter.

5.3.5 Observations

En analysant les actions des participants et leurs échanges de messages par *chat* ou par audio, nous nous rendons compte que les participants ont appliqué différentes stratégies et qu'ils ont dû faire face à divers problèmes.

Détection des malentendus

La lettre attachée à l'objet géométrique permet à la paire de participants de vérifier s'ils considèrent le même objet. En cas de malentendu, les deux participants vont vérifier s'ils ont choisi le bon objet géométrique. Le participant ayant trouvé le bon objet avait tendance à utiliser son avatar et son pointeur pour en indiquer la position et pour aider ainsi son partenaire à se repérer dans le monde 3D. Ce phénomène s'est produit assez souvent (7 fois) et implique une perte de temps. A titre d'exemple, la figure 5.9 présente les messages échangés par les participants lors de la constatation d'un tel malentendu.

```
PartA > cone U
PartB > brun U
PartA > du meme cote qu'avant
PartA > tu zoomes en arrieres
PartB > rose F
PartA > ??? cone R
PartA > c pas normal ;-)
PartB > on reviens au cone brun
PartA > oui
PartA > je le vois du cote de depart en haut a droite
PartA > reinit ta vue si tu es pas la
PartA > regarde ou je suis
PartA > c bon?
PartB > vert T ?
PartB > non !
PartA > cone R
PartB > rouge R !
PartA > merci
PartB > jaune U
PartA > cylindre U
```

Figure 5.9 : Texte tiré du *chat* ; les participants ont trouvé un malentendu

Erreurs

Dans quatre parties, les participants ont donné une combinaison erronée. Les paires en question n'avaient pas contrôlé s'ils avaient découvert la même lettre de la combinaison, ce qui a conduit à une erreur dans le choix de l'objet géométrique suivant. Dans deux cas, c'était la dernière lettre de la combinaison qui était erronée. Les paires voulaient gagner du temps à la fin de la partie et ont omis de contrôler si la dernière lettre trouvée correspondait à celle de leur partenaire.

Texte du *chat*

Si les participants n'échangent que les informations importantes, taper le texte sur le *chat* demande peu de temps, ce qui est le cas dans les dialogues de la figure 5.9 et de la figure 5.10. Les participants avaient tendance à ne pas écrire les accents ou à utiliser des abréviations afin d'écrire plus rapidement. Ce phénomène se retrouve dans d'autres domaines, comme dans l'écriture des SMS. Pour contrôler que l'objet géométrique sélectionné est correct, une stratégie particulièrement efficace consiste à transmettre uniquement l'information (couleur ou forme) que l'on est seul à détenir et la lettre. Les participants mettent ainsi en pratique le principe du moindre effort collaboratif décrit par Clark [19].

```
admin > 1er objet: cylindre jaune
PartB > sphere C
PartA > Jaune C
PartA > ---
PartA > vert E
PartB > cube E
PartA > ---
PartA > Jaune P
PartB > Cone P
PartB > Cylindre E
PartA > rose E
PartA > --
PartA > rose B
PartB > cube B
PartA > CODE CEPEBA
Admin > Gagne
```

Figure 5.10 : Texte tiré du *chat* ; les participants n'indiquent que l'information essentielle

Grounding

Pendant une partie, les participants acquièrent de nouvelles connaissances (position des objets, endroit permettant de voir plus d'objets, etc.). L'objectif étant de finir une partie le plus vite possible, ils n'avaient souvent pas le temps d'échanger ces connaissances. En revanche, juste avant le début ou juste après la fin d'une partie, les participants ont procédé à ces échanges afin d'avoir une même base commune d'informations et ainsi pouvoir développer une meilleure stratégie lors de la partie suivante. Il s'agit d'un processus de *grounding*.

Informations liées au monde 3D

Durant les premières parties, les participants donnaient de nombreuses informations sur leur position ainsi que sur celle des objets. Avec l'expérience, ils tendaient à échanger uniquement l'information indispensable. Par exemple, les informations de vérification telles que « ok, on a la même lettre » disparaissaient. La partie du *grounding*, consistant à confirmer que les participants ont bien le même objet géométrique, tendait à disparaître.

Stratégies

Durant les premières parties, un participant ayant trouvé un objet géométrique avait tendance à indiquer sa position (que se soit par le *chat*, l'audio ou en utilisant son avatar et son pointeur) pour que son partenaire découvre plus rapidement l'objet. Dans les parties suivantes, cette technique tendait à disparaître.

Avec l'expérience, les participants avaient tendance à se placer de manière à voir le plus d'objets possible et ainsi à limiter les déplacements. Ils demandaient moins souvent à leur partenaire de les suivre grâce à leur avatar. Ainsi le besoin d'utiliser les moyens mis à disposition diminue avec la connaissance.

Conclusion des observations

Nous observons que l'importance du *grounding* et de l'*awareness* diminue avec la connaissance. Nous remarquons que les participants utilisent le média de communication qui demande le moins d'effort collaboratif possible. Ces phénomènes ne se produisent pas uniquement dans la VSC, mais aussi dans d'autres domaines.

5.3.6 Questionnaire ayant servi à l'évaluation du système

A la fin des quatre parties, les participants ont rempli le questionnaire q_3 qui évalue différents aspects de la facilité d'utilisation du système. Les résultats sont présentés à la figure 5.11.

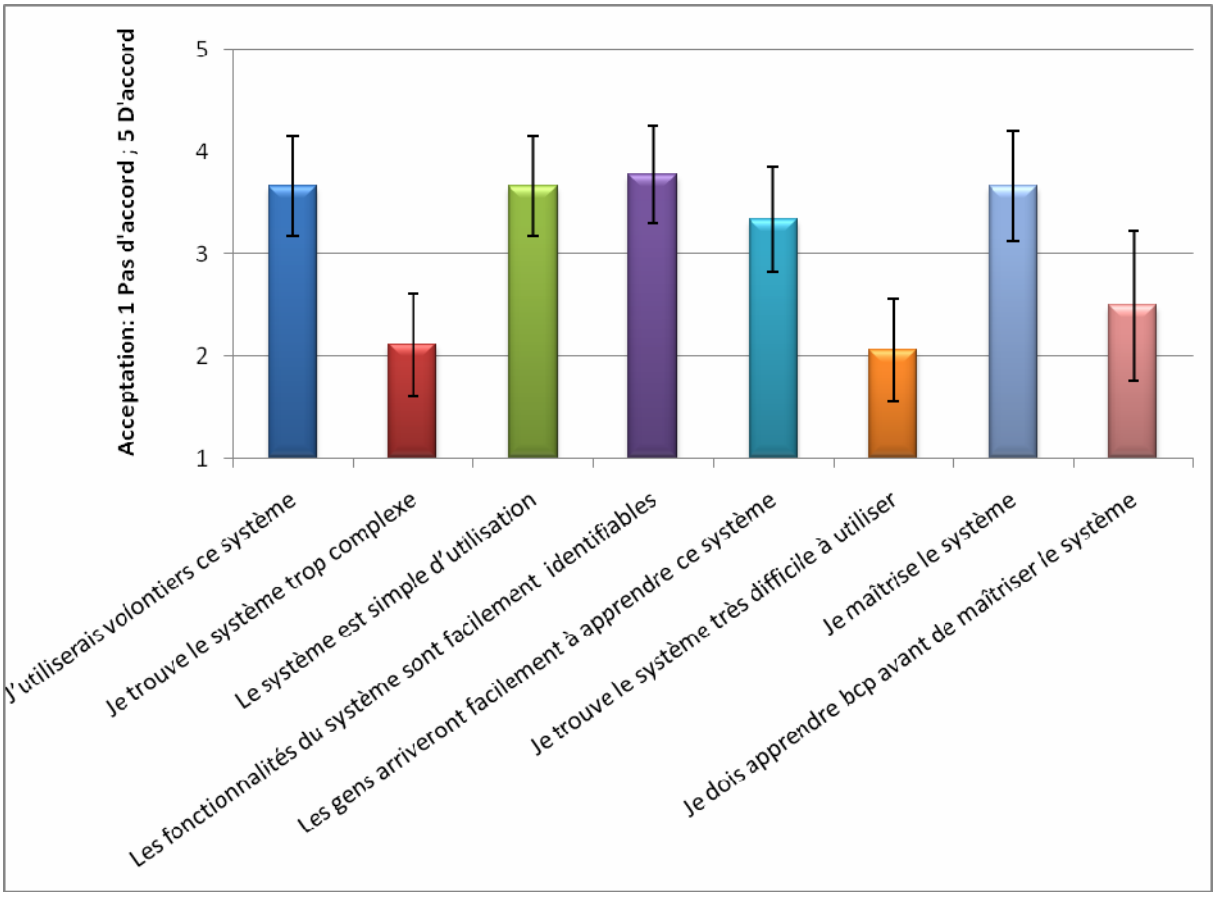

Figure 5.11 : Résultat du questionnaire q_3

Nous nous rendons compte que les participants ont trouvé le système simple d'utilisation. Etant donné que seule une petite partie des fonctionnalités du système de VSC ont été utilisée, nous ne prétendons pas que ces résultats puissent être généralisés, mais ils permettent au minimum de conclure que notre système est facile d'utilisation pour accomplir cette tâche rudimentaire.

5.4 Autres évaluations

Une autre manière de démontrer l'utilisabilité d'un système consiste à participer à des concours. Notre système, ZoomIn, a été sélectionné pour participer au concours EASA 2004 [27] (*European Academic Software Award* – Concours des logiciels académiques européens) et a été classé dans les 30 meilleurs logiciels académiques européens (158 logiciels ont participé à ce concours).

Il est aussi possible de faire tester le système par des personnes étrangères au projet. Un test indépendant de notre système (sans aucune demande préalable de notre part) a été effectué par Milaho [65]. Il révèle la facilité d'utilisation du système ainsi que son intérêt à l'appliquer à différents domaines. Comme défaut, il cite la difficulté à installer le système et à intégrer de nouvelles données au système.

5.5 Conclusion

Notre expérimentation montre l'intérêt et le gain engendrés par l'audio pour collaborer. Toutefois, nous pensons que le *chat* doit toujours être disponible. En effet, comparé à l'audio, le *chat* offre la possibilité de consulter ce qui a été envoyé, ne nécessite pas de matériel spécial et il est mieux adapté aux utilisateurs qui ne maîtrisent pas bien la langue.

L'expérimentation a permis de déceler et de corriger des défauts du système. Par exemple, nous avons ajouté un mécanisme permettant de se connecter directement à la vue d'un autre participant sans devoir chercher son avatar.

Nous avons effectué cette expérimentation avec des paires de participants. Il serait intéressant d'examiner comment plus de deux participants collaborent, et surtout d'examiner si, dans cette situation, l'utilisation des médias de communication sera la même (par exemple, avec l'audio, une seule personne peut parler à la fois). En outre, il serait intéressant d'évaluer l'apport de la vidéo ; la vidéo prend beaucoup de place sur l'écran, mais elle apporte des informations sur la présence, sur les activités, sur les gestes ou les émotions des participants.

Notre expérimentation a permis de montrer quel était l'apport de l'audio. Par contre, l'ajout de deux de nos indicateurs (les avatars et les pointeurs) censés contribuer à l'*awareness* n'a pas montré d'effet significatif. Malgré tout, il existe des situations particulières, comme lorsqu'un participant peine à trouver un objet graphique, dans lesquelles ces indicateurs sont utiles. Les corrections apportées à la réalisation des indicateurs sur la base des commentaires des participants devraient permettre d'en faciliter l'utilisation.

En résumé, nous croyons, qu'au pire des cas, fournir de l'information d'*awareness* grâce à ces indicateurs ne va pas pénaliser les participants et qu'au mieux les performances ainsi que la satisfaction du système seront améliorées.

Chapitre 6 ZoomIn comme prototype

Ce chapitre traite de l'architecture de notre prototype, basé sur ZoomIn, un système de visualisation scientifique initialement développé comme un système mono-utilisateur. Nous allons d'abord brièvement en présenter l'architecture. Par la suite, nous verrons les différentes modifications apportées afin de pouvoir obtenir un système collaboratif ainsi que les difficultés qu'il a fallu surmonter.

6.1 Architecture de la version mono-utilisateur

6.1.1 Historique

ZoomIn est un système de visualisation scientifique conçu par Hervé Sanglard dans le cadre de son doctorat [86]. L'un de ses objectifs consistait à réaliser un système extensible et simple d'utilisation. Le système est essentiellement prévu pour visualiser des champs vectoriels et scalaires. Il a d'abord été réalisé sur une plateforme Silicon Graphics (SGI) et ensuite porté sur Windows. D'un point de vue technique, il est programmé en C++ et utilise la librairie graphique *OpenInventor* [107]. Il ne contient aucune fonctionnalité collaborative. Dans le cadre d'un travail de diplôme, il a en revanche été tenté de répartir les différentes fonctions sur les ordinateurs d'un réseau local, un essai qui n'a pas été très concluant [5].

6.1.2 Architecture de base

Dans ZoomIn, l'accès aux données se fait comme indiqué dans la figure 6.1. Le pilote de données permet d'accéder aux données brutes. ZoomIn demande au pilote de données les valeurs à certains points définis par leurs coordonnées. Ce pilote est chargé d'effectuer les calculs et de retourner le résultat.

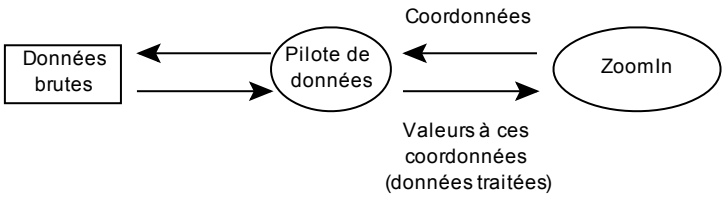

Figure 6.1 : Accès aux données dans ZoomIn

Comme l'illustre la figure 6.2, un utilisateur-développeur peut ajouter à ZoomIn des fonctionnalités (appelées *plugins*). Cela peut se faire dans deux contextes :

1) On peut créer un nouveau pilote de données pour accéder à un nouveau format de données brutes. Dans la pratique, la plupart des utilisateurs possède des données brutes selon un format particulier, ce qui signifie qu'il faut réaliser un pilote de données pour chaque nouveau format.

2) On peut créer un nouvel outil de visualisation capable de construire des objets graphiques représentant certains aspects des données brutes.

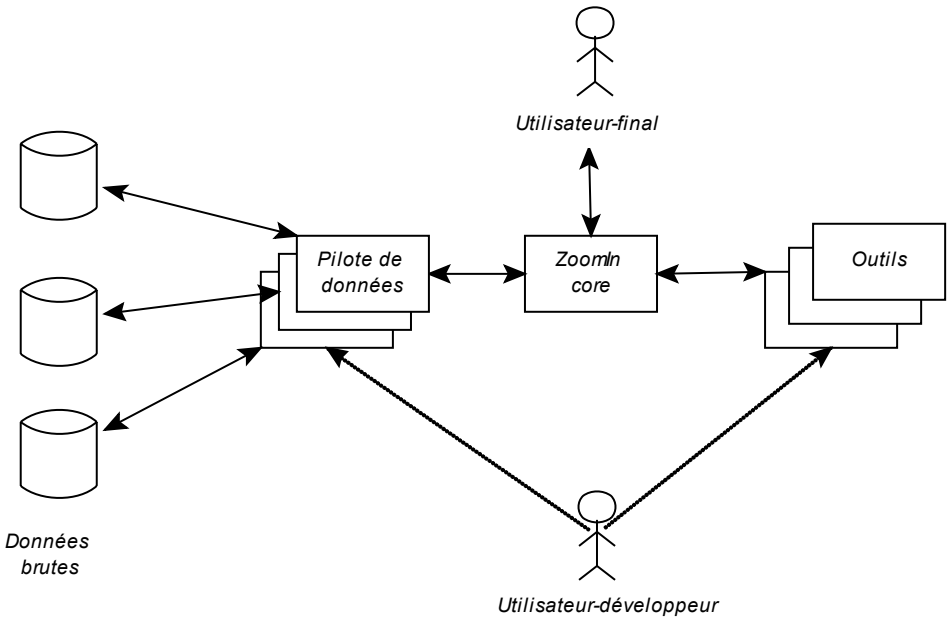

Figure 6.2 : Architecture globale de ZoomIn

Afin de faciliter le travail des utilisateurs, ZoomIn met à disposition des pilotes de données génériques ainsi qu'un certain nombre d'outils de visualisation standards [44], tels que Isosurface, *Arrows* ou Sonde.

6.1.3 Avantages et désavantages

Par rapport à d'autres systèmes du même genre, un des avantages de ZoomIn réside dans sa simplicité d'utilisation, ainsi que dans la disponibilité de son code source. Le système offre un bon compromis entre l'extensibilité et la facilité d'utilisation (section 2.1.2). Un des désavantages de ce système est son manque de portabilité (C++ n'est pas un langage particulièrement portable). Il n'a pas été conçu pour le travail collaboratif.

6.1.4 Choix de ce système

Au vu des avantages du système ZoomIn, nous avons décidé de nous baser sur lui pour réaliser notre prototype [15].

Une autre approche aurait consisté à utiliser un *framework* collaboratif comme décrit à la section 2.2.6. Mais en raison du manque d'extensibilité de tels *frameworks* et parce que la majorité d'entre eux ont des défauts conceptuels ou techniques [34], la réalisation d'un système de visualisation scientifique efficace serait très ardue.

6.2 Architecture étendue à la collaboration

L'ajout de fonctionnalités collaboratives à un système existant soulève cependant des difficultés techniques que nous allons passer en revue.

6.2.1 Informations à transmettre

La collaboration nécessite des échanges d'information entre les participants. Afin de bien structurer cet échange, nous proposons une classification hiérarchique des différents types d'informations pertinentes de notre système. La figure 6.3 présente cette classification.

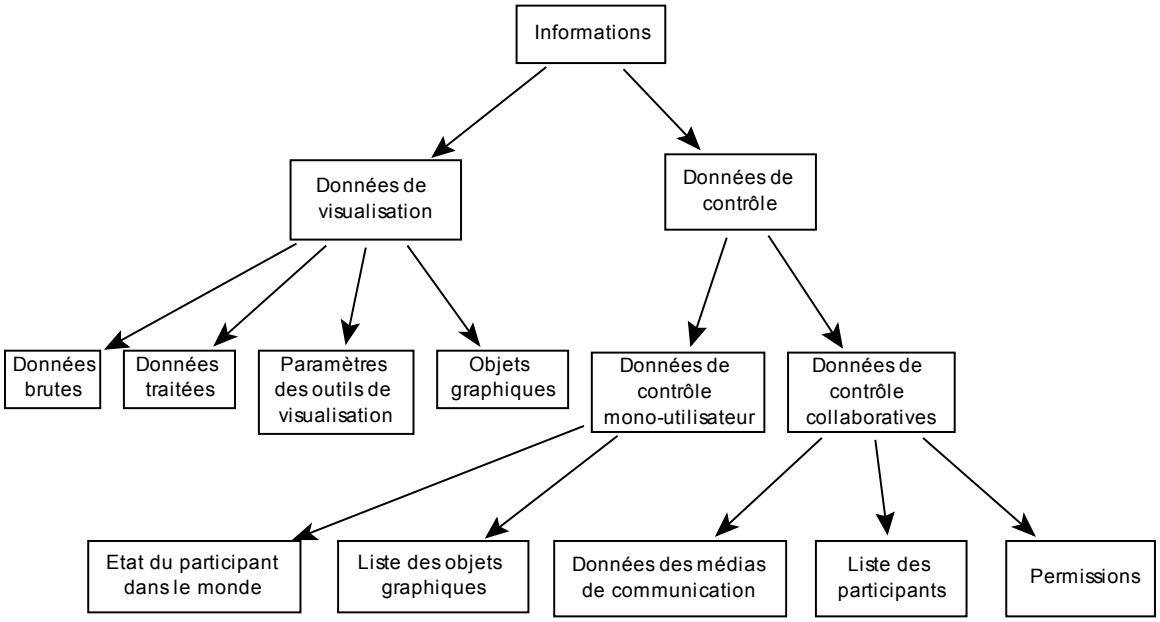

Figure 6.3 : Types d'information

Nous expliquons les termes qui apparaissent dans cette classification et qui n'ont pas encore été définis dans les sections précédentes :

Informations : Ce terme comprend toutes les données, qu'elles soient fournies par l'utilisateur (par exemple les données brutes ou les messages du *chat*) ou produites par le système (par exemple les données traitées ou les objets graphiques).

Données de visualisation : Informations dont le traitement permet de créer des objets graphiques.

Données de contrôle : Données internes du système qui sont nécessaires à son fonctionnement. On distingue les données de contrôle **mono-utilisateur** (nécessaire tant dans un environnement mono-utilisateur que dans un environnement collaboratif) et **collaboratives** (nécessaire uniquement dans un environnement collaboratif).

Etat du participant dans le monde : Informations concernant un participant et le monde 3D, comme sa position actuelle dans le monde 3D ou le pas de temps auquel il se trouve.

Données des médias de communication : Données transmises d'un participant à un autre à travers un média de communication tel qu'un *chat*, un téléphone ou une transmission vidéo.

Permissions : Les permissions régissent l'accès aux informations par les différents participants.

6.2.2 Propriétés des informations à transmettre

Le tableau 6.1 énumère les principales propriétés de chaque type d'information.

Tableau 6.1 : Propriétés des différentes informations

Type d'information	Inclus dans version mono-utilisateur	Partage nécessaire dans un système collaboratif	Estimation du volume	Fréquence de la mise à jour de l'information
Données brutes	Oui	Oui*	Grand	Jamais
Données traitées	Oui		Petit à grand	Jamais (dépend de la programmation)
Objets graphiques	Oui		Petit à grand	Rarement
Paramètre des outils de visualisation	Oui		Petit	Rarement
Etat du participant dans le monde	Oui	Oui	Petit	Souvent
Données du média de communication	Non	Oui	Petit à grand	Souvent
Liste des objets graphiques	Oui	Oui	Petit	Rarement
Permissions	Non	Oui	Petit	Rarement

* = au moins une de ces quatre informations doit être transmise.

Afin de donner un ordre de grandeur, nous considérons qu'un volume « petit » correspond à moins de 50Ko et qu'un volume « grand » à plus de 1 Mo. Une mise à jour est « peu fréquente » si elle a lieu moins d'une fois par minute ; elle est « fréquente » si elle a lieu plus d'une fois par seconde.

A la lecture de ce tableau, on s'aperçoit que les besoins pour transmettre les informations sont différents d'un type d'information à l'autre (par exemple pour des objets graphiques, la description des participants ou des permissions). Une transmission efficace des informations requiert donc différentes techniques, suivant le type d'information :

- Pour la transmission des données de visualisation, un système de *broadcast* est utilisé.
- Pour la transmission des données de contrôle qui sont rarement mises à jour, un système de **mémoire partagée** [13] est employé. Tous les participants ont donc accès aux mêmes informations. Les accès en lecture se font rapidement, alors que les accès en écriture sont nettement plus lents.
- Pour la transmission des données de contrôle qui sont mises à jour souvent (comme la position des participants), un protocole dédié [50] est utilisé.
- Pour la transmission des données des médias de communication, on fait appel à l'une des techniques précédentes (par exemple la mémoire partagée pour le *chat*) ou à un système annexe (par exemple le téléphone pour l'audio).

6.3 Fonctionnalités nouvelles liées à collaboration

6.3.1 Connexion initiale

Pour un participant, la première connexion au système doit être aussi intuitive que possible. C'est particulièrement vrai dans le cas d'un mode maître-élève, où l'élève ne disposant pas d'assistance doit pouvoir se connecter aisément.

Lors de sa première connexion, un participant doit indiquer son nom et choisir une couleur qui l'identifie dans la représentation du monde 3D (son avatar et son pointeur auront cette couleur). Cela aide à répondre à la question « Qui ? » de l'*awareness* décrit dans la section 2.2.3.

Deux types de connexions sont possibles :

1) Si le participant désire créer une nouvelle session de visualisation, il crée un **coordinateur** (serveur de connexion). Tous les participants qui désirent participer à sa session devront se connecter à ce coordinateur.

2) Si le participant désire se connecter à une session existante, il se connectera à un coordinateur. Dans notre prototype, cela se fait encore en indiquant l'adresse IP d'un coordinateur.

6.3.2 Déplacement dans le monde 3D

Les avatars et les pointeurs aident à répondre à la question « Où ? » de l'*awareness*. Les déplacements dans le monde 3D se font grâce à la souris.

Un bouton *bookmark* permet à un participant de marquer sa position actuelle, ce qui lui permet d'y revenir plus tard. Au cas où un participant est perdu dans le monde 3D, un bouton *reset* permet de revenir à l'origine.

6.3.3 Interface utilisateur - Monde public et monde privé

La figure 6.4 représente une copie d'écran du système pendant une session de visualisation. La partie supérieure contient les menus et les contrôles permettant de gérer les pas de temps et d'effectuer les actions (créer, mettre à jour, effacer, cacher, montrer des objets graphiques).

La partie gauche contient les paramètres de l'outil de visualisation en cours d'utilisation. Ces paramètres diffèrent d'un outil de visualisation à un autre : par exemple l'outil Isosurface n'accepte qu'une ROI (*Region Of Interest*) de type « parallélépipède rectangle ».

La partie inférieure contient le *chat* qui est le média de communication par défaut. La partie centrale contient les mondes partagé et privé ainsi que les contrôles de déplacement dans le monde 3D. Le titre indiqué sur la fenêtre permet d'identifier s'il s'agit d'un monde partagé ou privé.

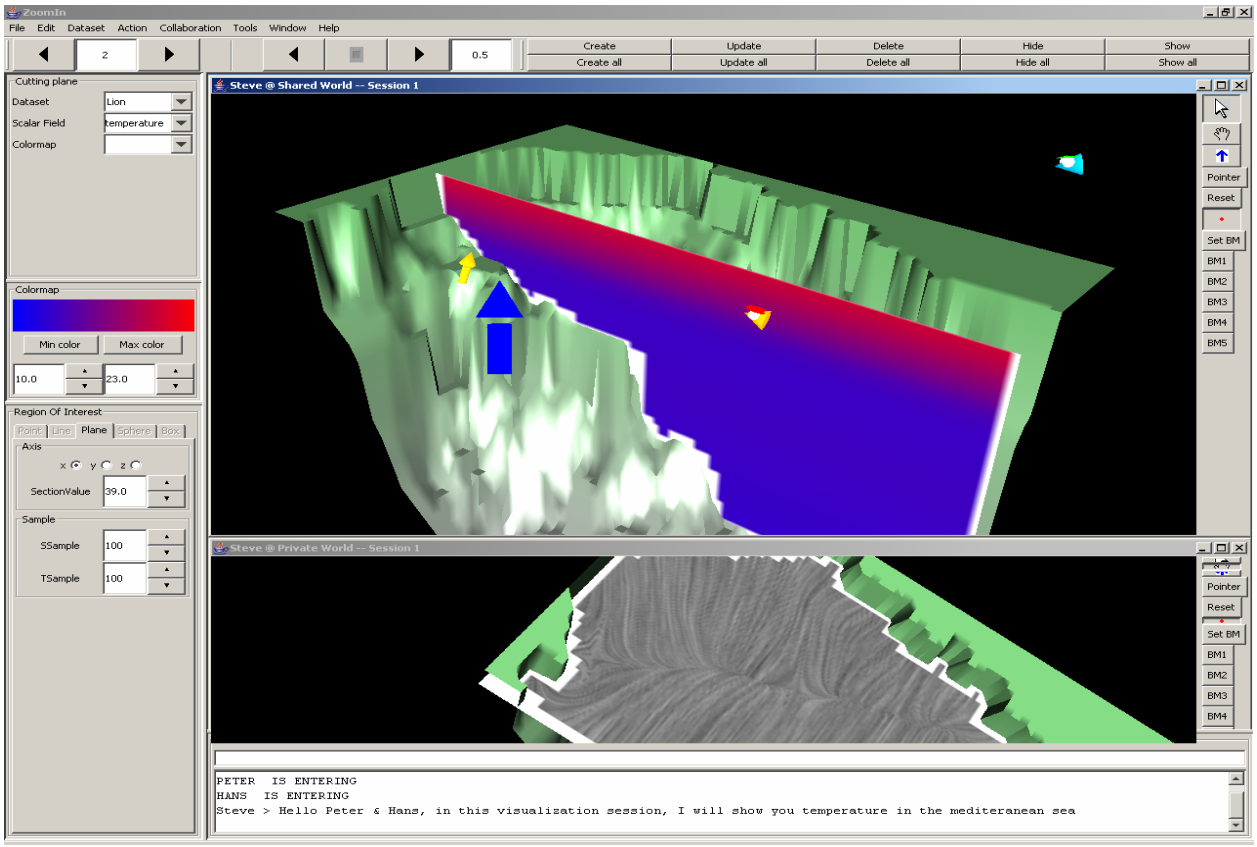

Figure 6.4 : Interface de ZoomIn

6.3.4 Permissions

Une permission spécifie si un participant peut, ou ne peut pas, effectuer une opération. Elle se configure grâce à deux options pour chaque participant, comme illustré par la figure 6.5.

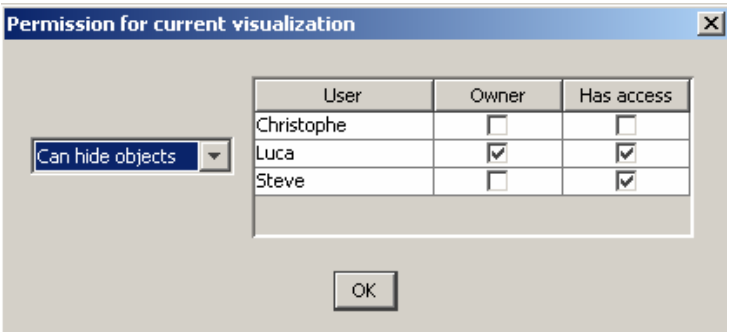

Figure 6.5 : Exemple de permission dans ZoomIn. Il y a ici trois participants : Luca est propriétaire, Steve peut effectuer l'opération et Christophe ne le peut pas

La case à cocher à droite (*Has access*) indique qui peut effectuer l'opération. La case à cocher à gauche (*Owner*) indique le propriétaire de la permission (c'est-à-dire qui a le droit de modifier les permissions des participants). Naturellement, il faut toujours au minimum un propriétaire par permission. Le fait d'être le propriétaire implique de pouvoir effectuer l'opération. Un propriétaire peut céder sa propriété à un autre participant

Si l'on attachait les permissions à toutes les opérations accessibles aux utilisateurs, du fait de leur nombre, leur gestion serait difficile. Pour simplifier la gestion, nous les avons attachées à trois « opérations de base » : création d'objets graphiques, modification du pas de temps courant, masquage des objets graphiques.

Comme indiqué à la section 3.3, la configuration des modes de travail se fait en fixant les permissions. Pour obtenir le mode libre, il suffit d'autoriser les participants à effectuer toutes les opérations, alors qu'il faut être plus restrictif pour obtenir un mode de présentation.

6.4 Cohérence des actions sur le monde 3D

6.4.1 Actions

Une action est un ordre donné par un participant pour agir sur des objets graphiques du monde 3D. Les participants interagissent avec le monde 3D uniquement à travers des actions.

Il y a trois actions principales :

- Créer : Cette action permet de créer un nouvel objet graphique. Le participant fournit les paramètres, puis l'outil de visualisation se charge de construire l'objet graphique. Son exécution peut demander beaucoup de temps.

- Effacer : Cette action permet d'effacer un objet graphique. Son exécution demande peu de temps ; en effet, d'un point de vue technique, il s'agit simplement d'enlever un élément de l'arbre représentant le monde 3D.

- Modifier : Cette action permet de modifier un objet graphique existant. Le participant sélectionne l'objet graphique et en modifie les paramètres. Nous avons choisi de réaliser la modification d'un objet graphique par un effacement de ce dernier, suivi de la création d'un nouvel objet graphique. Son exécution peut demander beaucoup de temps.

Il y a en outre deux actions secondaires :

- Cacher : Cette action permet de cacher un objet graphique existant.

- Montrer : Cette action permet de montrer tous les objets graphiques.

L'exécution de ces deux actions demande peu de temps ; en effet d'un point de vue technique, il s'agit d'ajouter ou d'enlever des objets graphiques existants à l'arbre représentant le monde 3D.

Ces actions étant susceptibles d'être effectuées par tous les participants, il est important d'assurer la cohérence du monde 3D.

6.4.2 Cohérence du monde 3D

Nous définissons la **cohérence** du monde 3D comme suit : la séquence des actions effectuées est identique pour tous les participants.

Pour être cohérent, il faut que les actions parviennent dans le même ordre à tous les ordinateurs des participants. Afin de résoudre ce problème, le **coordinateur** évoqué dans la section 6.3.1 gère les actions. Ce coordinateur peut être considéré comme un serveur central, chargé d'assurer la **validité**, **l'ordre** et **l'exécution** des actions. Ce choix contraste avec la transmission des données traitées dans laquelle il n'y a pas besoin de coordinateur, chacun s'occupant de prendre les données traitées là où il le souhaite.

Les actions à effectuer doivent être **valides**. Si, par exemple, deux actions d'effacement d'un même objet graphique sont demandées plus ou moins au même moment, la deuxième sera invalide.

Le coordinateur gère **l'ordre** des actions. On peut distinguer deux phases : la réception de la demande d'action par le coordinateur et la transmission de cette demande d'action aux destinataires, qui devront l'exécuter.

Comme représenté à la figure 6.6, toutes les demandes d'action provenant d'un participant sont transmises au coordinateur. Le coordinateur est chargé de **transmettre** la demande d'action à tous les destinataires. A la réception d'une demande, le coordinateur la transmet à tous les ordinateurs des participants. Ensuite, chacun envoie une quittance au coordinateur lorsqu'il a fini d'exécuter l'action. Si une action est encore en cours d'exécution (c'est-à-dire que toutes les quittances n'ont pas encore été reçues) et qu'une nouvelle demande d'action est transmise au coordinateur, deux stratégies existent :

1) Les demandes sont mémorisées. Quand une demande arrive au coordinateur, elle est stockée suivant un ordre FIFO (*First In-First Out*). Dans ce cas, il faut veiller à la validité des demandes stockées et, dans un souci d'intuitivité, informer l'expéditeur si la demande d'action n'est pas satisfaite immédiatement.

2) Les demandes ne sont pas mémorisées. Quand la demande arrive au coordinateur, celui-ci la renvoie à l'expéditeur en lui signalant qu'elle ne peut pas être satisfaite pour le moment.

Nous avons réalisé la première stratégie. Tous les participants effectuent les mêmes actions valides dans la même séquence.

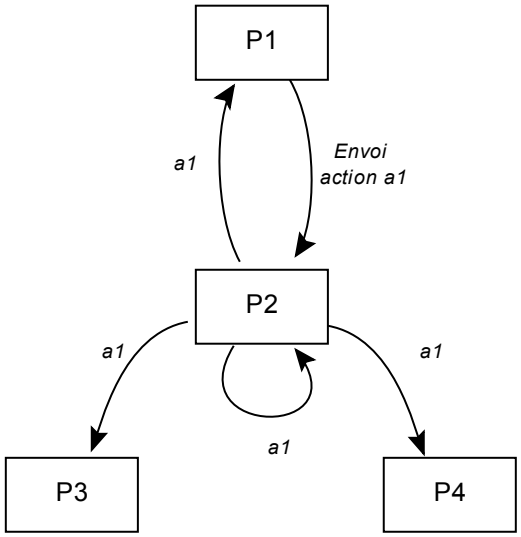

Figure 6.6 : Exécution d'une action *a1*. Premièrement, le coordinateur (ici P2) reçoit cette action, puis la transmet à tous les participants

6.4.3 Stratégies d'ordonnancement des actions

Pour réaliser l'ordonnancement des actions, il faut choisir une stratégie pour transmettre les actions. Ces stratégies influencent le temps d'attente et la satisfaction perçue par les

participants. Dans cette section, le temps est vu comme perçu par les participants et non à l'échelle d'un ordinateur.

Deux stratégies pour l'ordonnancement des actions sont discutées :

1) avoir un système avec ou sans attente d'exécution d'une action,

2) avoir un système bloquant ou non bloquant.

Système avec ou sans attente d'exécution d'une action

On peut attendre ou non qu'une action soit exécutée par tous les participants avant d'en effectuer une autre. La figure 6.7 et la figure 6.8 représentent l'exécution de deux actions respectivement avec et sans attente.

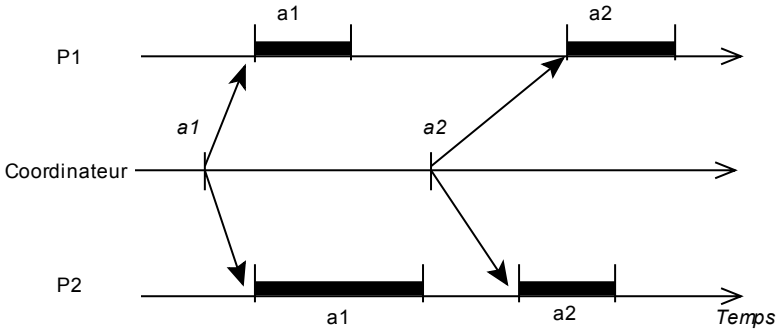

Figure 6.7 : Exécution des actions *a1* et *a2* avec attente

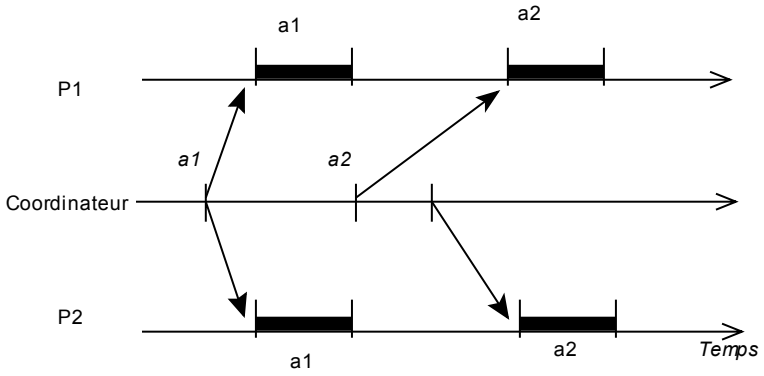

Figure 6.8 : Exécution des actions *a1* et *a2* sans attente

Le fait d'attendre qu'une action soit effectuée par tous les participants implique qu'à un temps donné, tous les participants ont les mêmes objets graphiques. En contrepartie, le temps d'attente augmente, car le plus lent participant ralentit tout le monde. En effet, les participants ayant déjà fini d'effectuer une action et voulant en effectuer une autre doivent attendre. Concernant la satisfaction, il faut remarquer qu'il est toujours pénible de devoir attendre pour effectuer une action.

Système bloquant ou non bloquant

Pendant qu'une action est en train d'être effectuée, il est possible d'interdire aux participants d'effectuer d'autres actions ; dans ce cas, nous parlons d'un **système bloquant**. Evidemment, les participants ont toujours la possibilité de se mouvoir dans le monde 3D. Dans le cas où il est possible d'effectuer une action même si une action est en cours d'exécution, nous parlons d'un **système non bloquant**.

Un système bloquant implique une perte d'interactivité ; en effet un participant ne peut pas toujours effectuer des actions. En contrepartie, le monde 3D est cohérent, donc il n'y a pas de risques d'effectuer une action invalide. Aucun participant n'est privilégié, en effet chacun peut effectuer des actions selon son désir.

Dans le cas d'un système non bloquant, il est judicieux de limiter le nombre d'actions stockées afin d'éviter un temps d'attente trop grand ; par exemple dans le cas où un participant clique plusieurs fois sur un bouton pour effectuer une action.

Analyse des stratégies

Le cas sans attente et non bloquant est celui dont le temps d'attente est le moindre. Dans le cadre de démonstrations, comme une seule personne à la fois interagit avec le système, le fait d'avoir un système bloquant ou non bloquant a peu d'influence sur la satisfaction. Par contre, le fait d'avoir un système avec attente semble judicieux, en effet il faut être sûr que tous les participants voient la même chose sur leur écran avant de procéder à des commentaires. Dans le cas contraire, certains participants risquent d'être déstabilisés en ne voyant pas s'afficher les objets graphiques immédiatement.

En comparant les choix présentés, il n'y a malheureusement pas de meilleure solution, il faut trouver le moins mauvais compromis en fonction de l'utilisation désirée. Dans notre cas, nous avons choisi d'avoir un système avec attente mais non bloquant. L'attente permet d'assurer, qu'à un temps donné, tous les participants auront les mêmes objets graphiques tandis que le système non bloquant permet d'effectuer d'autres actions.

6.5 Détails techniques

Pour résoudre le problème de la portabilité, ZoomIn a été récrit en Java en faisant usage de la librairie graphique Java3D [93], (OpenInventor n'existant pas dans ce langage) et du *middleware* RMI [96]. L'implémentation des fonctionnalités collaboratives est basée sur deux types de protocole de réseau :

1) Client-serveur pour accéder à un pilote de données.

2) ZiMulticast [13] pour gérer la mémoire partagée.

L'architecture logicielle est présentée dans la figure 6.9. Les principaux modules du système peuvent être décrits comme suit :

- *Dataset manager* : Ce module gère l'accès aux pilotes de données, qu'ils soient situés sur le même ordinateur ou sur celui d'un autre participant. Un participant peut choisir s'il désire rendre accessible à tous un pilote de données (et donc indirectement les données brutes liées à ce pilote). Une approche client-serveur est utilisée pour la transmission des informations liées à un pilote de données.

- *Collaboration manager* : Ce module gère les différentes demandes liées à la collaboration, par exemple la liste des permissions ou des participants. Il utilise les informations déposées dans la *Replicated Memory*.

- *Replicated memory* : Ce module réalise une mémoire commune au service des *Collaboration manager* des différents ordinateurs du système. Ce module garantit la cohérence des données dans le système réparti, donc qu'à tout instant, les différents *Collaboration manager* ont accès aux mêmes données. Cette mémoire est efficace uniquement s'il y a peu d'accès en écriture.

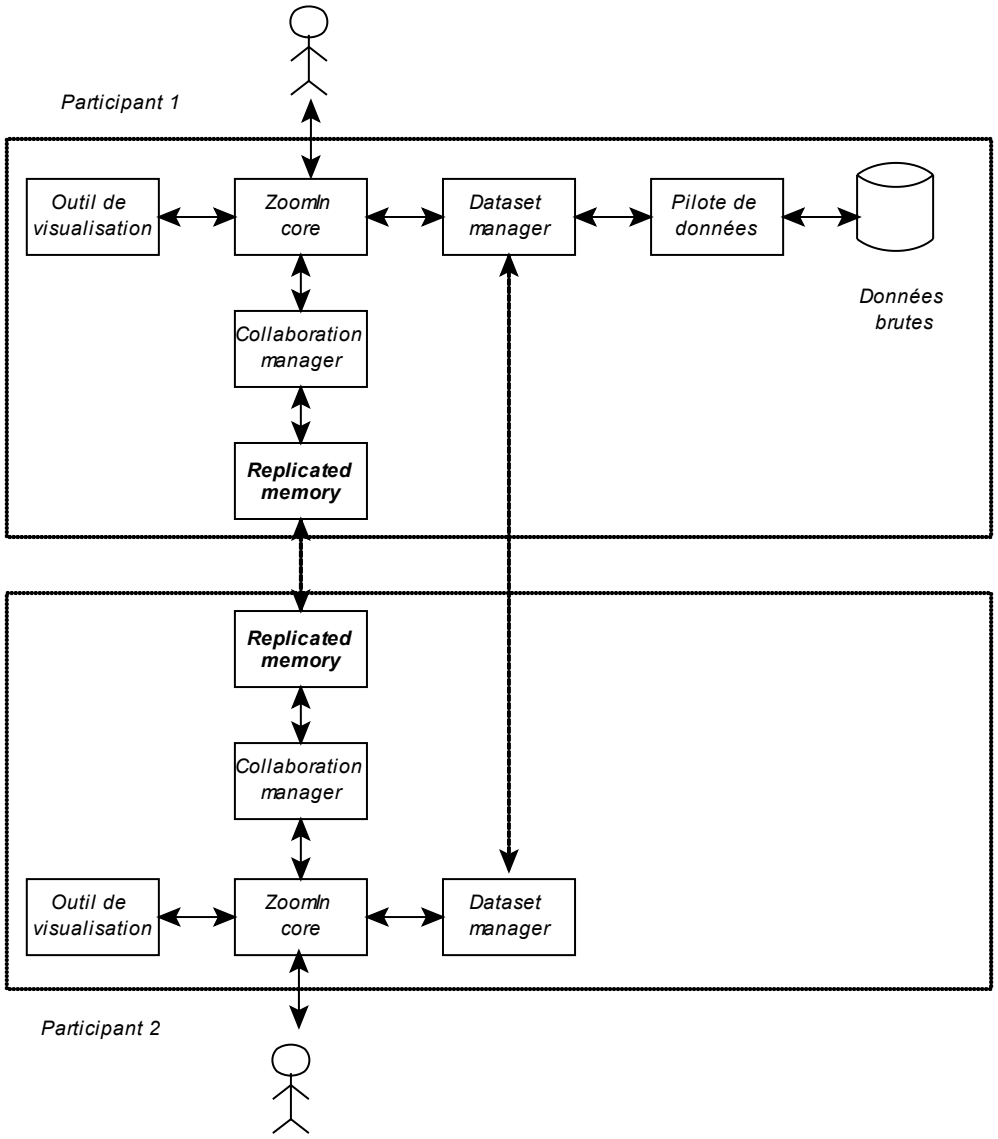

Figure 6.9 : Exemple de l'architecture de ZoomIn avec deux participants, dont l'un possède les données brutes

Les transmissions d'informations entre les participants se font uniquement à travers les modules *Replicated memory* et *Dataset manager*.

Le *Collaboration manager* est décrit plus en détail par la figure 6.10. Le flux d'information venant de *ZoomIn core* est envoyé à travers un *manager* qui est chargé de traiter l'information. Nous décrivons les différents *managers* et leurs rôles :

- *CVO manager* gère les actions des participants.
- *Dataserver manager* gère la liste des pilotes de données. Cela permet aux autres participants d'y accéder.
- *View manager* gère la position des avatars et des pointeurs.
- *Permission manager* gère les permissions.
- *Participant manager* gère les propriétés des participants, tel que leur nom, état ou couleur.
- *Memory accessor* gère l'accès à la *Replicated memory*.

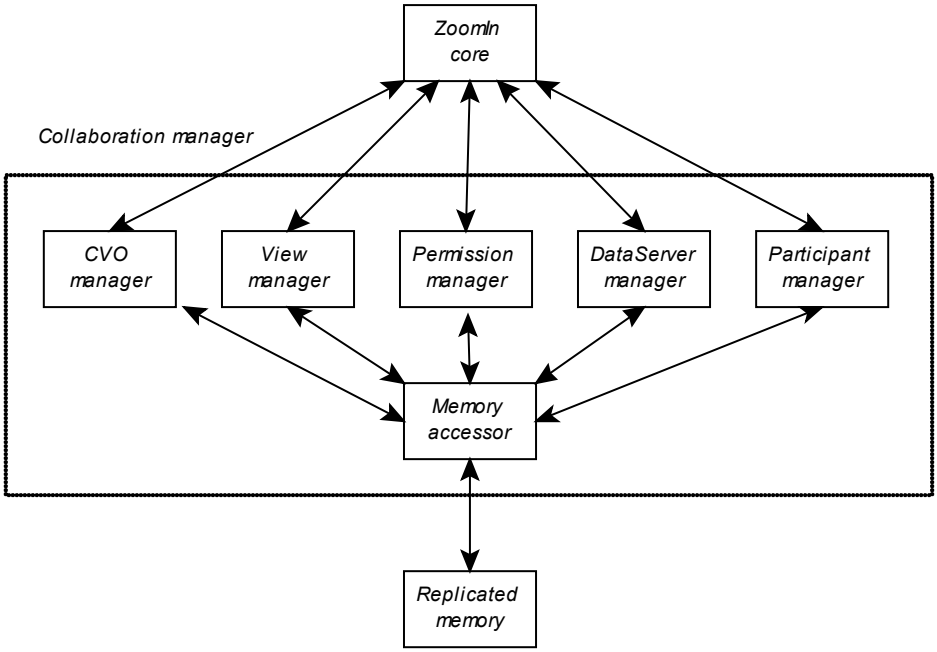

Figure 6.10 : Détail du module *Collaboration manager*

Pour plus de détails techniques, il faut se référer au rapport d'implémentation de ZoomIn [13] ainsi qu'au code source et à la documentation disponibles sur le site web [14].

6.6 Points forts de l'architecture

Voici les points forts de notre architecture en rapport avec les défis énoncés à la section 3.3 :

- **Portabilité sur différents systèmes d'exploitation – Support de matériels différents** : Le fait d'utiliser Java, Java3D et RMI confèrent une grande portabilité au système. Ce dernier a été testé avec succès sous Windows, Linux et Solaris. Le système est lancé depuis notre page web grâce à *« Java Web Start »*.
- **Contrôle d'accès et protection des données de visualisation** : Le fait de devoir transiter par un pilote de données pour accéder aux données brutes permet de mettre en place un mécanisme simple de protection basé sur les permissions. Les modes « libre »

et « présentation » peuvent être configurés en fixant les permissions de manière adéquate.

- **Système intuitif** : Les avantages du système mono-utilisateur tels que la facilité d'utilisation, l'intégration de nouveaux types de données ou l'ajout de nouveaux outils de visualisation sont conservés dans cette architecture. L'heuristique qui choisit la configuration à utiliser (une des cinq configurations évoquées dans la section 4.1.3) décharge l'utilisateur final d'un travail fastidieux. De plus, diverses optimisations (comme l'amélioration des transmissions lorsque les mêmes données brutes sont situées sur plusieurs ordinateurs) sont rendues possibles grâce à l'architecture mise en place.

- **Flexibilité des rôles** : Notre architecture autorise les participants à changer dynamiquement leur rôle en cours de session, par exemple en autorisant un élève à devenir le maître ; ce que très peu de systèmes permettent. Elle ne limite pas le nombre de participants.

6.7 Améliorations possibles

Notre système de VSC conserve tous les avantages du système de VS (dont l'extensibilité et la facilité d'utilisation) sur lequel il est basé, tout en offrant des fonctionnalités liées à la collaboration La flexibilité de son architecture nous a permis d'y intégrer de nouvelles fonctionnalités, comme par exemple la réalisation d'un cache (section 4.5.1) ou la possibilité d'indiquer des endroits dans le monde 3D (section 6.3.2).

Concernant notre prototype, il reste encore certains points sur lesquels une amélioration est possible :

- Amélioration de la fiabilité et de la robustesse : le système est actuellement un prototype, il souffre encore d'un certain nombre de défauts de jeunesse (par exemple, une panne sur un ordinateur implique que tous les ordinateurs seront déconnectés). Il n'a d'ailleurs pas encore pu être soumis à des tests intensifs.

- Résolution du problème des pare-feux : une connexion ne peut souvent pas être établie lorsque les ordinateurs doivent communiquer à travers un pare-feu. Il existe des techniques pour contourner ce problème, mais les performances s'en trouvent diminuées.

- Amélioration de l'efficacité : le temps d'attente peut encore être réduit, par exemple en réalisant un mécanisme permettant une compression judicieuse des informations à transmettre.

- Amélioration de l'intuitivité : la connexion à une session de visualisation peut s'effectuer de manière plus intuitive qu'en fournissant une adresse IP ; une idée consiste à mettre à disposition une page web indiquant toutes les sessions disponibles et permettant à un participant de se connecter à une d'entre elles.

Chapitre 7 Conclusion

7.1 Résumé

La visualisation scientifique est utilisée habituellement pour résoudre des problèmes complexes, en faisant appel aux compétences de plusieurs experts, souvent dispersés géographiquement. En outre, dans l'enseignement la visualisation scientifique est un instrument particulièrement utile, par exemple pour présenter des résultats aux élèves. La collaboration joue ainsi un rôle crucial dans le domaine de la visualisation scientifique

Dans cette thèse, nous avons présenté les différentes notions jouant un rôle essentiel dans le domaine de la collaboration, puis discuté divers systèmes de visualisation scientifique collaborative et établi les besoins des utilisateurs pour de tels systèmes. Nous avons montré que la plupart des systèmes de visualisation scientifique collaborative actuellement disponibles ne satisfont pas toutes les exigences du travail collaboratif à distance. Certains sont mieux adaptés à la collaboration alors que d'autres cherchent avant tout une grande efficacité. Nous avons conçu un système qui soit à la fois bien adapté à la collaboration et efficace. Une expérimentation avec un petit groupe de participants a été réalisée pour analyser les effets des outils de collaboration du système.

7.2 Contributions

Collaborer implique des transmissions d'informations entre les participants. Notre contribution porte sur deux points : réduire le temps d'attente pour les participants et obtenir un système facile d'utilisation.

Afin de réduire le temps d'attente pour les participants, le concepteur d'un système de visualisation scientifique collaborative doit analyser les différentes possibilités de transmettre les données de visualisation entre les ordinateurs concernés : cette transmission peut se faire au niveau des données brutes, des paramètres soumis par l'utilisateur ou des objets graphiques engendrés. Nous avons établi cinq configurations possibles pour transmettre ces données. Malheureusement, la meilleure configuration dépend à la fois de l'outil de visualisation utilisé, de la valeur des paramètres soumis et des valeurs des données brutes concernées. C'est pourquoi, nous avons développé une heuristique qui, pour chaque action demandée par les participants, choisit une configuration adéquate. Comme notre heuristique ne dispose pas de suffisamment d'informations pour espérer trouver toujours la meilleure configuration, notre heuristique comporte une composante probabiliste pour limiter l'effet du choix sur le temps d'attente.

Concernant la facilité d'utilisation, nous avons développé des indicateurs censés assurer un support de l'*awareness*. Pour vérifier la facilité d'utilisation de notre système, nous avons effectué une expérimentation. Des participants ont utilisé notre système pour accomplir une tâche assez simple. Nous avons comparé les résultats obtenus en fonction des moyens mis à disposition. Il en ressort que l'audio, comparé au *chat* seul, permet d'améliorer les performances et de faciliter la communication. Nous n'avons en revanche pas pu mettre en évidence un effet significatif imputable à la disponibilité de nos indicateurs (avatars et pointeurs) ; cependant, nous pouvons affirmer, sur la base des tendances observées et des impressions subjectives des participants, qu'ils sont utiles au minimum dans certaines situations.

Nous estimons que nos objectifs principaux ont été atteints : le système est à la fois performant et simple d'utilisation. Néanmoins, l'évolution des performances des réseaux et des ordinateurs permet d'envisager diverses améliorations.

7.3 Recherches et améliorations futures concernant la visualisation scientifique collaborative

Pour conclure nos travaux, nous proposons quelques voies de recherche future ainsi que des prévisions quand à l'avenir de la visualisation scientifique collaborative.

Ces dernières années, les transmissions par le réseau sont devenues de plus en plus rapides. Cette évolution a malheureusement été compensée par l'augmentation du volume des données brutes. En effet, les simulations numériques et les expérimentations dans certains domaines scientifiques, telle que la physique des particules, produisent une quantité de données brutes de plus en plus grande. Il reste donc difficile, voire impossible, de transférer, même à travers un réseau très rapide, toutes les données brutes pour que, dans une visualisation collaborative, les accès aux données puissent se faire localement.

C'est pourquoi une amélioration des performances des réseaux et des algorithmes de visualisation s'avère encore nécessaire. L'utilisation d'algorithmes de compression lors du transfert de données ou la répartition des calculs sur les différents ordinateurs devraient contribuer à réduire le temps d'attente. Grâce à ces évolutions, on peut penser que, dans les prochaines années, une participation à une session de visualisation scientifique collaborative deviendra possible indépendamment du lieu physique pour la plupart des utilisateurs.

Les possibilités de représenter un phénomène complexe sur un écran (donc par des représentations essentiellement bidimensionnelles) sont limitées. Dans de nombreux cas, il manque au moins une dimension, ce que tout utilisateur remarque immédiatement lorsqu'il a de la peine à situer un point précis dans l'espace. De nouveaux périphériques de réalité virtuelle devraient permettre d'améliorer cette perception de l'espace.

Un autre défi qui se pose est le passage de la visualisation faite sur un ordinateur à d'autres appareils permettant de représenter les résultats d'une visualisation ; il faut tenir compte des caractéristiques propres à chaque appareil que ce soit, par exemple, un PDA (*Personal Digital Assistant*), un téléphone portable, un ordinateur portable, un ordinateur de bureau ou un mur d'image.

Les défis liés à l'évaluation en visualisation sont particuliers : tandis qu'il est assez aisé de juger les performances, que ce soit en temps ou en mémoire, d'un algorithme, de tels métriques sont difficiles à définir pour démontrer le gain lié à l'utilisation de la visualisation et à la facilité d'utilisation du système. C'est pourquoi les liens avec d'autres domaines tels que les interfaces hommes-machines ou la psychologie doivent se faire de plus en plus souvent.

L'augmentation en volume des données n'implique pas une augmentation proportionnelle du volume d'informations utiles. Un des principaux défis consiste à comprendre et à utiliser efficacement le volume d'informations fournies.

L'usage de solution basée sur les grilles informatiques (*grid*) permet de répondre en partie aux défis des calculs et du stockage qu'implique la visualisation scientifique. Les techniques et méthodes traditionnelles sont difficiles à utiliser dans de tels cas, en effet le volume de

données est d'un ordre de grandeur nettement plus grand que celui auquel on a habituellement à traiter. Il en résulte qu'il faut encore développer des abstractions qui permettent d'avoir une vue d'ensemble des données tout en pouvant représenter les détails.

Références

1. Adbelhak, S., *Amélioration de la couche réseau de ZoomIn*. 2005, Travail de diplôme, IIUN, Université de Neuchâtel.
2. *AVS,* 2005. http://www.avs.com. Dernière visite: 06.09.2005.
3. Bajaj, C. and S. Cutchin. *Web based collaborative visualization of distributed and parallel simulation*. In *Proceedings of the 1999 IEEE Symposium on Parallel Visualization and Graphics*. 1999. San Francisco. p. 47-54.
4. Baker, K., S. Greenberg, and C. Gutwin. *Empirical development of a heuristic evaluation methodology for shared workspace groupware*. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*. 2002. p. 96-105.
5. Berner, E., *Base de données distribué pour le logiciel ZoomIn*. 1998, IIUN, Université de Neuchâtel.
6. Boyd, D.R.S., J.R. Gallop, and J.P.R.B. Walton. *Putting you in the picture: enhancing visualization with a virtual environment*. In *IEEE Visualization '99—Late Breaking Hot Topics*. 1999.
7. Boyles, M., R. Raje, and S. Fang, *CEV: Collaborative environment for visualization using Java RMI.* ACM 1998 Workshop on Java for High-Performance Network Computing, 1998.
8. Brodlie, K., D. Duce, J. Gallop, M. Sagar, J. Walton, and J. Wood. *Visualization in grid computing environments*. In *Proceedings of IEEE Visualization*. 2004. p. 155-162.
9. Brodlie, K., D.A. Duce, J.R. Gallop, J.P.R.B. Walton, and J.D. Wood, *Distributed and collaborative visualization.* Computer Graphics Forum, 2004. **23**(2): p. 223-251.
10. Brodlie, K., et al., Collaborative visualisation using access grid. 2002. http://www.comp.leeds.ac.uk/kwb/all_hands/BOF.pdf. Dernière visite: 29.3.2007.
11. Carlsson, C. and O. Hagsand, *DIVE - A platform for multi-user virtual environments.* Computers and Graphics, 1993. **17**(6): p. 663-669.
12. Casera, S., *Intégration d'un nouvel outil dans ZoomIn: Textures représentant un champ vectoriel*. 2000, Travail de diplôme, IIUN, Université de Neuchâtel.
13. Casera, S., *Visualisation scientifique collaborative : Rapport technique global*. 2004, IIUN, Université de Neuchâtel.
14. Casera, S., ZoomIn. 2006. http://iiun.unine.ch/paral/zoomin. Dernière visite: 22.5.2007.
15. Casera, S., H.-H. Naegeli, and P. Kropf. *A collaborative extension of a visualization system*. In *DFMA 2005*. 2005. Besançon. p. 176-182.
16. Casera, S., H.-H. Naegeli, and P. Kropf. *Improving usability of collaborative scientific visualization systems* In *Proceedings of the Sixth IASTED International Conference on Visualization, Imaging, and Image Processing*. 2006. Palma de Mallorca, Spain: ACTA Press. p. 76-81.
17. CERN, LHC computing project (LCG). 2006. http://lcg.web.cern.ch/LCG/. Dernière visite: 03.10.2006.
18. Chabert, A., E. Grossman, L.S. Jackson, S. Pietrowiz, and C. Seguin, *Java object-sharing in Habanero.* Communications of the ACM, 1998. **41**(6): p. 69-76.

19. Clark, H.H. and S.E. Brennan, *Grounding in communication*, In *Perspectives on Socially Shared Cognition*, L.B. Resnick, J. Levine, and S.D. Teasley, Editors. 1991, APA. p. 127-149.
20. Cruz-Neira, C., D.J. Sandin, and T.A. DeFanti. *Surround-screen projection-based virtual reality: The design and implementation of the CAVE*. In *Computer Graphics*. 1993. New York: ACM Press. p. 135-142.
21. Danis, C. *Extending the concept of awareness to include static and dynamic person information*. In *Proceedings of the International Workshop on Awareness and the WWW*. 2000. Philadelphia, PE, USA.
22. Dillenbourg, P., *What do you mean by collaborative learning ?*, In *Collaborative Learning : Cognitive and Computational Approaches* P. Dillenbourg, Editor. 1999, Elsevier Science/Pergamon: Amsterdam p. 1-19.
23. Dourish, P. and V. Bellotti. *Awareness and coordination in shared workspaces*. In *CSCW'92 Conference on Computer Supported Cooperative Work*. 1992. Toronto, Canada. p. 107-114.
24. Duce, D.A., *The computer graphics reference model*, In *Distributed Window Systems - Theory and Practice*. 1991, Eurographics Publications Series.
25. Duce, D.A., J.R. Gallop, I.J. Johnson, K. Robinson, C.D. Seelig, and C.S. Cooper. *Distributed cooperative visualization - Experiences and issues from MANICORAL project*. In *Eurographics Workshop on Visualization in Scientific Computing*. 1998: Eurographics Association.
26. Duce, D.A., et al., *Reference models for distributed cooperative visualization.* Computer Graphics forum, 1998. **17**(4): p. 219-233.
27. *EASA award,* 2004. http://www.easa-award.net/. Dernière visite: 22.3.2007.
28. Ellis, C.A., S.J. Gibbs, and G.L. Rein, *Groupware: Some issues and experiences.* Communications of the ACM, 1991. **34**(1): p. 38-58.
29. Endsley, M., *Toward a theory of situation awareness in dynamic systems.* Human Factors, 1995. **37**(1): p. 32-64.
30. Engel, K., P. Hastreiter, B. Tomandl, K. Eberhardt, and T. Ertl. *Combining local and remote visualization techniques for interactive volume rendering in medical applications*. In *Proceedings of IEEE Visualization*. 2000. p. 6-10.
31. *Fluent* 2005. http://www.fluent.com. Dernière visite: 22.3.2007.
32. Fogel, K. and M. Bar, *Open source development with CVS*. 3rd ed. 2003: Paraglyph Press.
33. Fuhrmann, A., *Studierstube: a collaborative virtual environment for scientific visualization*. PhD Thesis, 1999. Technischen Universitat Wien.
34. García López, P. and A.F. Gómez-Skarmeta, *ANTS framework for cooperative work environments.* IEEE Computer, 2003. **36**(3): p. 56-62.
35. Grudin, J., *Eight challenges for developers.* Communications of the ACM, 1994. **37**(1): p. 93-105.
36. Grudin, J., *Utility and usability: research issues and development contexts.* Interacting with Computers, 1992. **4**(2): p. 209-217.
37. Gutwin, C. and S. Greenberg, *A descriptive framework of workspace awareness for real-time groupware.* Computer Supported Cooperative Work, 2002. **11**(3): p. 411-446.
38. Gutwin, C. and S. Greenberg, *The importance of awareness for team cognition in distributed collaboration.* Team Cognition: Understanding the Factors that Drive Process and Performance, 2004: p. 177-201.

39. Gutwin, C., D. Pinelle, and S. Greenberg, *Task analysis for groupware usability evaluation: modeling shared-workspace tasks with the mechanics of collaboration.* ACM Transactions on Computer-Human Interaction, 2004. **10**(4): p. 281-311.
40. Gutwin, C., M. Roseman, and S. Greenberg. *A usability study of awareness widgets in a shared workspace groupware system*. In *Proceedings of ACM CSCW'96 Conference on Supported Cooperative Work*. 1996. Boston. p. 258-267.
41. *gViz project website,* 2005. http://www.comp.leeds.ac.uk/vis/gviz/. Dernière visite.
42. Haber, R.B. and A. McHabb, *Visualization idioms: A conceptual model for scientific visualization systems*, In *Visualization in scientific computing*, G.M. Nielson, B. Shriver, and L.J. Rosenblum, Editors. 1990, IEEE Press. p. 74-93.
43. Hancock, M., J.D. Miller, S. Greenberg, and S. Carpendale. *Exploring visual feedback of change conflict in a distributed 3D environment*. In *Proceedings of Advanced Visual Interfaces (AVI'06)*. 2006. Venezia, Italy.
44. Hansen, C. and C. Johnson, *The visualization handbook*. 2004: Elsevier Academic Press, Morgan Kaufmann.
45. Hibbard, W., *VisAD: connecting people to computations and people to people.* Computer Graphics 1998. **32**(3): p. 10-12.
46. Hibbard, W., et al., *Java distributed components for numerical visualization in VisAD* Communications of the ACM 2005. **48**(3): p. 98-104.
47. Ibanez, L., W. Schroeder, L. Ng, and J. Cates, *The ITK software guide*. Second ed. 2005: Kitware, Inc.
48. ISO9241-11:1998, *Ergonomic requirements for office work with visual display terminals (VDTs) -- Part 11: Guidance on usability*. 1998.
49. ISO/IEC, *Information processing systems computer graphics - Computer graphics reference model. ISO 11072*. 1999: ISO Central Secretariat.
50. Jeannotat, C., *Adaptation de ZoomIn au travail à distance*. 2003, IIUN, Université de Neuchâtel.
51. Johnson, C.R., R. Moorehead, T. Munzner, H. Pfister, P. Rheingans, and T.S. Yoo, *NIH-NSF visualization research challenges report*. First ed. 2006, Los Alamitos, CA, USA: IEEE Press.
52. Johnson, G., *Collaborative visualization 101.* Computer Graphics, 1998. **32**(2): p. 8-11.
53. King, B.M. and E.W. Minium, *Statistical reasoning in psychology and education*. Fourth ed. 2003, Hoboken, New Jersey: John Wiley & Sons, Inc.
54. Kircher, M. and D.C. Schmidt, *DOVE: A Distributed Object Visualization Environment.* C++ Report, 1999. **11**(2).
55. Krauss, R.M. and S.R. Fussell. *Mutual knowledge and communicative effectiveness*. In *Intellectual Teamwork: Social and Technical Bases of Collaborative Work*. 1990: Erlbaum. p. 111-145.
56. Kraut, R.E., S.R. Fussell, and J. Siegel, *Visual information as a conversational resource in collaborative physical tasks.* Human-Computer Interaction, 2003. **18**: p. 13-49.
57. Lee, A. and A. Girgensohn, *Design, experiences and user preferences for a web-based awareness tool.* International Journal of Human-Computer Studies, 2002. **56**(1): p. 75 - 107.
58. Liechti, O., *Awareness and the WWW: an overview.* ACM SIGGROUP Bulletin, 2000. **21**(3): p. 3-12.
59. *FieldView,* 2005. http://www.ilight.com. Dernière visite: 22.3.2007.

60. Lorensen, W.E. and H.E. Cline, *Marching cubes: A high resolution 3D surface construction algorithm.* Computer Graphics (SIGGRAPH '87 Proceedings), 1987. **21**(4): p. 163-170.
61. *Iris Explorer,* 2005. http://www.nag.co.uk/. Dernière visite: 22.3.2007.
62. Luke, E.J. and C.D. Hansen. *Semotus Visum: a flexible remote visualization framework*. In *Proceedings of IEEE Visualization '02*. 2002. p. 61-68.
63. Mayhew, D., *The usability engineering lifecycle: a practitioner's handbook for user interface design*. 1999, San Francisco: Morgan Kaufmann Publishers.
64. *Netmeeting* 2003. http://www.microsoft.com/netmeeting/ Dernière visite: 22.3.2007.
65. Milaho, W.E., Review: A collaborative extension of a visualization system. 2006. http://www.reviews.com. Dernière visite: 23.9.2006.
66. *FAST,* 1995. http://www.nas.nasa.gov/Software/FAST. Dernière visite: 08.09.2005.
67. *FastExpeditions,* 1997. http://www.nas.nasa.gov/Software/FAST/FASTexpeditions/. Dernière visite: 08.09.2005.
68. Neale, D.C., J.M. Carroll, and M.B. Rosson. *Evaluating computer-supported cooperative work: models and frameworks* In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work* 2004. Chicago, Illinois, USA p. 112-121.
69. Neale, D.C., D.R. Dunlap, P. Isenhour, and J.M. Carroll. *Collaborative critical incident development*. In *Proceedings of the 44th Annual Meeting of the Human Factors and Ergonomics Society*. 2000. Santa Monica, CA, USA. p. 598-601.
70. Nielsen, G., H. Hagen, and H. Müller, *Scientific visualization: Overviews, methodologies, and techniques*. 1997: IEEE Press.
71. Nielsen, J., *Usability engineering*. 1994, Boston: Academic Press.
72. Nova, N., *The impact of awareness tools on mutual modelling in a collaborative game*. MSc Thesis, 2002. University of Geneve.
73. O'Malley, C., S. Langton, A. Anderson, G. Doherty-Sneddon, and V. Bruce, *A comparison of face-to-face and video-mediated interaction.* Interacting with Computers, 1996. **8**(2): p. 177-192.
74. Oberli, J., *Amélioration de l'outil isosurface dans ZoomIn*. 2003, IIUN, Université de Neuchâtel.
75. Oberli, J., *Intégration d'un outil ruban dans ZoomIn*. 2002, IIUN, Université de Neuchâtel.
76. OMG, *Common Object Request Broker Architecture: core specification.* Specification formal/04-03-12, 2004.
77. Ou, J., X. Chen, S.R. Fussell, and J. Yang. *DOVE: Drawing over Video Environment*. In *Proceedings of the Eleventh ACM International Conference on Multimedia*. 2003. Berkeley, CA, USA.
78. Pang, A., *Spray rendering.* Computer Graphics and Applications, 1994. **14**(5): p. 57-63.
79. Pang, A. and C.M. Wittenbrink, *Collaborative 3D visualization with CSpray.* Computer Graphics, 1997. **17**(2): p. 32-41.
80. Pedersen, E. *People presence or room activity: Supporting peripheral awareness over distance*. In *Proceedings of the Conference on Human Factors in Computing Systems*. 1998. p. 283-284.
81. Petraglio, L., *Implémentation d'un cache dans ZoomIn*. 2006, Travail de semestre, IIUN, Université de Neuchâtel.
82. *RealVNC, RealVNC Ltd* 2005. http://www.realvnc.com. Dernière visite: 22.3.2007.

83. Rhyne, T.-M., M. Tory, T. Munzner, M. Ward, C. Johnson, and D.H. Laidlaw. *Information and scientific visualization: Separate but equal or happy together at last*. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*. 2003. p. 611-614.
84. Roschelle, J. and S.D. Teasley, *The construction of shared knowledge in collaborative problem solving*, In *Computer Supported Collaborative Learning*, C. O'Malley, Editor. 1995, Springer: Berlin. p. 69-97.
85. Roseman, M. and S. Greenberg, *Building groupware with GroupKit*, In *Tcl/Tk Tools*, M. Harrison, Editor. 2003. p. 535-564.
86. Sanglard, H., *Toward an easy-to-learn and extensible platform for scientific visualization*. PhD Thesis, 2001. University of Neuchâtel, Switzerland.
87. Schissel, D.P., et al., *Data Management, code deployment, and scientific visualization to enhance scientific discovery in fusion research through advanced computing.* Fusion Engineering and Design, 2002. **60**: p. 481.
88. Scholl, J., J. McCarthy, and R. Harr. *A comparison of chat and audio in media rich environments*. In *CSCW '06: Proceedings of the 2006 20th Conference on Computer Supported Cooperative Work*. 2006: ACM. p. 323-332.
89. Schroeder, W., K. Martin, and B. Lorensen, *The visualization toolkit: An object-oriented approach to 3D graphics*. 2003: Kitware, Inc.
90. *Skype, Skype Limited,* 2007. http://www.skype.com. Dernière visite: 22.3.2007.
91. Sohlenkamp, M., W. Prinz, and L. Fushs, *PoliAwac: design and evaluation of an awareness enhanced groupware client.* AI & Society Journal, 2000. **14**: p. 31-47.
92. Sonnenwald, D.H., M.C. Whitton, and K.L. Maglaughlin, *Evaluating a scientific collaboratory: results of a controlled experiment* ACM Transaction on Computer-Human Interaction, 2003. **10**(2): p. 150-176.
93. Sowizral, H., K. Rushforth, and M. Deering, *The Java 3D API specification*. Second ed. 2001: Addison-Wesley.
94. Steed, A., D. Alexander, P. Cook, and C. Parker, *Visualizing diffusion-weighted MRI data using collaborative virtual environment and grid technologies*, In *Theory and Practice of Computer Graphics (TPCG03)*. 2003, IEEE Computer Society Press. p. 156-161.
95. *Studierstube project,* 2005. http://www.studierstube.org/. Dernière visite.
96. SunMicrosystemsInc, Java Remote Method Invocation (RMI). 2006. http://java.sun.com/products/jdk/rmi/. Dernière visite: 27.09.2006.
97. *Amira,* 2007. http://www.tgs.com. Dernière visite: 22.3.2007.
98. TessellaCompany, Visualization of data with VIVRE. 2005. http://www.tssp.co.uk/Literature/articles/tessarchive/vivre.htm. Dernière visite: 09.09.2005.
99. *TightVNC, OpenSource Software,* 2005. http://www.TightVNC.org. Dernière visite: 22.3.2007.
100. van Dam, A., D. Laidlaw, and R. Simpson, *Experiments with immersive virtual reality for scientific visualization.* Computers and Graphics, 2002. **26**(4): p. 535-555.
101. van Wijk, J., *Views on visualization.* IEEE Transactions on Visualization and Computer Graphics, 2006. **12**(4): p. 421-432.
102. Varrin, D., *Sonde multi-paramètres et résolution des ambiguités des Marching cubes* 2001, IIUN, Université de Neuchâtel.
103. Vidal, F.P., et al., *Principles and applications of computer graphics in medicine.* Computer Graphics Forum, 2006. **25**(1): p. 113-137.
104. *VisAD,* 2005. http://www.ssec.wisc.edu/~billh/visad.html. Dernière visite: 22.3.2007.
105. Walton, J.P.R.B., *NAG's IRIS Explorer*, In *Visualization handbook* N.Y. Academic Press, Editor. 2003.

106. Weir, P. and A. Cockburn. *Distortion-oriented workspace awareness in DOME*. In *Proceedings of the 1998 British Computer Society Conference on Human-Computer Interaction*. 1998. p. 239-252.
107. Wernecke, J., *The inventor mentor*. 1994: Addison-Wesley Professional.
108. Wood, J.D., *Collaborative visualization*. PhD Thesis, 1998. The University of Leeds, School of Computer Studies.
109. Wood, J.D., H. Wright, and K. Brodlie. *CSCV - Computer Supported Collaborative Visualization*. In *BCS Displays Group International Conference on Visualization and Modelling*. 1995.