Mise en contexte de la conscience de groupe : adaptation et visualisation

Christophe Bouthier

Département de formation doctorale en informatique

Institut National
Polytechnique de Lorraine

École doctorale IAEM Lorraine

Mise en Contexte de la Conscience de Groupe : Adaptation et Visualisation

THÈSE

présentée et soutenue publiquement le 13/09/2004

pour l'obtention du

Doctorat de l'Institut National Polytechnique de Lorraine
(spécialité informatique)

par

Christophe Bouthier

Composition du jury

Présidents :	Karl TOMBRE	Professeur, École des Mines de Nancy, *Référent interne*
Rapporteurs :	Michel BEAUDOUIN-LAFON	Professeur, Université Paris Sud
	Manuel ZACKLAD	Professeur, Université de Technologie de Troyes
Examinateurs :	Christian BRASSAC	Maître de Conférence HDR, Université Nancy 2
	Gérôme CANALS	Maître de Conférence, Université Nancy 2
	Claude GODART	Professeur, Université Henri Poincaré Nancy 1, *Directeur de thèse*

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thloria.

Résumé:

Cette thèse propose deux contributions à la problématique de la conscience de groupe. La première contribution porte sur la mise en contexte de la conscience de groupe pour sa construction et son adaptation. Le contexte de travail de l'utilisateur est utilisé comme source des informations constituant la conscience de groupe, ainsi que pour adapter la propagation et la représentation de ces informations. L'utilisation du contexte se fait suivant un modèle cognitif proposé après analyse des mécanismes cognitifs de la conscience de groupe. Il en résulte une conscience de groupe plus pertinente et moins intrusive. La deuxième contribution porte sur la représentation des informations de conscience de groupe, et plus particulièrement sur l'utilisation combinée de deux techniques de visualisations, la visualisation *treemap* et la visualisation *hypertree*. L'emploi combiné de ces deux techniques de visualisation permet d'utiliser la représentation la plus adéquate suivant la tâche à accomplir.

Mots-clés: Conscience de groupe, modèle cognitif, contexte, visualisation, *treemap*, *hypertree*.

Group Awareness in Context : Adaptation and Visualization

Abstract:

This thesis deals with two contributions for the group awareness problematic. The first contribution is to put group awareness in context in order to build and adapt it. User's work context is used not only as the source for group awareness information, but also to adapt the propagation and representation of this information. Context use is based on a cognitive model derived from the analysis of group awareness cognitive mechanisms. The result is a more pertinent and less intrusive group awareness. The second contribution deals with the representation of group awareness, and more especially with the combinated uses of two visualization techniques, the *treemap* visualization and the *hypertree* visualization. The combination of those two visualization techniques allows to use the more adapted representation for the task at hand.

Keywords: Group awareness, cognitive model, context, visualization, *treemap*, *hypertree*.

Discipline: informatique

Laboratoire:

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503
Campus Scientifique - B.P. 239 - 54506 Vandœuvre-lès-Nancy CEDEX

Remerciements

Je remercie en premier lieu ma femme, Julia, de m'avoir supporté pendant toutes ces années de thèse, et de continuer même maintenant alors que je n'ai plus d'excuse. Je remercie aussi Émilie, ma fille, de bien avoir voulu naître lors d'une semaine où son papa n'avait pas trop de travail. Je porte seul l'entière responsabilité de tous les retards survenus.

Je remercie aussi ma famille de m'avoir soutenu dans cette voie, si peu rentable financièrement, mais si enrichissante intellectuellement. Je remercie en particulier mon père de m'avoir inculqué le goût des sciences, et ma mère de m'avoir toujours encouragé. Je remercie aussi ma belle-famille pour son soutien morale.

Je remercie grandement tous les membres du jury d'avoir accepté de juger mon travail, et ceci malgré sa multidisciplinarité. Je remercie en particulier les rapporteurs externes, M. Beaudouin-Lafon et M. Zacklad du temps qu'ils y ont consacré.

Je remercie Claude Godart et Gérôme Canals pour avoir accepter d'encadrer ma thèse, et pour m'avoir soutenu même lorsque je prenais du retard. Je remercie en particulier Gérôme pour m'avoir donné la liberté nécessaire à un tel travail interdisciplinaire. Je remercie Claude pour avoir su être sévère mais juste lorsqu'il le fallait. Je remercie aussi le C.N.R.S. d'avoir accepté de financer les trois premières années de ma thèse.

Je remercie tous les membres de l'équipe ECOO avec qui j'ai partagé ces dernières années, et en particulier les amis qui m'ont fait l'honneur de venir à mon mariage, à savoir la famille Molli ainsi qu'AnCé et Olivier Zendra. Je remercie aussi Karim Baina, mon colocataire de bureau, qui a subi mes expériences d'aménagement d'un bureau « social » sans se plaindre.

Je remercie les responsables universitaires qui ont bien voulu me confier la responsabilité des cours à donner à leurs étudiants. Je remercie aussi l'ensemble des étudiants que j'ai eu la chance d'avoir, qui ont non seulement supporté mes cours, mais aussi mes pauvres calembourgs. Enseigner est un plaisir toujours renouvelé grâce à vous tous.

Je remercie enfin mes amis, tout d'abord mes deux témoins Benjamin Bergé et Matthieu Dazy, pour leur amitié sans faille et leur soutien inconditionnel. Je remercie aussi le groupe d'amis de mes années à l'École des Mines de Nancy, avec qui j'ai découvert, pendant de nombreuses nuits caféinées, les joies de l'informatique.

J'ai une pensée spéciale pour certains enseignants que j'ai eu la chance de croiser pendant mon parcours scolaire, du lycée à l'école d'ingénieur en passant par la taupe, et qui m'ont donné non seulement le goût d'apprendre, mais aussi le goût d'enseigner. Leur travail se fait dans l'ombre, sans la reconnaissance du grand public, alors que leur influence peut marquer à vie un étudiant.

La première ébauche de cette thèse a été réalisée sous Unix avec *vim* et LaTeX et sur un PowerMac 9500 sous MacOS 9.1 avec *OzTex*. L'introduction de la version aboutie a été écrite sur une NeXTstation Color. Le reste de la version aboutie, ainsi que toutes les corrections jusqu'à la version finale et les captures d'écrans ont été faites sur un eMac sous MacOS X avec *TexShop* et *pdflatex*. Toutes les autres figures ont été réalisées sur le PowerMac 9500 avec *Canvas 6*, version complète et illimitée distribuée dans le magazine *Computer Art* numéro 44.

Je dédie cette thèse
à mes parents, pour m'avoir donner les moyens de la commencer,
et à Julia et Émilie, pour m'avoir donner ceux de la terminer
(à temps, ou presque...).

Table des matières

Introduction

Les progrès réalisés ces dernières années dans les technologies de communication ont permis l'avènement du travail coopératif distribué. Que ce soit par le biais de communautés d'intérêt ou d'équipes distribuées, de plus en plus de personnes distribuées dans le temps et dans l'espace travaillent ensemble.

Il est vite apparu qu'une difficulté principale du travail coopératif à distance réside dans le manque de *conscience de groupe*. Lorsque des personnes travaillent en même temps et dans le même lieu, elles ont conscience de leur appartenance au groupe ainsi que de la place de leur activité au sein du groupe, et cela leur permet notamment de mieux se coordonner. Ce sentiment diminue très fortement lorsque ces personnes sont distribuées dans le temps et/ou dans l'espace.

Le problème de la conscience de groupe est important pour le travail coopératif assisté par ordinateur et a fait l'objet de nombreux travaux ces dix dernières années. Plusieurs catégories de consciences de groupe ont été répertoriées, des modèles ont été proposés, et de nombreux outils et systèmes ont été créés et testés.

L'origine de la conscience de groupe fait l'objet d'un consensus. La conscience de groupe émerge de l'ensemble des informations, formelles et informelles, qui sont échangées entre les membres du groupe, de manière intentionnelle ou non. Lorsque les personnes ne sont plus ensemble en même temps, ces informations ne sont plus échangées, et la conscience de groupe est quasi inexistante. Le but des outils de conscience de groupe est de permettre aux personnes distribuées dans le temps ou dans l'espace d'échanger à nouveau ces informations.

Cependant, si ces outils permettent effectivement de reconstruire une conscience de groupe et apportent ainsi une meilleure coordination, ils ajoutent aussi souvent un autre problème : celui de la surcharge d'information. Le problème est alors de trouver un compromis afin de fournir suffisamment d'informations pour permettre la construction de la conscience de groupe, sans toutefois surcharger l'utilisateur, la solution idéale étant de ne fournir à chaque instant que les informations pertinentes pour l'utilisateur.

Cette thèse contient deux propositions complémentaires à ce sujet. La première est de replacer les informations dans leur contexte, et d'utiliser ce contexte pour adapter et visualiser la conscience de groupe, ceci afin de rendre les informations transmises plus pertinentes et moins intrusives. La deuxième proposition est d'utiliser les visualisations *treemap* et *hypertree* pour représenter les informations de la conscience de groupe afin d'adapter la représentation à la tâche à accomplir.

1 Plan de la thèse

La suite du mémoire est structurée de la manière suivante.

Le chapitre « Problématique » détaille la problématique abordée par la thèse. Le cadre du travail y est décrit, suivi par une définition du concept de conscience de groupe, puis par un passage en revue des principaux travaux sur le sujet. Une problématique sur la pertinence des informations de conscience de groupe en est déduite. Cette thèse propose une double contribution concernant cette problématique, proposition qui termine le chapitre.

Le chapitre « Analyse cognitive » décrit l'analyse des mécanismes de construction de la conscience de groupe à l'aide de différents outils et théories de la psychologie cognitive. La place d'un outil de conscience de groupe parmi ces mécanismes est aussi étudiée. Les résultats de ce chapitre servent de base pour les propositions des chapitres suivants.

Les deux chapitres suivants, « Contextualisation » et « Visualisation » sont consacrés respectivement à l'utilisation du contexte pour la construction et l'adaptation de la conscience de groupe, et à l'utilisation combinée des techniques de visualisation *treemap* et *hypertree*. Chaque contribution est décrite en détail, complétée par un état de l'art sur le sujet correspondant.

Le chapitre « Contributions logicielles » décrit les implantations logicielles réalisées pour la validation des propositions décrites dans cette thèse. Le prototype de gestion du contexte *CommanderMo* y est d'abord décrit, suivi par les librairies de visualisation *Treemap Java Library* et *Hypertree Java Library*.

Finalement, le dernier chapitre, « Bilans et perspectives » dresse le bilan de l'utilisation des idées proposées et des systèmes développés et ouvre de nouvelles perspectives en relation avec ce bilan.

Problématique

1 Cadre de travail

Le sujet de cette thèse (la problématique de la conscience de groupe) appartient au domaine de recherche du Travail Coopératif Assisté par Ordinateur — TCAO (*Computer Supported Cooperative Work — CSCW*), dont les conférences majeures sont ACM CSCW, ACM GROUP et ECSCW (European CSCW).

Cette thèse a été effectuée dans l'équipe ECOO, dont le thème de recherche est le TCAO, et plus particulièrement les problèmes liés à la coordination des équipes distribuées [GBC+01, GHB+01]. Une équipe est dite *distribuée* lorsque ses membres sont dispersés dans le temps ou dans l'espace. Les axes de recherche explorés dans l'équipe vont du système de *workflow* [Bit03] aux espaces de travail partagés [CBGM98], avec la réalisation et l'utilisation de nombreux prototypes, comme *Coopéra*[1] [GCP+03], ou comme *LibreSource*[2] [GMO+04].

1.1 Travail Coopératif Assisté par Ordinateur

Les temps ont changé depuis la création de l'ordinateur. De son rôle de machine à calculer, il est devenu, avec l'avènement des réseaux et notamment d'internet, une machine à communiquer. De nouveaux outils ont vu le jour pour créer ou permettre ces nouvelles communications. Le courrier électronique, les forums de discussion, les messageries instantanées ont permis à des millions de personnes de s'échanger idées et fichiers.

Le but de la recherche dans le domaine du *TCAO* est de créer et d'évaluer de nouvelles technologies pour assister la partie sociale et distribuée du travail coopératif, le plus souvent entre collaborateurs distants dans le temps ou l'espace. Les chercheurs du *TCAO* tirent leurs idées non seulement de l'informatique, mais aussi de la psychologie, de la sociologie, et de l'anthropologie.

Il existe plusieurs types de travaux. Parmi ces travaux, certains proposent des modèles de la coopération, permettant par exemple de décrire le travail collaboratif [Edw97, FLMM91], les politiques de collaboration [FS94, MM93], les échanges d'information entre utilisateurs [NMHR+98], etc. D'autres se concentrent sur la création d'outils permettant la collaboration : outil de communication [GB95], de partage de fichiers [BHST95, CMG98, EMP+97], de coordination (workflow) [LED+99], d'édition synchrone de documents [KW99]... Enfin, certains travaux utilisent les résultats de la psychologie pour

[1] `http://coopera.loria.fr/`
[2] `http://www.libresource.org/`

essayer de mieux comprendre l'utilisateur et permettre ainsi d'adapter et d'améliorer les modèles et les outils des axes précédents [BBF+95, Gru88]. Bien sûr, ces recherches ne sont pas opposées mais bien complémentaires [BL99, Lon03]. Les modèles proposés sont mis en place dans des outils, eux-même validés grâce à l'expérience et l'étude psychologique, ce qui permet d'améliorer les modèles ou même d'en proposer de nouveaux.

Le terme *collecticiel* (*groupware*) désigne les outils logiciels implantant ces nouvelles technologies. Il existe plusieurs classifications de ces système. La plus connue est sans doute la matrice de Johansen (cf. fig. 1), qui classifie les différents systèmes coopératifs en fonction de la répartition des utilisateurs dans le temps et/ou dans l'espace : dans le même lieu ou dans des lieux différents, en même temps (synchrones) ou non (asynchrones). Ces deux dimensions orthogonales créent une matrice à quatre quadrants dans lesquelles les systèmes peuvent être classés. Ainsi, un outil de téléconférence sera classé dans le quadrant *lieux différents - synchrone*, un outil de messagerie dans le quadrant *lieux différents - asynchrone*, etc. Cependant, il est de plus en plus fréquent qu'un système ne soit pas dans un seul quadrant mais plutôt à cheval sur un ou plusieurs quadrants. Par exemple, un outil de tableau blanc partagé, où les utilisateurs peuvent dessiner et partager de l'information de manière synchrone, pourra être utilisé aussi bien par des personnes réparties dans l'espace que par des personnes présentes dans le même lieu.

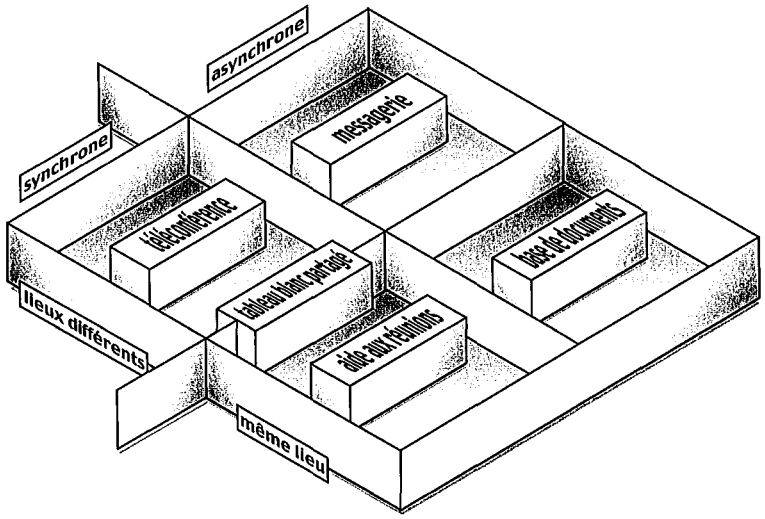

FIG. 1 – La matrice de Johansen [Joh88].

Les utilisateurs des systèmes coopératifs des quadrants autres que *même lieu - synchrone* sont tous confrontés au même problème : l'absence de « conscience de groupe ».

1.2 Conscience de groupe

Le concept de « conscience de groupe » (*awareness*) a émergé au début des années 90 pour décrire le phénomène suivant : lorsque des personnes travaillent ensemble dans le même lieu et en même temps, alors il émerge chez ces individus une « conscience du

groupe » : les membres de l'équipe ont le sentiment de faire partie d'un groupe et arrivent notamment à mieux se coordonner, par exemple en favorisant les échanges spontanés et informels. Ce phénomène, présent naturellement dans les équipes travaillant dans le même lieu et en même temps, diminue fortement lorsque les personnes sont réparties dans le temps ou dans l'espace. Il en résulte pour ces personnes une plus grande difficulté à se coordonner.

L'étude de la conscience de groupe est récente, tant au point de vue informatique que socio-psychologique. La conscience de groupe est désormais un problème important du domaine du Travail Coopératif Assisté par Ordinateur, mais cette importance ne s'est révélée, paradoxalement, que par les problèmes engendrés par la forte diminution de la conscience de groupe dans les équipes distribuées, elles-mêmes rendues possibles par les progrès des télécommunications et l'explosion des réseaux informatiques.

On a d'abord cru que le problème de la conscience de groupe n'était somme toute qu'un problème dû à un manque de richesse du canal de communication, le courrier électronique étant, par exemple, bien moins riche qu'une conversation face-à-face. Pour permettre la construction de la conscience de groupe, il suffisait donc de recréer une communication comparable à la communication « face-à-face » (*face-to-face*). C'est notamment ce qui a été fait dans les projets de type « média-space » [GMM+92, LGS97] où chaque personne avait en permanence une communication audio et vidéo avec tous les autres membre du groupe. L'idée était que ce canal audio/vidéo, en fournissant approximativement la même richesse que la conversation face-à-face, permettrait l'émergence de la conscience de groupe par les mêmes phénomènes que dans un environnement non distribué.

Force a été de constater que malheureusement, le problème n'est pas aussi simple. Même avec la possibilité de communiquer non seulement par la voix mais aussi par les gestes ou les regards, les membres du groupe ont des difficultés à se coordonner et à avoir un sentiment d'appartenance au groupe. Il manque notamment la possibilité d'avoir des communications impromptues (comme lorsque l'on tombe par hasard sur une personne), la possibilité de juger de l'activité de la personne sans la déranger... L'émergence de la conscience de groupe ne peut être réduite à un simple problème de richesse du canal de communication. Mais pour bien comprendre comment se construit la conscience de groupe, il faut d'abord s'entendre sur ce qu'est la conscience de groupe.

2 Concept de « conscience de groupe »

2.1 Définition

La plus connue et la plus utilisée des définitions de la conscience de groupe est celle donnée par Dourish et Belloti [DB92a] :

> « la conscience de groupe désigne une compréhension des activités des autres, qui procure un contexte pour votre propre activité. Ce contexte est utilisé pour s'assurer que les contributions individuelles sont pertinentes pour l'activité globale du groupe, et pour évaluer les actions individuelles par rapport aux buts et à la progression du groupe. »[3]

[3] *Awareness is an understanding of the activities of others, which provides a context for your own*

La conscience de groupe se construit par échange et intégration d'informations sur l'activité des autres membres du groupe.

Différents moyens peuvent être employés pour ces échanges. La parole, orale ou écrite, est souvent le moyen le plus efficace pour échanger une information précise, par exemple dans une conversation, une réunion, un rapport ou même un courriel. Le geste est utilisé pour désigner et pour préciser le sens ou le sujet d'une phrase. Ces échanges peuvent être formels (comme un rapport) ou informels (comme une discussion), mais sont le plus souvent volontaires. D'autres échanges peuvent être involontaires. C'est le cas des informations transmises par le langage corporel : la posture, le mouvement du corps peuvent traduire l'état mental de la personne : calme, nerveuse, inquiète ... Bien que le décryptage fin de ces indices soit l'affaire de spécialistes, tout un chacun est socialement apte à comprendre, même inconsciemment, les plus évidents de ces messages. Enfin, les objets eux-mêmes peuvent transmettre des informations, par leurs états ou bien par leurs histoires. La date de modification d'un fichier peut par exemple indiquer non seulement qu'une personne est passée travailler, mais aussi l'heure et la date de son passage. La lumière filtrant sous la porte d'un bureau, ou encore un bruit de conversation téléphonique derrière une porte traduisent, de manière involontaire, une présence dans le bureau. Lorsque les membres d'un groupe travaillent en même temps et dans le même lieu, ils ont de nombreuses opportunités d'échanger ces types d'informations.

La conscience de groupe a pour principal effet d'améliorer la synergie des personnes qui coopèrent [GG99]. La conscience de groupe permet à chacun de connaître sa place et son rôle au sein du groupe [DB92a]. Elle permet par exemple de savoir quelle est sa part de travail à effectuer dans le groupe pour atteindre le but commun. Cela permet, en retour, une meilleure coordination entre les membres du groupe : il est plus facile de se coordonner avec les autres en sachant ce que chacun fait (ou ne fait pas). La conscience de groupe favorise aussi les interactions impromptues et informelles entre personnes [Gut97]. En fournissant à chacun la connaissance des activités des autres membres du groupe, elle permet à une personne qui recherche une information :

- de trouver la personne la plus susceptible de lui fournir cette information, en fonction de l'historique des activités effectuées ;
- de juger à quel moment la personne sera susceptible d'être interrompue pour lui demander l'information, en fonction de l'activité en cours.

Ces contacts impromptus et informels entre membres d'une équipe ont un grand rôle dans l'auto-coordination de l'équipe.

2.2 Différentes consciences de groupe

Différents aspects de la conscience de groupe peuvent être distingués, suivant le type des informations fournies :

- Conscience des personnes (*People Awareness* [CJMS00, Dan00]) : la conscience des autres personnes du groupe, qui permet de répondre à la question « qui » : qui appartient au groupe, qui n'y appartient pas ? Qui exécute l'activité ?

activity. This context is used to ensure that individual contributions are relevant to the group's activity as a whole, and to evaluate individual actions with respect to group goals and progress.

- Conscience de la présence (*Presence Awareness* [GS00, Pal99]) : la conscience des personnes présentes dans le même temps, mais pas forcément dans le même lieu. Des outils comme *ICQ* ou *Instant Messaging* ont popularisé cette conscience de groupe, en permettant de savoir, dans une communauté ou un groupe, quels sont les membres accessibles en ligne.
- Conscience des activités (*Activity Awareness* [Edw97, NHHG98]) : la conscience des activités exécutées par les autres membres du groupe. Elle permet de répondre à la question : que font les autres ?
- Conscience des documents (*Document Awareness* [BFH00, CJMS00]) ou plus généralement, conscience des artefacts : conscience des outils utilisés et des objets partagés sur lesquelles la collaboration s'effectue. Elle permet de répondre à la question : sur quels documents les autres membres du groupe travaillent-ils ?
- Conscience des émotions (*Emotional Awareness* [GFLM99, GFM99]) : conscience des émotions et de l'état émotionnel des membres du groupe.

Ces différents types de conscience de groupes peuvent être regroupés en consciences de groupe de plus haut niveau, caractérisées par leur domaine d'action :

- Conscience du cercle social (*Social Awareness* [BS98, CSMD00]) : la conscience des informations sociales du groupe : quelles sont les relations entre les personnes, quelles sont leurs émotions ... ?
- Conscience de la structure du groupe (*Group Structural Awareness* [BAB+97, HW00]) : la conscience des informations structurelles : quels sont les rôles de chacun dans le groupe, quelle est la place de chacun par rapport aux autres ...
- Conscience de l'espace de travail (*Workspace Awareness* [BHST95, GG96]) : conscience de ce qui se passe dans l'espace de travail (le plus souvent partagé) : qui regarde quoi, qui fait quoi, quels sont les activités et les objets utilisés par chacun ...
- Conscience de la communauté (*Informal Awareness* [DB92b, GR98]) : conscience des autres informations sur les membres du groupe : qui est présent, que compte-il faire ...

Tous ces types de conscience de groupe ne sont que des facettes de *la* conscience de groupe, définie précédemment comme le contexte des activités de chacun.

3 Principaux travaux et approches existants

Bien que la problématique de la conscience de groupe soit récente, elle a donné lieu à une littérature riche et variée, dans laquelle plusieurs travaux et approches peuvent être distingués. Ces approches ne sont pas antagonistes, et ne représentent en rien une segmentation de l'espace des solutions. Il faut au contraire les voir comme complémentaires car abordant chacun un aspect différent de *la* conscience de groupe.

3.1 Approches *périphériques*

Historiquement, le premier courant que l'on peut distinguer dans la littérature sur la conscience de groupe s'organise autour des approches dites « périphérique ». Il est fondé principalement sur les travaux de Dourish et Belloti [DB92a], lui-même l'un des premiers

articles à parler explicitement de conscience de groupe dans le TCAO. L'approche proposée par ce courant est, comme son nom l'indique, de se focaliser sur la partie des échanges « périphériques », presque inconscients, de la conscience de groupe.

Cette approche est basée sur la constatation suivante : dans son « milieu naturel » — c'est-à-dire lorsque les personnes travaillent en même temps et dans le même espace — la conscience de groupe se construit sans que les membres du groupe ne s'en rendent vraiment compte. Il n'y a pas de « conscience de la conscience de groupe ». Ce n'est que lorsque les équipes sont distribuées que l'utilisateur s'aperçoit qu'il lui manque quelque chose. Ce n'est qu'avec sa quasi disparition dans les équipes distribuées que la nécessité de nommer et d'expliquer ce qui semblait aller de soi s'est faite sentir. Pour l'approche périphérique, le fait que la conscience de groupe se construise sans que les membres du groupe ne s'en rendent compte est due à la partie « périphérique » de cette dernière. C'est parce que la conscience de groupe se construit en grande partie de manière périphérique, sans demander d'attention de la part des membres du groupe, qu'elle semble aller de soi pour les personnes travaillant dans le même espace en même temps. Il est donc normal de privilégier cet aspect « périphérique » lorsque l'on propose des outils permettant de reconstruire la conscience de groupe des équipes distribuées.

En pratique, cela signifie que les mécanismes de conscience de groupe doivent s'appliquer sans que l'utilisateur n'ait à y accorder une attention particulière. L'utilisateur doit prendre conscience de l'information, lui donner un sens et l'intégrer sans y prêter une attention active. Ainsi, les informations doivent être collectées et échangées automatiquement, et présentées de la manière la moins intrusive possible pour l'utilisateur. Pour cela, plusieurs stratégies sont possibles, principalement basées sur des solutions visuelles, plus facile à interpréter et à intégrer de manière périphérique [Koh47].

Cette approche souffre cependant de plusieurs problèmes. En premier lieu, il est très contraignant de se limiter à une collecte d'informations uniquement passive, car un certain nombre d'informations ne sont alors pas disponibles. Les informations de haut niveau par exemple, telles que les relations entre les personnes ou le but de l'action en cours, ne sont connues que des êtres humains. De plus, même dans son milieu naturel, la conscience de groupe n'est pas construite entièrement et uniquement de manière périphérique. Les personnes se posent des questions, s'interrompent pour fournir de l'information, écrivent des rapports et font des présentations en réunion. Il est donc judicieux de mélanger collectes active et passive d'information, afin de ne pas passer à côté d'informations pertinentes pour l'utilisateur.

Un problème plus sérieux concerne la surcharge d'informations. Les informations échangées et présentées peuvent être très nombreuses, particulièrement lorsque le groupe est formé d'un grand nombre de membres, ou bien lorsqu'il y a beaucoup d'artefacts manipulés. Les ressources mentales de l'utilisateur, comme sa mémoire ou son attention, sont limitées. Il doit pourtant interpréter et intégrer ces informations pour pouvoir construire sa conscience de groupe. La surcharge d'informations survient lorsque l'utilisateur se retrouve en face d'un trop grand nombre d'informations à traiter. Il subit alors une situation de stress qui peut l'amener à rejeter en bloc l'ensemble des informations.

Par exemple, dans un système comme *Porthole* [DB92b], chaque bureau est équipé d'une caméra vidéo qui envoie périodiquement une petite image (une vignette) du bureau et de son occupant. Chaque membre du groupe peut ainsi voir en permanence l'ensemble

des autres membres du groupe. Un tel système pose plusieurs problèmes de surcharge d'informations.

Il y a tout d'abord un problème de passage à l'échelle : lorsque le nombre de participants augmente, le nombre de vignettes à afficher augmente aussi, alors que l'espace affichable n'est pas extensible à l'infini. Il semble par exemple presque impossible d'afficher plusieurs centaines de vignettes sur un écran standard, et encore plus de s'en servir pour comprendre les activités des autres membres du groupe afin de construire sa conscience de groupe. Le système devient donc inutilisable s'il y a trop de participants. Ces mêmes problèmes apparaissent dans d'autres systèmes comme *ShrEdit* [DB92a], où le nombre de fenêtres croît proportionnellement au nombre de participants.

Il y a aussi un problème de pertinence des informations. Toujours dans *Porthole* [DB92b], la caméra d'un bureau envoie la même image à l'ensemble des membres du groupe, quelles que soient leur relations avec l'occupant du bureau ou leurs activités courantes. Parmi toutes les vignettes affichées, seules quelques-unes sont pertinentes pour l'utilisateur. Le reste ne fait que surcharger son champ de vision.

Par contre, l'approche « périphérique » limite la surcharge d'informations au niveau de leur présentation. En mettant en avant des mécanismes ne demandant pas d'attention active de la part de l'utilisateur, elle permet d'économiser les ressources d'attention de l'utilisateur. Une présentation graphique périphérique, par exemple, permet à l'utilisateur de ne pas avoir à se concentrer dessus pour interpréter et intégrer les informations présentées.

L'approche « périphérique » met donc l'accent sur les mécanismes de la conscience de groupe qui ne demandent pas d'attention particulière de la part de l'utilisateur. Ces mécanismes sont importants, particulièrement pour la présentation des informations. Cependant, utilisés seuls, ils ne permettent pas de résoudre le problème de la surcharge d'informations.

3.2 Approches *graphiques*

Un autre courant majeur dans la littérature sur la conscience de groupe est celui du traitement « graphique » de la conscience de groupe. Cette approche, présente en particulier dans les travaux de Gutwin et Greenberg [GGC96, GGR96, Gut97, GG01], mais aussi dans les prototypes *BSCW* [BHST95] et *TuaMotu* [MSMB01], axe ses efforts sur la création de nouvelles interfaces graphiques pour représenter les informations de conscience de groupe.

Cette approche a généralement pour cadre les espaces de travail partagés [BHST95, MSMB01], synchrones ou asynchrones, dont les fonctionnalités principales sont le partage de documents et d'informations ainsi que l'édition coopérative synchrone [BLK92, KTBL93]. Dans ce cadre, le but du courant « graphique » est de fournir de nouveaux outils graphiques, des *widgets*, (contraction de *window* et *gadgets*) permettant de rendre compte de la collaboration. Par exemple, les travaux de Gutwin et Greenberg décrivent un outil graphique représentant la vue qu'ont les autres membres du groupe de l'espace de travail. Grâce à cette représentation, chaque participant peut savoir où se porte l'attention des autres. Les résultats montrent que ces outils graphiques ont un réel impact sur l'efficacité de la coordination entre membres d'un groupe [GG98, GG99, GRG96]. L'ap-

proche « graphique » se concentre donc sur la représentation d'une conscience de groupe particulière, la conscience de l'espace de travail.

Ce courant n'est pas très éloigné du courant précédent, le courant « périphérique ». L'accent mis sur les nouveaux outils graphiques permet une assimilation des informations de la conscience de groupe de manière peu intrusive, comme prôné par le courant « périphérique ». La différence entre les deux approches tient surtout au but poursuivi : le courant « graphique » cherche un moyen de représenter une conscience de groupe particulière, alors que le courant « périphérique » prône une méthode d'approche possible de la conscience de groupe en général.

L'approche « graphique » soulève cependant plusieurs difficultés. La première est que cette approche est limitée à un contexte précis : celui de la conscience des espaces de travail partagé. En effet, les outils graphiques développés sont spécifiques à ce cadre : les informations collectées ainsi que leurs représentations sont typiquement liées à l'utilisation de l'espace de travail partagé. Par exemple, les activités représentées sont limitées aux fonctionnalités de l'espace de travail. Cela simplifie la représentation tout en la rendant spécifique donc non réutilisable.

En ce qui concerne la surcharge d'informations, l'approche graphique a un avantage : les informations sont représentées de manière graphique, ce qui permet généralement à l'utilisateur de saisir facilement et rapidement les informations importantes, économisant ainsi ses ressources cognitives. Par exemple, les ascenseurs multi-utilisateurs [Gut97] permettent de voir directement dans quelles parties du document se trouvent les autres membres du groupe, sans grand effort cognitif.

De plus, une représentation graphique permet le plus souvent de représenter les informations dans leur contexte. Par exemple, les télépointeurs [Gut97] ou bien les *viewports* [BLK92] représentent la vue des autres membres du groupe directement dans l'espace de travail partagé. Cette représentation dans le contexte a plusieurs avantages : comme l'information est directement placée dans son contexte, elle est encore plus facile à interpréter, et demande donc encore moins de ressources cognitives.

Mais surtout, l'avantage d'une représentation dans le contexte est de filtrer les informations, car seules les informations en relation avec le contexte de l'utilisateur lui sont présentées. Ainsi, même si des informations identiques sont envoyées à tous les membres du groupe, seules les informations pertinentes pour l'utilisateur sont représentées sur son outil. De plus, ce filtrage améliore le passage à l'échelle. Ainsi, un système comme les ascenseurs multi-utilisateurs [Gut97] n'est plus utilisable lorsqu'il y a trop d'utilisateurs : avoir une centaine d'ascenseurs (un par participant) sur le côté de la fenêtre semble peu utilisable pour en tirer une quelconque information sur les activités du reste du groupe. Par contre, un système comme les télépointeurs [Gut97] reste utilisable avec de nombreux participants, car seuls les pointeurs se trouvant dans la zone de travail de l'utilisateur lui sont présentés.

Cependant, toutes les représentations, même en contexte, ne passent pas forcément bien à l'échelle, lorsque le nombre de participant ou bien le nombre d'artefacts utilisés augmentent fortement. C'est par exemple le cas des vues radar (*radar view*) [Gut97] : avec trop d'utilisateurs ou trop d'objets dans l'espace de travail, la vue radar est complètement surchargée et devient inutilisable.

De plus, la représentation en contexte est complètement liée à l'environnement de

travail, c'est-à-dire à l'espace de travail partagé. Les informations échangées ainsi que leur représentations sont limitées au contexte de l'espace de travail. Cela rend la représentation dépendante de l'utilisation d'un espace de travail partagé, et donc non réutilisable dans un cadre général.

L'approche « graphique » met l'accent sur la création de nouvelles interfaces graphiques pour représenter la conscience de groupe, mais dans le cadre spécifique des espaces de travail partagé. Cette spécification permet une simplification des informations, mais est un obstacle à la généralisation des solutions à d'autres cadres d'utilisation et à d'autres consciences de groupe. De plus, elle n'aborde que partiellement le problème de la surcharge d'informations.

3.3 Travaux orientés *diffusion de messages*

Certains travaux dans la littérature sur la conscience de groupe s'intéressent plus particulièrement à l'infrastructure technique nécessaire à un système de conscience de groupe, et notamment à la « diffusion des messages ». Cette approche, représentée par les travaux de Fitzpatrick (avec *Elvin*) [FMK+99], de Prinz (avec *NESSIE*) [Pri99] et de Walker (avec *SIS*) [Wal98], se concentre sur la partie technique de l'échange des informations.

L'idée de cette approche est de fournir des mécanismes pour la propagation des informations permettant l'émergence de la conscience de groupe. La collecte de ces informations avant la propagation ainsi que leur représentation après la propagation ne sont pas abordées par cette approche. Les résultats de ce courant sont le plus souvent des infrastructures de propagation de messages, possédant différents mécanismes de souscription, d'envoi-réception, etc. Ces infrastructures sont pensées et construites pour être génériques et réutilisables dans n'importe quel cadre, pourvu que des facilités pour la collecte des informations et leurs représentations soient déjà présentes.

Cette approche peut être vue comme une approche orthogonale et complémentaire aux approches précédentes. Son intérêt est de spécifier les besoins et fonctionnalités de l'étape de diffusion des informations, étape qui n'est pas ou peu abordée dans les approches précédentes. Par contre, elle ne forme qu'une partie de la réponse au problème de la conscience de groupe, et doit être combinée avec d'autres techniques (par exemple de visualisation [FPSK98]) pour pouvoir être mise en pratique.

En ce qui concerne la surcharge d'informations, l'approche « diffusion de messages » permet le filtrage des informations. Ce filtrage peut avoir lieu soit au niveau de l'émetteur (le message n'est pas émis — filtrage sortant), soit au niveau du récepteur (le message n'est pas reçu — filtrage entrant). Ce filtrage permet de limiter les informations reçues, et donc aussi la surcharge d'informations.

Cependant, c'est généralement l'utilisateur qui spécifie lui-même les messages qu'il veut recevoir et ceux qu'il ne veut pas. Cela pose plusieurs problèmes. Tout d'abord, cette tâche peut être une surcharge de travail importante pour l'utilisateur, même si cette spécification ne doit être faite qu'une seule fois. Ceci est particulièrement vrai si le nombre de messages possibles est grand. L'utilisation de règles ou d'expressions régulières n'est pas une solution car elle demande un savoir-faire particulier supplémentaire à l'utilisateur. De plus, l'utilisateur ne sait pas forcément lui-même quels sont les messages qui seront pertinents pour son activité. Une mauvaise spécification peut ainsi lui faire manquer des

informations importantes. Finalement, la spécification ne change pas automatiquement lorsque l'activité de l'utilisateur change. L'utilisateur a alors le choix : garder un filtrage qui n'est peut-être plus adapté pour son activité présente et prendre le risque de manquer des informations pertinentes, ou bien repasser par la tâche de spécification du filtrage, ce qui représente une nouvelle surcharge de travail.

L'utilisateur ne doit pas avoir à spécifier son filtrage. Celui-là doit être fait le plus automatiquement possible. Or, l'approche « diffusion de messages » pose un problème pour un filtrage automatique. Lors de la collecte des informations, il est possible d'obtenir des méta-informations, sur la source des informations par exemple, ou bien sur les informations déjà reçues précédemment. Ces méta-informations peuvent servir à automatiser et à améliorer le filtrage. Par exemple, les informations déjà reçues et jugées non pertinentes peuvent servir à filtrer d'autres informations du même type. Cela n'est pas possible si la diffusion des informations est pensée sans aucun lien avec la collecte et la présentation des informations.

L'approche « diffusion de message » apporte des solutions techniques pour l'échange d'informations. Elle permet un filtrage des informations qui peut en réduire la surcharge. Mais comme ce filtrage doit être spécifié par l'utilisateur, cette baisse de la surcharge d'informations se fait au prix d'une surcharge de travail. De plus, la séparation de l'étape de diffusion du reste de la conscience de groupe est un obstacle à l'utilisation d'autres informations pour améliorer et automatiser le filtrage.

3.4 Travaux sur la *modélisation de la conscience de groupe*

Enfin, il existe un dernier courant, peu représenté mais souvent cité, dans la littérature. C'est celui de la « modélisation de la conscience de groupe ». Les deux principaux modèles cités sont le modèle « focus + nimbus » de Rodden [Rod96] et le modèle « réaction/diffusion » de Simone et Bandini [SB97].

Chacun de ces modèles a pour but de modéliser les relations entre les informations constituant la conscience de groupe et les membres du groupe. Ces relations représentent les zones d'intérêt des utilisateurs, et permettent donc de définir une mesure de la pertinence des informations en fonction de leur « distance » ou « force » avec ces zones. Cette mesure de pertinence peut ensuite être utilisée dans un mécanisme de filtrage comme décrit précédemment.

Le modèle « focus + nimbus » est basé sur la métaphore physique de l'espace. Cet espace est peuplé d'objets, représentant les personnes et les artefacts de la collaboration. Chaque objet est le sujet de deux zones : une zone de *focus*, et une zone de *nimbus*. La zone de *focus* contient les objets qui intéressent le sujet ; la zone de *nimus* contient les objets qui sont intéressés par le sujet. La zone *focus* représente l'attention du sujet, et la zone *nimus* son observabilité. Ainsi, le sujet peut facilement paramétrer ses zones d'intérêts en spécifiant les objets appartenant à ses deux zones. Plusieurs motifs spatiaux sont possibles. Ces motifs servent à mesurer le « niveau d'intérêt » existant entre deux objets.

Le modèle « réaction/diffusion » est lui basé sur la métaphore des champs physiques. Le modèle considère un espace peuplé d'entités représentant les personnes et les artefacts de la collaboration. La métaphore de la physique des champs est utilisé pour modéliser la

diffusion d'information entre les entités. Une entité émettrice propage un champ d'information en suivant une certaine fonction de diffusion, qui modélise ainsi l'observabilité de l'entité émettrice. Ce champ est perçu par les autres entités lorsqu'il dépasse un certain seuil de réception, qui modélise l'intérêt de l'entité réceptrice pour le champ en question. Lorsqu'un champ est perçu, il change l'état de l'entité réceptrice, qui peut par exemple devenir émettrice d'un autre champ, ou bien changer son seuil de perception. Le modèle permet ainsi de modéliser non seulement le niveau d'intérêt entre les objets, mais aussi leur comportement dynamique.

Ce courant est totalement orthogonal aux autres courants. Il ne se préoccupe pas de solutions, et ne possède pas d'implantations de tests. Il se concentre uniquement sur la modélisation des relations entre informations et participants.

Un des principaux problèmes des deux modèles cités est qu'ils ne fournissent qu'un moyen de décrire et d'utiliser les relations entre informations. Mais ces relations doivent tout de même être mises en place par le concepteur de l'outil de conscience de groupe ou bien par l'utilisateur. De plus, rien n'est dit sur l'adaptation de ces relations au cours du temps. Enfin, les critères retenus pour calculer la pertinence d'une information sont souvent simples (définis sous forme de zones), et ne prennent pas en compte des critères multiples (activité de l'émetteur + activité du récepteur + relations entre les deux personnes + type de l'information + ...). Finalement, ces modèles sont peu adaptés pour être utilisés dans la représentation des informations.

4 Problématique

Le problème de la surcharge d'informations est un problème récurrent dans les systèmes de conscience de groupe. Les informations échangées et présentées peuvent être très nombreuses, particulièrement lorsque le groupe est formé d'un grand nombre de membres, ou bien lorsqu'il y a beaucoup d'artefacts manipulés. Mais les ressources mentales de l'utilisateur, comme sa mémoire ou son attention, sont limitées. Il doit pourtant interpréter et intégrer ces informations pour pouvoir construire sa conscience de groupe. La surcharge d'informations survient lorsque l'utilisateur se retrouve en face d'un trop grand nombre d'informations à traiter. Les conséquences peuvent alors être multiples, suivant la stratégie adoptée par l'utilisateur. Ce dernier peut essayer de traiter un maximum d'informations, avec le risque de mobiliser pour cela trop de ressources cognitives (attention, temps, mémoire ...), qui seront autant de ressources non consacrées à son activité primaire (le travail en cours). L'utilisateur met alors plus de temps et d'efforts à tenter de construire sa conscience de groupe qu'à travailler vraiment. L'utilisateur peut aussi choisir de ne plus traiter toutes les informations, mais il risque alors de passer à coté d'informations importantes pour son activité. Sa conscience de groupe est alors moins complète, rendant sa coopération avec ses partenaires moins performante. Finalement, dans le cas d'une trop grande surcharge, le stress subi par l'utilisateur peut l'amener à rejeter en bloc l'ensemble des informations, rendant le système inutile. La surcharge d'informations est un problème sérieux pour l'utilisabilité des systèmes de conscience de groupe.

Dans le cas des informations de conscience de groupe, plusieurs problèmes peuvent être à l'origine de la surcharge d'informations. Il peut tout d'abord y avoir un problème de

passage à l'échelle. Certains systèmes, qui fonctionnent correctement lorsque la collaboration implique peu de personnes et d'artefacts, deviennent de moins en moins utilisable au fur et à mesure que le nombre de participants et d'artefacts manipulés augmente. Il y a alors trop d'informations à diffuser et à représenter.

Il peut aussi y avoir un problème de pertinence des informations. Certains systèmes diffusent des informations identiques à l'ensemble des membres du groupe. Or, certaines informations ne sont pertinentes que pour certaines personnes et dans certaines situations. La proportion d'informations intéressantes dans l'ensemble des informations reçue est alors trop basse. Le problème est similaire à celui posé par les courriels commerciaux non sollicités (les *spams*).

Enfin, le dernier problème peut venir de la représentation des informations. Certains systèmes représentent les informations d'une manière qui ne permet pas à l'utilisateur de saisir facilement et rapidement les informations essentielles. L'utilisateur doit alors passer plus de temps et consacrer plus d'attention pour interpréter et saisir les informations représentées.

Pour éviter les risques de surcharge d'informations, il faut limiter les informations présentées à l'utilisateur. Cela peut se faire à deux niveaux : en limitant les informations reçues aux seules informations pertinentes (filtrage), ou bien en utilisant une représentation qui facilite l'interprétation et l'intégration des informations (visualisation). Ces deux solutions ne doivent pas se faire en imposant une charge de travail supplémentaire à l'utilisateur.

Les différents courants existant dans la littérature sur la conscience de groupe ne traitent ce problème au mieux que partiellement. Le courant « périphérique » ne s'intéressent qu'à la visualisation. Le courant « graphique » propose un filtrage dépendant de l'utilisation d'un espace de travail partagé. Le courant « diffusion de messages » propose un filtrage, mais qui peut imposer une surcharge de travail à l'utilisateur. Enfin, le courant « modélisation » propose une manière de définir la pertinence, mais figée et demandant une charge de travail.

La double proposition de cette thèse concerne le problème de la surcharge d'informations. La première proposition est d'utiliser le contexte de travail de l'utilisateur pour adapter la conscience de groupe, afin d'éviter les surcharges d'informations. Le contexte est utilisé comme source des informations constituant la conscience de groupe. Il est aussi utilisé pour adapter automatiquement la propagation et la représentation de ces informations, afin que seules les informations pertinentes pour l'activité de l'utilisateur lui soient présentées. La deuxième proposition est d'utiliser la combinaison de deux techniques de visualisation pour représenter ces informations, de façon à permettre à l'utilisateur d'utiliser la visualisation la plus adaptée suivant l'activité à accomplir. Avec ces deux propositions, l'utilisateur ne reçoit que des informations pertinentes, présentées d'une manière adaptée à son activité, et cela de manière automatique, sans surcharge de travail.

Analyse cognitive

Le chapitre précédent a posé les bases de la problématique de la conscience de groupe. La conscience de groupe se construit à partir des échanges d'information intervenant lorsque les personnes travaillent en même temps dans le même lieu. Le but des outils de conscience de groupe est de permettre l'échange et l'accès à ces informations pour les personnes distribuées dans le temps et/ou dans l'espace. Cette thèse aborde le problème de la surcharge d'informations dans les outils de conscience de groupe : le principal inconvénient de ce type d'outil est qu'il surcharge l'utilisateur en lui fournissant un trop grand nombre d'informations, la plupart peu pertinentes pour son activité.

Cette problématique pose un certain nombre de questions : quelles sont les informations échangées lorsque les personnes sont en même temps dans le même lieu ? Comment ces informations sont-elles échangées ? Quels sont les mécanismes de construction de la conscience de groupe ? Quelles sont les étapes cognitives ? Comment les personnes présentes en même temps et dans le même lieu évitent-elles la surcharge d'informations ? Quelle doit-être la place d'un outil de conscience de groupe dans l'activité des personnes ? Quelles sont les limitations inhérentes à un tel outil ? Quels sont ses avantages ?

Ce chapitre a pour but d'essayer de répondre à ces questions à l'aide de plusieurs théories et outils relevant du domaine élargi de la psychologie cognitive. Une première partie fait l'analyse cognitive des mécanismes de construction de la conscience de groupe, pour en déduire un modèle cognitif. Une seconde partie analyse la place d'un outil de conscience de groupe parmi ces mécanismes, et les prérequis fonctionnels qui en découlent.

1 Construction de la conscience de groupe

Cette section a pour but d'analyser les mécanismes cognitifs mis en œuvre lors de la construction de la conscience de groupe. La psychologie de groupe [BT94] et la sociologie de groupe [Rou98], traitant plutôt du fonctionnement du groupe en tant que tel, ainsi que de la place et des rôles des participants dans le groupe, ne sont pas vraiment adaptés pour cette tâche. En effet, les mécanismes de construction d'un groupe, d'acceptation ou de rejet d'un membre du groupe, ou encore d'identification/individualisation au sein du groupe, sujets de la psychologie et de la sociologie de groupe, ne semblent pas pertinents pour la conscience de groupe. La psychologie cognitive semble par contre l'outil idéal pour cette tâche.

1.1 Psychologie cognitive

La psychologie cognitive s'intéresse aux processus cognitifs qui prennent place dans l'esprit de l'être humain. Un processus cognitif est un processus traitant, utilisant ou interagissant avec le savoir. La mémoire, l'attention, la représentation du savoir, la perception et la résolution de problème sont des exemples de processus cognitifs étudiés par la psychologie cognitive [FR89, PRS+94].

La principale hypothèse de la psychologie cognitive est de pouvoir utiliser la métaphore du traitement de l'information pour étudier les processus cognitifs. Pour les psychologues cognitifs, l'esprit se comporte globalement comme un ordinateur : il est composé (au point de vue logique) de plusieurs modules fonctionnels, ayant pour but de traiter l'information. L'information arrive du monde extérieur par l'intermédiaire des organes sensoriels, est traitée par les centres perceptifs, puis est propagée aux centres de la mémoire et de la connaissance. C'est dans ces derniers que des processus comme le raisonnement et la résolution de problème ont lieu. Finalement, les résultats du traitement sont envoyés aux modules expressifs, en charge de produire les effets externes (paroles et/ou gestes, par exemple). Des résultats de neurosciences comme la neurobiologie ont confirmé certaines théories de la psychologie cognitive.

Neisser [Nei76] a décrit, dans le cadre de la psychologie cognitive, un modèle décrivant la relation entre la cognition et la perception (cf. fig. 1). Ce modèle met notamment l'accent sur l'influence réciproque existant entre le savoir d'une personne — ses connaissances propres et sa représentation mentale du monde extérieur —et sa perception du monde extérieur. Le modèle se présente comme un cycle reliant le monde extérieur, le savoir de la personne et ses actions : le savoir de la personne dirige ses actions, qui permettent l'exploration du monde extérieur, dont la perception modifie en retour le savoir de la personne, qui ainsi va diriger de nouvelles actions d'explorations, et ainsi de suite.

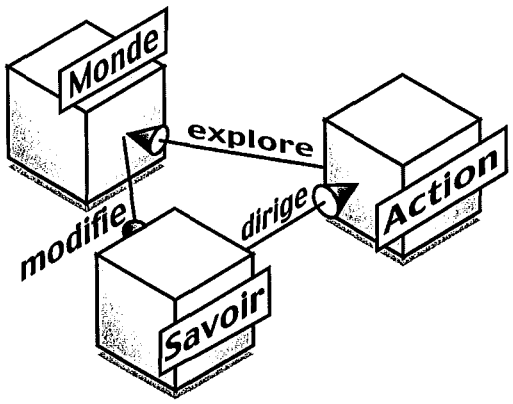

FIG. 1 – Le modèle de Neisser.

Ce modèle, décrivant un cadre général d'interactions entre cognition et perception, a fait l'objet de plusieurs adaptations à des domaines ou à des circonstances spécifiques. Les travaux de Jager Adams et al. [JATP95], par exemple, appliquent le modèle de Neisser

au cadre particulier de la « conscience de la situation » [4] (*Situation Awareness — SA* en anglais). Dans le même cadre, le travail d'Endsley [End95] détaille le modèle de Neisser en y distinguant trois étapes (cf. fig. 2) :

- la perception des éléments de l'environnement ;
- la compréhension de la situation courante ;
- la prédiction du futur état de l'environnement.

Enfin, Gutwin et Greenberg [GG01] utilisent ces trois étapes comme structure de leur analyse afin d'adapter le modèle de Neisser au cadre spécifique de la conscience de groupe dans les espaces de travail partagés synchrones.

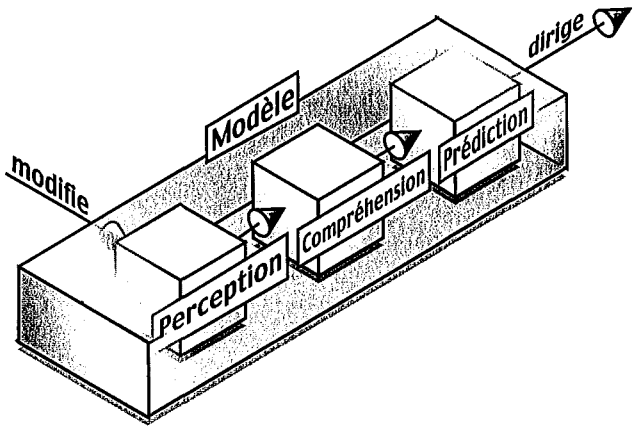

FIG. 2 – Les trois étapes décrites par Endsley.

Contrairement aux travaux de Gutwin et Greenberg qui se situent dans un cadre très spécifique (les espaces de travail partagés synchrones), cette thèse aborde la conscience de groupe dans son cadre le plus général. Dans ce cadre, le modèle décrit les interactions entre l'environnement du groupe, la conscience de groupe et les activités des membres du groupe. Les trois étapes décrites par Endsley deviennent respectivement (cf. fig. 3) :

- la perception des informations de conscience de groupe ;
- la construction ou la modification de la conscience de groupe ;
- l'action de l'utilisateur au sein du groupe.

La suite de l'analyse de la conscience de groupe suit la structure suggérée par ces trois étapes.

1.2 Perception des informations

En suivant le modèle de Neisser, la première étape cognitive dans le mécanisme de construction de la conscience de groupe est la perception d'informations. Mais quelles sont

[4] La « conscience de la situation » peut être considérée comme l'ancêtre de la conscience de groupe. Appartenant au domaine de l'étude, d'abord militaire puis civile, de l'aviation et plus particulièrement des capacités des pilotes à interagir dans un environnement hautement dynamique, la conscience de la situation regroupe l'ensemble des connaissances et aptitudes des pilotes à savoir ce qui se passe à chaque instant lors d'un vol, afin de pouvoir réagir rapidement et correctement dès qu'un problème apparaît.

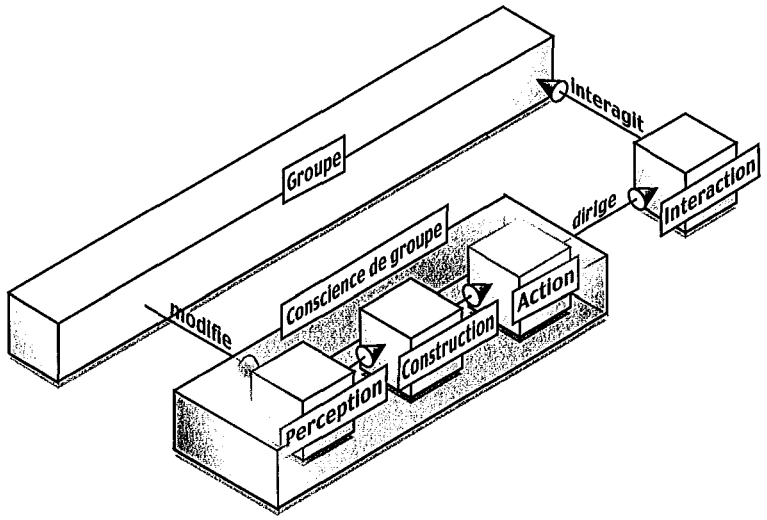

FIG. 3 – Le modèle de Neisser et les trois étapes de Endsley adapté au cadre de la conscience de groupe.

ces informations perçues qui permettent de former la conscience de groupe ? Le chapitre précédent a donné une piste sur ces interrogations : il s'agit des informations échangées par les autres membres du groupe sur leurs activités dans le groupe. Pour comprendre cette première étape, il faut donc analyser le type des informations échangées.

Dans les travaux de psychologie s'intéressant aux groupes et plus particulièrement aux communications à l'intérieur du groupe, on trouve une notion commune et centrale pour le concept de groupe : la notion de contexte partagé.

Ainsi, d'après la psychologie de groupe, l'un des attributs d'un groupe est l'existence de références partagées entre les membres du groupe [BT94, Rou98]. Ces références, provenant de mémoires et de moments vécus ensembles ou racontés, forment un contexte commun aux différents membres du groupe.

Ce concept de contexte partagé se retrouve dans la notion de « base commune »(*common grounding*) introduite par Clark [CB91]. Pour Clark, le contexte est nécessaire à la communication entre personnes. Lorsque nous communiquons, nos phrases n'ont pas un sens unique et objectif. Le sens d'une phrase est construit par la personne qui l'entend ou la lit, en remettant la phrase dans son propre contexte, et plus particulièrement dans la partie du contexte que la personne pense être partagée avec l'auteur de la phrase. Ainsi, le sens d'une phrase est construit dans le cadre du contexte partagé entre les personnes communicantes. De plus, chaque média de communication possède ses propriétés propres qui influencent la communication [CB91].

La notion de contexte partagé se retrouve aussi dans les travaux de Bateson et plus généralement dans les travaux sur la « Nouvelle Communication ». L'expression « Nouvelle Communication », inventée par John H. Weakland, désigne un modèle de communication qui n'est pas basé sur la métaphore du télégraphe, comme la théorie de l'information de Shannon [Dio97, Sha48], mais sur la métaphore de l'orchestre, où chaque participant fait

partie de la communication, par l'ensemble de son comportement : par la voix, le geste, le comportement social... [Win00]. Ce modèle est commun à un groupe de chercheurs (Bateson, Birdwhistell, Goffman, Hall, Scheflen, Watzlavick...), venant de différents domaines (psychologie, psychiatrie, anthropologie, sociologie), et formant ensemble le dénommé « Collège Invisible ». La partie psychiatrique du groupe a pour sa part reçu le nom d'« École de Palo-Alto ».

L'idée principale de la « Nouvelle Communication » est que la communication n'est pas un envoi à sens unique d'un message entre un expéditeur et un destinataire. La communication est quelque chose qui a lieu entre tous et de toutes les façons possibles, dès lors qu'il y a deux personnes ensemble. Tout ce que nous faisons lorsque nous sommes en présence d'autrui fait partie de la communication. Bien sûr, nous communiquons avec des mots, mais aussi avec nos gestes (Birdwhistell) et par notre comportement (Goffman [Gof73a, Gof73b, Gof74]). Même notre conception du temps et de l'espace font partie de la communication (Hall [Hal71, Hal79, Hal84a, Hal84b]). Comme le dit Watzlavick : « on ne peut pas ne pas communiquer ». Chaque partie de notre comportement (détaillée par Schelfen, entre autres) est un code, interprété dans le contexte de l'interaction, qui permet d'organiser nos interactions personnelles et interpersonnelles.

Il découle de tout cela que les informations permettant de construire la conscience de groupe proviennent du contexte de l'interaction avec les autres membres du groupe, c'est-à-dire par exemple des informations sur le succès d'une tâche, sur les activités d'un autre membre, sur la coordination, mais aussi sur l'état moral des autres (fatigué, stressé, pressé), sur leurs occupations (surchargé de travail)... De plus, ces informations sont transmises par différents canaux (oral, écrit, gestuel...), dont chacun a ses propriétés et ses influences sur les différents types d'échanges (formels, informels, conscients, inconscients, déductifs, volontaires...).

Dans la littérature sur la conscience de groupe, un certain nombre de travaux concernent le type des informations participant à la construction de la conscience de groupe [Gut97, MB97, Sch02]. L'apport principal de ces travaux est de structurer l'espace des informations, en utilisant les questions habituelles : « qui », « quoi », « quand », « où », « comment », « pourquoi » ... Ces questions permettent de choisir et classer les informations à retenir, propager et présenter pour construire la conscience de groupe.

1.3 Construction de la conscience de groupe

Les informations servant de base à la conscience de groupe proviennent donc du contexte de l'interaction avec les autres membres du groupe. Ce cadre peut être structuré en utilisant les questions habituelles : « qui », « quoi », « quand », « où », « comment », « pourquoi » ... Ces informations sont transmises grâce à différents médias de communications (oral, écrit, gestuel...) et suivant différents types d'échanges (formels ou informels, conscients ou inconscients, ...). Il reste à savoir comment la conscience de groupe se construit à partir des informations provenant du contexte.

La théorie de la Gestalt est pertinente dans ce cadre car elle aborde notamment le problème du traitement des informations en fonction du contexte. La théorie de la Gestalt a été fondée à la fin du XIX^{e} siècle en réaction à la théorie du béhaviorisme en vigueur à l'époque. Les principaux fondateurs de cette théorie sont trois psychologues allemands,

Wertheimer, Köhler and Koffka.

La principale hypothèse du béhaviorisme est que l'esprit doit être traité comme une boîte noire, car personne ne peut vérifier ce qui arrive réellement à l'intérieur. Donc la seule technique d'étude valable pour un psychologue est d'étudier l'esprit uniquement par son comportement (*behaviour* en anglais, d'où le nom de ce mouvement), c'est-à-dire par ses réactions à des stimuli spécifiques. Afin d'obtenir des résultats reproduisibles et réutilisables, les stimuli complexes sont décomposés en stimuli élémentaires, et l'étude de l'esprit ne porte que sur les réponses à ces stimuli élémentaires. L'hypothèse implicite du béhaviorisme est que la réaction à un stimuli complexe est égale à la somme des réponses aux stimuli élémentaires le composant.

La principale idée de la théorie de la Gestalt est de reconnaître qu'il est possible dans certains cas que la somme des éléments ait des propriétés qui soient différentes de la somme des propriétés de ses éléments. Dans ce cas, l'ensemble des éléments ne peut pas être expliqué par l'étude de chacun de ses éléments pris séparément. L'ensemble doit être étudié en tant que tout, comme une *gestalt* (le mot *gestalt* est un mot allemand qui a pour signification la forme, la structure) [Koh47]. La théorie de la Gestalt a parfois été nommée « psychologie de la forme ».

Les résultats les plus connus de la théorie de la Gestalt ont trait au domaine de la perception, et plus particulièrement de la vision. Notre esprit ne traite pas les informations sensorielles de manière indépendantes. Il les structure au contraire par rapport au contexte des autres informations, et peut-être aussi par rapport à notre savoir personnel. Certaines informations sont même agrégées avant que la partie consciente de notre esprit ne les traite. C'est la raison pour laquelle nous voyons des « objets », et non pas de simples zones colorées (cf. fig. 4), ou bien encore pourquoi nous sommes capables de reconnaître une mélodie, même si elle a été transposée d'un ton. Les informations sont traitées ensemble, avec leur contexte, et non pas indépendamment les unes des autres : les informations « élémentaires » (les stimuli résultant des photons) sont agrégées en informations de haut niveau (points, lignes, objets...) en fonction du contexte (les autres stimuli et les connaissances propres). La biophysiologie et la neurobiologie nous apprennent que, pour la vision, la structuration des informations est faite par des centres spécialisés présents entre nos yeux et la partie « consciente » de notre cerveau.

Une fois agrégées à partir des informations élémentaires, les informations de haut niveau atteignent les centres de la mémoire et de la connaissance, où elles vont être interprétées. Or, d'après les travaux de Clark [CB91] et de la « Nouvelle Communication »(et plus particulièrement de Bateson) [Win00], l'interprétation des informations reçues nécessite l'utilisation du contexte, car une information prise hors contexte n'a pas de sens intrinsèque. Puisque le contexte est contenu dans la mémoire de la personne, il faut connaître la structure de la mémoire.

Il existe sur le sujet de la mémoire et de sa structure de nombreux travaux. Ceux de Sanford et Garrod [SG81], tels que décrits par Adams et al. [JATP95] dans le cadre de la « conscience de situation », structurent la mémoire en quatre « conteneurs » (*bins*) (cf. fig. 5). Ces quatre conteneurs représentent quatre types de mémoires spécifiques, dans lesquelles se retrouvent les notions connues de mémoire à court terme, mémoire à long terme, et mémoire de travail.

Au premier plan se trouve la mémoire d'« attention explicite » (*Explicit Focus*). Cette

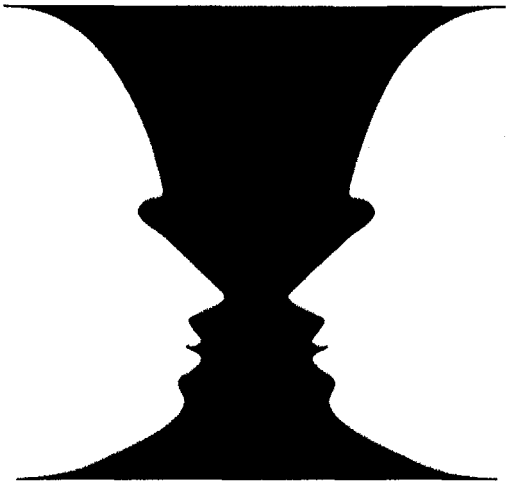

FIG. 4 – Un exemple d'application de la Gestalt : la figure représente soit un vase, soit deux visages, suivant ce que le cerveau interprète comme étant le fond de la figure.

mémoire, équivalente à la mémoire de travail, contient les connaissances immédiatement pertinentes pour la tâche en cours. Elle ne peut contenir que peu d'éléments, mais l'accès à ces éléments est direct. Elle est suivie par la mémoire d'« attention implicite » (*Implicit Focus*), qui contient la représentation complète des connaissances connexes à l'activité en cours. Cette mémoire peut contenir plus d'éléments que la mémoire d'« attention explicite », mais l'accès y est plus lent. Ces deux mémoires représentent la mémoire à court terme, et sont suivies par deux autres mémoires représentant la mémoire à long terme. La première de ces mémoires est la mémoire « épisodique à long terme » (*Long-term Episodic Memory*) ; elle contient les structures des connaissances générales nécessaires à l'activité globale de la personne. Finalement, la mémoire « sémantique à long terme » (*Long-term Semantic Memory*) contient le reste du savoir de la personne.

Plus la mémoire est externe, plus elle contient d'informations, mais plus l'accès à l'information demande du temps et de l'attention. Inversement, plus la mémoire est interne, plus l'accès à l'information est immédiat, mais moins elle peut contenir d'informations. Les deux cas extrêmes sont la mémoire sémantique à long terme, contenant toutes les informations, et la mémoire d'attention explicite, d'accès immédiat. Ces différentes mémoires représentent les différents types de contextes servant à l'interprétation des informations reçues.

La différence d'accès entre les différentes mémoires se traduit par un filtrage des informations, en fonction des ressources cognitives disponibles de la personne. En effet, pour pouvoir être interprétée, l'information de haut niveau doit être mise en relation avec l'un des contextes. Mais l'accès à ce contexte dépend du contexte lui-même ainsi que des ressources cognitives que la personne veut bien allouer à cette tâche. Ainsi, une information concernant l'activité directement réalisée par la personne est mise en relation avec le contenu de la mémoire d'attention explicite, dont l'accès se fait sans effort. Cette information est alors directement interprétée. Inversement, une information qui ne peut être mise

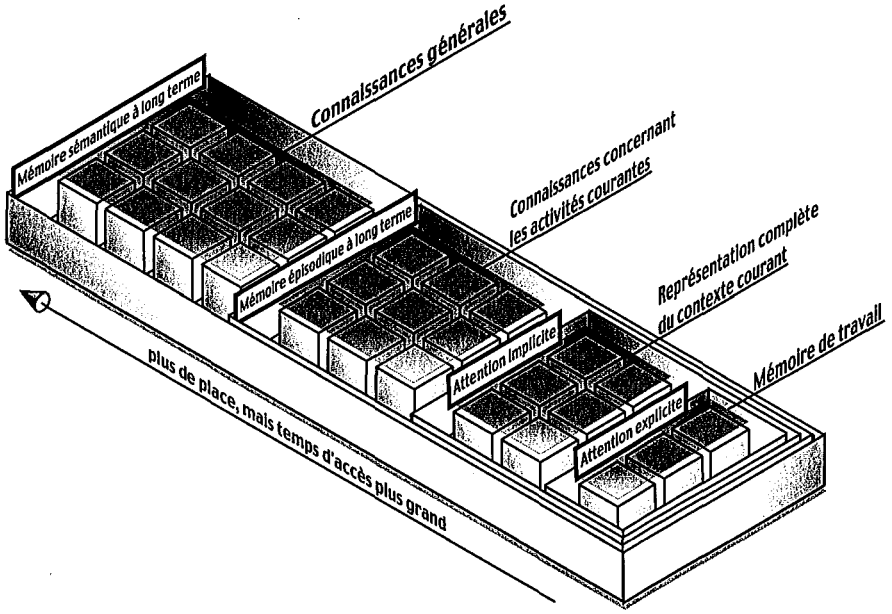

FIG. 5 – La structure de la mémoire suivant Sanford et Garrod, tels que décrit par Adams et al. [JATP95]

en relation avec aucun des trois premiers contextes n'est interprétée que si la personne a assez de temps et d'attention à lui accorder pour la mettre en relation avec l'ensemble de ses connaissances, c'est-à-dire sa mémoire sémantique à long terme.

À titre d'illustration, prenons une personne en train de rédiger un article. Un message d'erreur de son logiciel d'édition de texte concernant une erreur de frappe, par exemple, est interprété directement. En effet, cette information, directement pertinente pour son activité en cours, est mise en relation avec la mémoire d'attention explicite, et donc traitée directement sans effort. De même, la perception de l'arrivée d'un courrier électronique du co-auteur de l'article est interprétée sans trop de problèmes, sauf si l'utilisateur est profondément concentré dans l'écriture d'une phrase. C'est en effet une information connexe à l'activité en cours, donc mise en relation avec la mémoire d'attention implicite, sauf si vraiment toutes les ressources cognitives sont déjà prises. Par contre, entendre son chef discuter dans le couloir n'est pris en compte que si la personne n'est pas trop concentrée sur son travail (mémoire épisodique à long terme), et le passage d'un avion dans le ciel est ignoré, sauf si la personne est en pause (mémoire sémantique à long terme) (cf. fig. 6).

Cet exemple n'est qu'une illustration, dans un cas très particulier, du modèle développé. Le contexte y joue un très grand rôle : si par exemple la personne attend un message particulier de son chef (pour une augmentation ...), il est clair que cette information se retrouvera sûrement dans l'une des mémoires à court terme, et ainsi la discussion du chef dans le couloir sera traitée quasiment à tous les coups. Ce modèle ne prend pas non plus en compte les mécanismes mnémoniques facilitant le rappel et l'accès à la mémoire à long terme [Dor03, FR89, Pin97].

La structure de la mémoire impose donc un filtrage des informations, en fonction

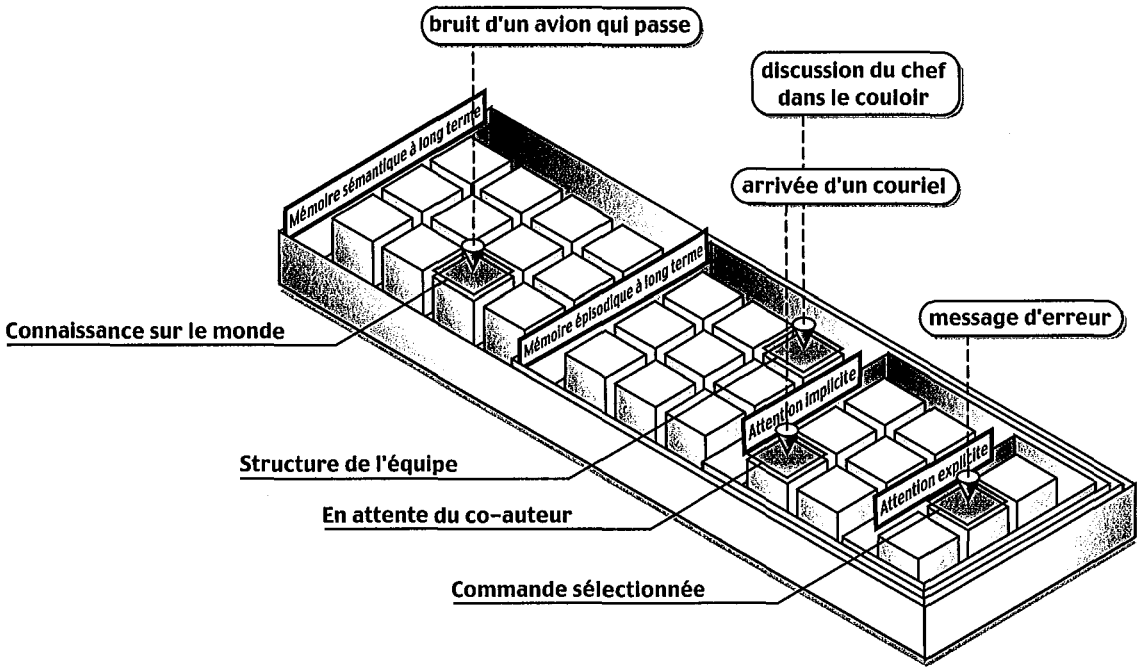

FIG. 6 – Un exemple de filtrage suivant le contexte.

de l'information elle-même, du contexte et des ressources cognitives de la personne. Les informations ayant passé le filtrage sont ensuite interprétées par rapport au contexte auquel elles sont liées [CB91, Win00]. L'interprétation peut être différente d'une personne à l'autre. Par exemple, l'information « le fichier a été changé » peut prendre deux sens différents suivant que la personne qui la reçoit est un simple lecteur ou bien l'auteur du fichier. Le lecteur peut l'interpréter comme « il y a une nouvelle version du fichier à lire ». L'auteur, lui, peut l'interpréter comme un problème de coordination, car cela signifie que quelqu'un a travaillé sur son fichier. Le sens d'une information dépend du contexte de la personne qui la reçoit.

Une fois l'information interprétée, elle est internalisée et devient une nouvelle partie du contexte. En effet, le contexte de la personne contient, à différent niveaux, toutes les informations connues de la personne. La nouvelle information qui vient d'être interprétée en fait donc partie. Ainsi, le contexte est modifié à chaque fois qu'une information est interprétée, et c'est par rapport à ce contexte mis à jour que la personne va décider de ses actions.

En résumé, la théorie de la Gestalt indique que les informations sont traitées ensemble, et non pas séparément. Elles sont agrégées en informations de plus haut niveau, et ce sont ces informations de haut niveau qui seront interprétées. Afin de pouvoir être interprétées, ces informations de haut niveau sont replacées dans leur contexte : elles sont mises en relation avec les informations déjà présentes en mémoire. Ces informations en mémoire représentent le contexte de la personne. Comme il existe plusieurs types de mémoires, aux temps d'accès et aux capacités différentes, la mise en relation des informations reçues

avec celles disponibles dépend non seulement du contenu des mémoires mais aussi de l'activité de l'utilisateur et de son état cognitif. Cette mise en contexte des informations haut niveau joue ainsi le rôle de filtre. Enfin, les informations sont interprétées en fonction des informations de la mémoire avec lesquelles elles ont été mises en relation. Finalement, les informations interprétées sont sauvegardées en mémoire, formant ainsi avec celles déjà présentes le nouveau contexte de la personne.

En synthétisant tout ceci, quatre étapes peuvent être distinguées dans la construction de la conscience de groupe :

Agrégation : les informations élémentaires sont agrégées en informations complexes grâce à l'utilisation du contexte ;

Filtrage : les informations complexes sont filtrées par rapport à l'information elle-même, au contexte et aux ressources cognitives de la personne ;

Interprétation : les informations ayant passé le filtrage sont interprétées par rapport au contexte auquel elles sont liées ;

Internalisation : les informations interprétées sont internalisées dans le contexte.

1.4 Actions

L'utilisateur décide des actions à entreprendre en fonction de ses connaissances propres et de sa vision du reste du groupe. La conscience de groupe de l'utilisateur influence donc ses actions, directement ou indirectement. Deux types d'action en particulier peuvent être distinguées par leur lien direct avec la conscience de groupe :

- les actions coopératives avec les autres membres du groupe ;
- les actions de demande d'informations supplémentaires.

Les actions coopératives avec les autres membres du groupe sont rendues plus efficaces grâce à la conscience de groupe [GG99]. C'est le but même de la conscience de groupe.

Les actions de demande d'informations supplémentaires sont comparables aux actions d'exploration décrites par Neisser [Nei76]. Le but de ces actions est d'améliorer ou de mettre à jour la conscience de groupe en recherchant des informations particulières, le plus souvent en relation avec les nouvelles informations venant d'être reçues.

1.5 Modèle cognitif

En résumé, les parties précédentes ont permis de dégager un modèle des étapes cognitives de la construction de la conscience de groupe. Prenant pour base le modèle de Neisser [Nei76], la construction de la conscience de groupe se fait en trois grandes étapes (cf. fig. 7) :

Perception des informations. Les informations permettant de construire la conscience de groupe proviennent du contexte des activités des membres du groupe. Ces informations peuvent être classifiées suivant les six questions habituelles : *qui*, *quoi*, *quand*, *où*, *pourquoi*, *comment*. Elles sont transmises par différents canaux, chacun ayant ses propriétés et son influence sur la communication, elle-même pouvant être de plusieurs types.

Construction de la conscience de groupe. Une fois les informations perçues, la conscience de groupe est construite en quatre étapes :
- Agrégation des informations élémentaires en informations complexes ;
- Filtrage des informations complexes ;
- Interprétation des informations ayant passé le filtrage ;
- Internalisation des informations interprétées dans le contexte.

Action de l'utilisateur. Enfin, lorsque la conscience de groupe est construite, l'utilisateur peut avoir deux types d'action. Il peut utiliser la conscience de groupe pour améliorer sa coordination avec les autres membres, ou bien il peut activement rechercher une information pour améliorer sa conscience de groupe.

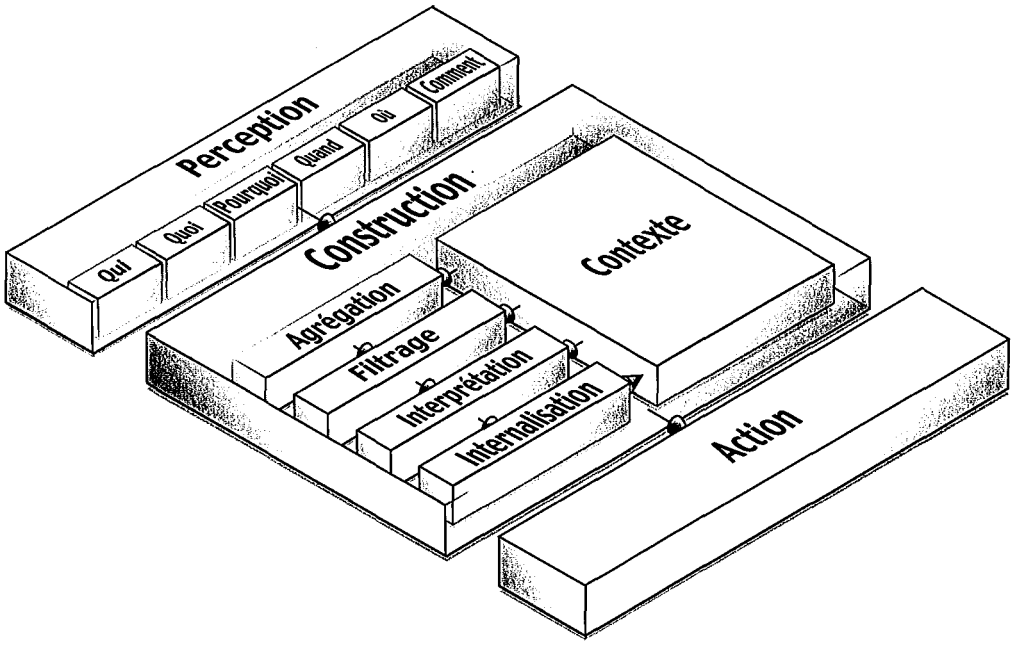

FIG. 7 – Un modèle des étapes cognitives dans la construction de la conscience de groupe.

2 Place du système de conscience de groupe

La section précédente a explicité les différents mécanismes cognitifs mis en œuvre dans la construction de la conscience de groupe. Pour compléter l'analyse, il reste à déterminer la place d'un outil de conscience de groupe parmi ces mécanismes, place dont découlera un ensemble de prérequis fonctionnels pour cet outil. Cette dernière étape nécessite l'utilisation d'outils de la psychologie cognitive aptes à analyser la relation entre cognition et contexte. La théorie de l'Action Située (*Situated Action*) [Suc87], la théorie des Cadres Locaux (*Locales Frameworks*) [FMK96], la théorie de l'Activité (*Activity Theory*) [Leo78, Nar96, Bød89] et la théorie de la Cognition Distribuée (*Distributed Cognition*) [HHK00, Hut95] répondent à ce critère. Cependant, les deux dernières théories, la

théorie de l'Activité et la Cognition Distribuée, ont comme avantage de traiter explicitement de la place des outils dans les processus cognitifs.

La suite de cette section utilise donc les outils fournis par les théories de l'Activité et de la Cognition Distribuée pour analyser la place que doit avoir un système de conscience de groupe dans le processus cognitif de la conscience de groupe.

2.1 La théorie de l'Activité

La théorie de l'Activité découle des travaux de Vygotsky concernant le rôle de la société sur la construction de l'esprit [Vyg78], mais c'est l'un de ses étudiants, Leontiev qui formula la théorie elle-même [Leo78]. La théorie de l'Activité a été récemment utilisée dans le domaine de l'Interaction Homme-Machine (*Computer-Human Interaction - CHI*) [Nar96, Bød89] et dans le domaine du Travail Coopératif Assisté par Ordinateur [Kuu92].

La principale hypothèse de la théorie de l'Activité est que la conscience d'une personne est le produit de ses interactions sociales avec autrui. Contrairement à la psychologie cognitive classique, qui considère cette partie sociale comme accessoire, la théorie de l'Activité accorde un rôle crucial à l'influence de la société sur la cognition. Ainsi, l'esprit humain ne peut être compris s'il est pris seul : il doit être étudié dans son contexte social. La théorie de l'Activité propose pour cela un certain nombre de concepts, qui sont autant d'outils pour l'analyse des processus socio-cognitifs :

Cadre des Activités : Le cadre d'exécution des activités cognitives humaines peut être décrit par un modèle en triangle (cf. fig. 8). Dans ce modèle, l'*Activité* représente l'ensemble des actions à effectuer pour atteindre un but particulier. Cette activité est réalisée par un *Sujet*, qui peut être une personne ou bien un groupe. Le but à atteindre est matérialisé par un *Objet*, qui ne désigne pas forcément un objet réel : dans cette section par exemple, l'objet de l'activité sera la conscience de groupe. Finalement, l'activité est réalisée grâce à un *Outil*, médiateur entre le sujet et l'objet. L'outil peut être matériel (par exemple un marteau) ou psychologique (par exemple une déduction logique). L'*Activité* représente ainsi le contexte des actions cognitives.

Structure hiérarchique des Activités : Une activité peut être décomposée en trois niveaux : le niveau de l'*Activité*, le niveau des *Actions*, et le niveau des *Opérations* (cf. fig. 9).

Activité : une activité représente un ensemble d'actions ayant pour but un objet (matériel ou non), principalement de haut niveau (comme chasser, produire un logiciel...). L'objet est décomposé en sous-buts plus simples, réalisés par les actions correspondantes.

Actions : une action est un processus conscient, dirigé par un but. Le but est une représentation mentale du résultat à obtenir. Une action peut se décomposer en sous-actions plus simples, son but étant alors lui aussi décomposé en sous-buts correspondants. Finalement, une action se décompose en opérations.

Opérations : une opération représente une tâche automatique s'effectuant suivant les conditions dans lesquelles le but a été donné.

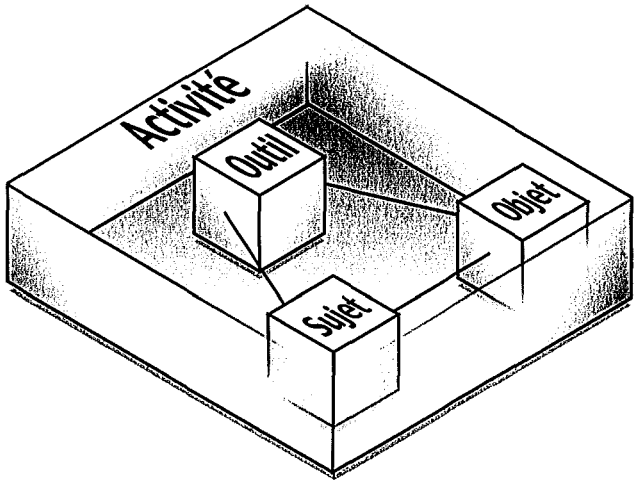

FIG. 8 – Le cadre liant l'Outil, l'Objet et le Sujet.

En résumé, une activité a pout but un objet. Une activité est un ensemble d'actions ayant pour buts des sous-buts de l'objet. Une action peut être décomposée en sous-actions plus simples. Finalement, une action est constituée d'opérations, c'est-à-dire de processus automatiques dirigés par le but et les conditions. Par exemple, conduire peut être vu comme une activité, ayant pour objet d'aller d'un point à un autre. Un exemple d'action peut être de changer de vitesse, dans le but d'accélérer. Une des opérations sera alors d'appuyer sur l'embrayage pendant que le levier de vitesse est manipulé.

La structure activité-action-opération n'est pas statique mais dynamique. La répétition d'une action tend à la rendre automatique. Elle devient alors une opération. Inversement, une opération normalement automatique peut, si son exécution rencontre un problème imprévu, devenir une action consciente.

Internalisation/externalisation : Il existe deux types d'activités. Les activités internes prennent place dans l'esprit de la personne, à l'aide d'outils psychologiques (par exemple, calculer mentalement une multiplication). Les activités externes prennent place dans le monde extérieur, à l'aide d'outils réels (par exemple, calculer une multiplication avec une calculatrice). Le status interne-externe n'est pas statique mais dynamique. Une activité externe peut être internalisée pour être simulée dans l'esprit de la personne. Une activité interne peut être externalisée pour être partagée avec d'autres. Pour la théorie de l'Activité, ce passage constant entre internalisation et externalisation est cruciale pour la cognition humaine.

Médiation des outils : Dans les activités cognitives, les outils assurent la médiation entre le sujet et le but à atteindre. Non seulement ils permettent à une personne d'interagir avec le monde, mais en plus ils façonnent cette interaction. Ainsi, leurs utilisations reflètent certaines conventions sociales. Il y a deux catégories d'outils : les outils techniques, pour les objets physiques, et les outils psychologiques, pour les objets sociaux. L'étude des outils et de leur médiation est très importante pour la

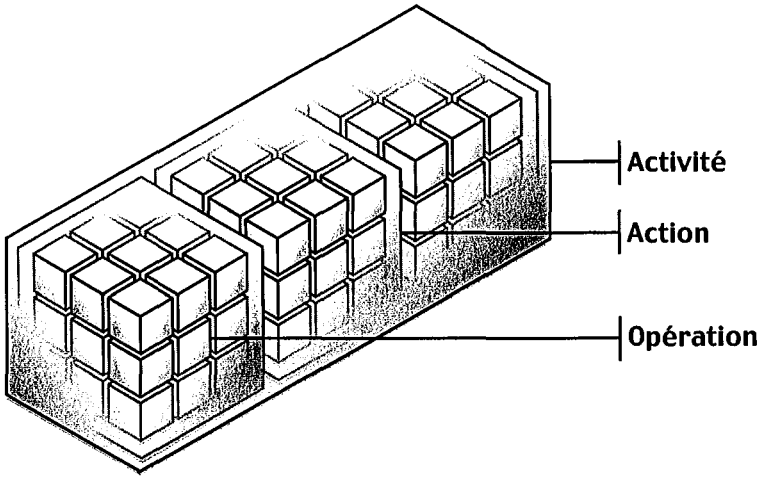

FIG. 9 – La structure hiérarchique des Activités.

théorie de l'Activité.

Développement : Les activités cognitives humaines ne sont pas figées, mais au contraire en constante évolution. Par exemple, une action peut devenir une opération, une activité interne peut être externalisée, etc. Les analyses faites à l'aide des concepts précédents doivent donc prendre en compte les évolutions possibles des objets analysés.

Tous ces concepts permettent d'analyser le rôle et la place d'un outil de conscience de groupe parmi les mécanismes cognitifs de la construction de la conscience de groupe.

Il est possible en premier lieu de définir la *construction de la conscience de groupe* comme une activité. Le *sujet* accomplissant cette activité est un membre du groupe, et l'*objet* matérialisant le but à obtenir est la conscience de groupe elle-même. Le système de conscience de groupe joue alors le rôle de l'*outil*, assurant la médiation entre le sujet et le but à atteindre, c'est-à-dire ici entre le membre du groupe et la conscience de groupe. Or, il a été vu précédemment qu'une personne construit sa conscience de groupe à partir des informations perçues sur les activités du groupe. Le rôle d'un outil de conscience de groupe est donc de fournir les informations nécessaires au membre du groupe.

En fait, construire sa conscience de groupe n'est pas un but en soi. C'est un sous-but d'une activité de plus haut niveau, autrement dit de l'activité principale du membre du groupe (son travail ou sa tâche au sein du groupe). De même, l'activité de chaque membre peut être vue comme une composante d'une activité d'encore plus haut niveau, l'activité du groupe. C'est l'activité du groupe qui fournit le contexte nécessaire aux activités de chacun des membres du groupe. D'après ce qui a été dit précédemment, ce contexte de l'activité du groupe est donc la source des informations constituant la conscience de groupe. Le contexte de l'activité du groupe doit donc être la source des informations analysées puis fournies par l'outil de conscience de groupe.

L'activité du groupe est composée de l'ensemble des activités de ses membres. Pour un membre particulier, cela veut dire son activité propre et l'ensemble des activités des autres membres du groupe. L'activité d'un membre fait donc partie des informations constituant

la conscience de groupe. L'outil de conscience de groupe doit donc fournir à l'utilisateur non seulement les informations concernant les activités des autres membres du groupe, mais aussi celles concernant sa propre activité.

L'activité de construction de la conscience de groupe peut notamment être vue comme une *action* ou une *opération* de l'activité principale de la personne. Or, l'outil de conscience de groupe fait partie, par sa médiation, de cette activité. Les informations le concernant, comme son état ou son utilisation, font donc aussi partie des informations qu'il doit fournir à l'utilisateur.

L'activité de construction de la conscience de groupe peut donc être vue comme une *action* ou une *opération* de l'activité principale de la personne. C'est une *opération* lorsqu'elle est réalisée de manière automatique par la personne, par exemple lorsque la personne capture et intègre de manière *périphérique* les informations de conscience de groupe. C'est une *action* lorsqu'elle résulte d'une action active et voulue de la personne, par exemple lorsque l'utilisateur recherche une information précise sur l'activité d'un autre membre du groupe. L'outil de conscience de groupe doit donc permettre à l'utilisateur de construire sa conscience de groupe de manière passive (opération), par exemple en présentant ses informations de manière périphérique, mais aussi de manière active (action), par exemple en permettant à l'utilisateur d'accéder, lorsqu'il en fait la demande, à plus de détails ou bien à une information particulière.

Il faut aussi prendre en compte le fait qu'une opération se fait dans le cadre d'une action possédant un but précis. Sous forme d'opération, la construction de la conscience de groupe doit donc être automatique pour que l'utilisateur se concentre sur l'action en cours. Sous forme d'action, la construction de la conscience de groupe doit permettre de répondre au but précis formulé par l'utilisateur. Enfin, l'action doit pouvoir se décomposer de manière simple en opérations : l'accès aux détails des informations doit pouvoir se faire de manière simple une fois que l'utilisateur en a fait la demande.

En ce qui concerne les activités des autres membres du groupe, un membre ne peut avoir accès qu'aux parties externalisées de ces activités. Inversement, le but de la construction de la conscience de groupe est d'internaliser les activités des autres, de façon à les intégrer à son propre contexte. L'outil de conscience de groupe doit donc permettre à l'utilisateur de définir (externaliser) les parties de l'activité auxquelles les autres membres du groupe peuvent accéder. Il doit aussi lui permettre de comprendre et d'intégrer (internaliser) les activités des autres à partir des informations fournies.

Finalement, toutes les activités sont dynamiques et en perpétuel développement. L'outil de conscience de groupe doit donc être lui aussi dynamique. Il doit pouvoir gérer des informations nouvelles ou changeantes. Il doit pouvoir prévenir l'utilisateur qu'il a reçu de nouvelles informations ou bien que certaines informations ont changé. Il doit aussi pouvoir indiquer si ces informations sont nouvelles ou modifiées.

2.2 La Cognition Distribuée

Si la psychologie cognitive est représentée par « le savoir est dans la tête », alors la cognition distribuée est représentée par « le savoir est dans le monde ». Le cadre théorique (*framework*) de la cognition distribuée a été développé par Hutchins et ses collègues [HHK00, Hut95], principalement sur la base d'études ethnologiques sur des tableaux

de bord et des systèmes de navigation d'avions et de navires.

La principale idée de la cognition distribuée est que, lorsque des utilisateurs travaillent ensemble, le savoir et le processus cognitif sont distribués entre les utilisateurs et l'environnement. Ainsi, pour comprendre le processus cognitif, il faut prendre en compte non seulement les utilisateurs, mais aussi leur environnement, c'est-à-dire principalement les outils utilisés. Les outils sont utilisés comme médiateurs pour la cognition en externalisant les représentations et en supportant la coordination entre les utilisateurs.

Comme la théorie de l'Activité [Nar98], la Cognition Distribuée propose un certain nombre de concepts pour l'analyse des processus cognitifs distribués :

Horizon d'observation : l'horizon d'observation d'une activité représente la partie de cette activité qui peut être perçue par les autres membres du groupe. L'horizon d'observation est comparable à la partie externalisée d'une activité dans la théorie de l'activité.

Connaissances partagées : les connaissances du groupe ne sont pas centralisées, mais réparties, entre les membres du groupe et aussi dans les outils utilisés par le groupe. Pour comprendre l'activité du groupe, il faut donc d'abord comprendre la répartition des connaissances dans le groupe. Le concept de « connaissances partagées » est une des idées principales de la cognition distribuée.

Trajet de l'information : la répartition des connaissances n'est pas statique, mais dynamique. Puisque les connaissances sont partagées, aucune personne ne possède l'ensemble des connaissances. Au cours des interactions entre membres du groupe, un certain nombre d'informations est donc échangé. Ces échanges ne sont ni centralisés, ni même forcément directs. L'information suit donc un certain trajet, à l'aide des personnes mais aussi des outils. Au cours de ce trajet, l'information peut changer de forme (de représentation). Pour comprendre le processus du groupe, il faut donc aussi comprendre les trajets suivis par les informations.

Ces concepts permettent d'analyser la place d'un système de conscience de groupe dans le processus cognitif qu'est le travail de groupe.

Le système de conscience de groupe est un artefact, dont le but est de permettre à l'utilisateur de construire sa conscience de groupe. Cette construction se fait à l'aide d'informations sur les activités du groupe. Ces informations sont de deux types : les informations concernant les activités propres de l'utilisateur, et les activités du reste du groupe. Ces informations sont donc *partagées* : chacun ne connaît *a priori* que ses propres activités. L'outil de conscience de groupe doit pouvoir collecter les informations sur les activités de *son* utilisateur.

Mais le but de chaque outil de conscience de groupe est de présenter à l'utilisateur les informations concernant les activités de *tous* les membres du groupe. Ces informations doivent donc être échangées. L'outil de conscience de groupe doit pouvoir agir en tant qu'*émetteur* des informations concernant son utilisateur, et en tant que *récepteur* des informations concernant les autres membres du groupe. Le *trajet* des informations doit donc aller d'un outil vers tous les autres.

Enfin, toutes les informations concernant l'activité d'un utilisateur ne sont pas échangées. Ne sont transmises que les informations correspondant à l'*horizon d'observation* de l'activité. L'outil doit donc permettre à l'utilisateur de définir *l'horizon d'observation*

de ses activités propres, c'est-à-dire le niveau d'accès aux informations de ses activités propres.

2.3 Prérequis

En résumé, l'analyse de la place de l'outil de conscience de groupe a permis de spécifier un certain nombre de prérequis. L'outil doit ainsi :

- collecter les informations sur l'activité de l'utilisateur ;
- propager ces informations aux autres outils ;
- recevoir ces informations ;
- présenter ces informations à l'utilisateur ;
- présenter à l'utilisateur des informations non seulement sur les activités des autres membres, mais aussi sur les activités propres de l'utilisateur ;
- présenter les informations suivant deux modes :
  - un mode périphérique, pour présenter les informations globales ;
  - un mode actif, pour présenter, à la demande de l'utilisateur et de manière simple, le détail d'une information spécifique ;
- permettre à l'utilisateur de définir, pour chaque membre du groupe, son niveau d'accès aux différentes informations de son activité ;
- être dynamique et gérer les informations nouvelles ou changeantes.

3 Conclusion

L'analyse cognitive des mécanismes de construction de la conscience de groupe, ainsi que de la place des systèmes de conscience de groupe parmi ces mécanismes, a permis de spécifier un modèle cognitif de la conscience de groupe ainsi qu'un certain nombre de prérequis pour ces systèmes.

Ces résultats fournissent un cadre de réflexion pour l'analyse du problème de la surcharge d'informations et pour la proposition de solution. C'est le sujet du prochain chapitre, dans lequel le modèle cognitif et les prérequis décrits dans ce chapitre servent de base à la création d'une architecture fonctionnelle adressant le problème de la surcharge d'informations et d'un outil de conscience de groupe basé sur cette architecture.

Contextualisation

Cette thèse aborde la problématique de la conscience de groupe, et plus particulièrement le problème de la surcharge d'informations dans les outils de conscience de groupe. En effet, ainsi que l'a décrit Simon [Sim72], les ressources cognitives de l'être humain sont limitées. Lorsqu'il est confronté à des situations complexes où les informations disponibles sont trop nombreuses, ses ressources cognitives limitées ne lui permettent pas de traiter toutes ces informations. Pour un outil de conscience de groupe, il y a ainsi un risque de surcharger l'utilisateur d'informations non pertinentes qui l'empêche de mener à bien son activité.

Dans le chapitre précédent, une analyse des mécanismes de construction de la conscience de groupe ainsi que de la place de l'outil de conscience de groupe a permis de déduire un modèle cognitif ainsi qu'un certain nombre de prérequis pour un tel outil. Le but de ce chapitre est d'utiliser ces résultats pour proposer un outil de conscience de groupe capable de prendre en compte le problème de la surcharge d'information.

1 Proposition

1.1 Présentation générale de l'approche

La problématique de cette thèse concerne la surcharge d'informations. Comme cela a été vu dans le chapitre « Problématique », dans un système de conscience de groupe, la surcharge d'informations peut provenir de trois problèmes :

- un problème de passage à l'échelle, lorsque le nombre de participants ou d'artefacts manipulés augmente ;
- un problème de pertinence des informations, lorsque tous les membres du groupe reçoivent les mêmes informations et les présentent de manière identique ;
- un problème de représentation, lorsque la représentation des informations ne permet pas une compréhension facile et rapide de la part de l'utilisateur.

Il faut remarquer que la surcharge d'informations n'apparaît pas lorsque les membres du groupe travaillent en même temps dans le même lieu, c'est-à-dire lorsque la conscience de groupe se construit naturellement entre les membres du groupe. Le modèle cognitif défini au chapitre précédent (cf. fig. 1) fournit une explication à cela. Il est en effet possible d'identifier dans ce modèle un certain nombre d'étapes qui participent à la régulation du flot d'informations. Ainsi, l'étape d'agrégation réduit le nombre d'informations en agrégeant les informations de bas niveau en informations de haut niveau, moins nombreuses car sémantiquement plus riches. De même, l'étape de filtrage réduit le nombre des in-

formations à interpréter en filtrant les informations non pertinentes. Enfin, l'utilisateur ne perçoit que les informations disponibles dans son contexte (l'endroit où il se trouve, les personnes qu'il rencontre), ce qui limite le nombre d'informations perçues. Ces trois étapes, en limitant le flot d'informations à interpréter aux seules informations pertinentes, permettent de limiter les effets de surcharge d'informations des deux premiers problèmes. En ce qui concerne la représentation, ce n'est pas un problème dans ce cas car les informations ne sont pas *représentées*, elles sont directement perçues dans leur format d'origine.

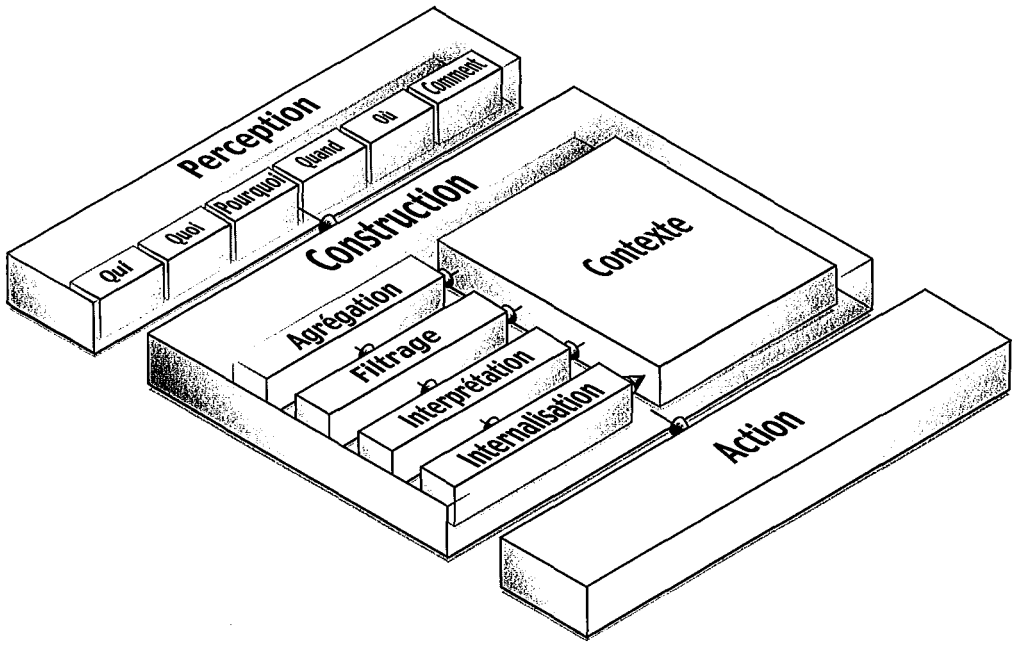

FIG. 1 – Un modèle des étapes cognitives dans la construction de la conscience de groupe.

Toutes ces analyses ont été faites en prenant pour cadre un groupe de personnes travaillant en même temps et dans le même lieu. Le but initial, « primitif », des outils de conscience de groupe est de fournir aux membres d'un groupe distribués dans le temps et dans l'espace non seulement les mêmes informations que s'ils étaient ensemble en même temps et dans le même lieu, mais en plus de les fournir de la même manière. Or, certains types d'informations ne sont pas accessibles à un ordinateur, et certains types de communications ne peuvent pas être simulés par le petit écran de l'ordinateur. L'utilisation de matériel spécifique peut permettre l'accès à certains types d'informations ou à certains types de communication, comme par exemple le regard et le geste, mais cela pose alors le problème de la généricité de l'outil. En effet, si l'outil se base sur la disponibilité de certaines informations, il devient dépendant du matériel spécifique y donnant accès. Si cela ne pose pas de problème pour un prototype de recherche dont le seul but est de valider des idées de recherche et dont le cadre d'utilisation peut ainsi être fixé arbitrairement, cela en pose en revanche pour la réalisation d'un outil qui a pour vocation d'être réellement adopté et utilisé. Comme le problème de la surcharge d'informations est plus un problème lié à un cadre d'utilisation réelle qu'à un cadre de recherche, cette thèse se place dans un

cadre général, sans faire d'hypothèse sur la présence d'un matériel spécifique.

Dans le cadre général, les outils de conscience de groupe ne peuvent donc ni fournir les mêmes informations que celles fournies aux membres d'un groupe travaillant en même temps et dans le même lieu, ni les fournir de la même manière. Plutôt que d'essayer d'imiter fidèlement ce qui arrive dans un groupe partageant le même lieu et le même moment, cette thèse propose de tirer avantage de la technologie pour améliorer le système de façon à compenser ses faiblesses, comme cela est suggéré par Hollan [HS92]. Ainsi, plutôt que d'essayer de fournir les mêmes informations de la même manière que la conscience de groupe « naturelle », l'outil de conscience de groupe doit tirer parti de la technologie pour aider l'utilisateur à construire sa conscience de groupe de la manière la plus efficace possible, et notamment en évitant de le surcharger d'information.

Le modèle cognitif a permis de distinguer trois étapes permettant, dans le cas d'une équipe travaillant en même temps dans le même lieu, de diminuer la surcharge d'information. Ces trois étapes ont un point commun : elles utilisent toutes les trois le contexte de l'utilisateur pour limiter le flux d'informations à interpréter. La première proposition de cette thèse est donc d'adapter la conscience de groupe au contexte de travail de l'utilisateur. Pour cela, cette thèse propose de réaliser certaines étapes cognitives, comme les étapes d'agrégation et de filtrage, directement dans l'outil, en utilisant notamment des techniques de l'intelligence artificielle. L'idée n'est pas de substituer les étapes cognitives de l'utilisateur par celles de l'outil, mais plutôt de faciliter le travail cognitif de l'utilisateur en lui présentant des informations prétraitées. En réalisant ces étapes dans l'outil, la diffusion des informations de conscience de groupe est adaptée à l'activité de l'utilisateur. Le flot d'information est limité aux informations pertinentes, ce qui répond aux deux premiers problèmes de la surcharge d'information. Pour compléter cela, cette thèse propose d'adapter aussi la représentation des informations au contexte de l'utilisateur, afin de répondre au troisième problème. En adaptant les informations fournies à l'utilisateur pour les rendre plus pertinentes, moins intrusives, et présentées d'une manière plus adaptée à son activité, le but est rendre le traitement de ces informations moins gourmand en ressources cognitives, afin de faire diminuer, voire même disparaître, la surcharge cognitive.

La proposition de cette thèse est donc d'utiliser le contexte de travail de l'utilisateur pour adapter la conscience de groupe à l'activité de l'utilisateur. Pour cela, il faut d'abord une définition claire et utilisable du concept de *contexte de travail*.

1.2 Concept de *contexte de travail*

Pour trouver une définition du *contexte de travail*, il faut commencer par s'intéresser au concept de *contexte*. En effet, le concept de *contexte* a déjà fait l'objet de nombreux travaux dans le domaine de recherche de l'*Ubiquitous Computing*[5]. Ainsi, Dey et Abowd [DA99] en donnent la définition suivante :

> « Le contexte est l'ensemble des informations qui peuvent être utilisées pour

[5]Le domaine de l'*ubiquitous computing* a pour objet d'étude une informatique embarquée dans les objets de la vie de tous les jours, d'une manière omniprésente mais invisible, et dont le but est de simplifier la vie de l'utilisateur dans ses activités courantes.

caractériser la situation d'une entité. »

Cette définition est pertinente pour le domaine de l'« Ubiquitous Computing », car son but est de permettre aux outils de la vie de tous les jours de s'adapter à la situation de l'utilisateur. Mais pour la conscience de groupe, ce n'est pas la situation de l'utilisateur qui est pertinente, mais son activité, définie — au sens de la théorie de l'activité — comme le contexte dans lequel s'effectuent les actions de l'utilisateur. L'adaptation au concept de *contexte de travail* de la définition de Dey et Abowd pour la conscience de groupe donne la définition suivante :

> « Le contexte de travail d'une personne est l'ensemble des informations qui peuvent être utilisées pour caractériser son activité. »

Puisque la conscience de groupe se forme à partir d'informations sur les activités d'autrui, le contexte de travail peut être vu comme la source des informations de conscience de groupe, c'est-à-dire des informations que l'outil de conscience de groupe doit capturer et propager à tous les membres du groupe pour que ces derniers puissent construire leur conscience de groupe. Les deux principales fonctionnalités de l'outil de conscience de groupe sont donc la capture des informations issues du contexte de travail de chacun et la diffusion de ces informations aux membres du groupe.

1.3 Architecture fonctionnelle

Cette thèse propose donc d'utiliser le contexte de travail de l'utilisateur pour adapter la conscience de groupe à l'activité de l'utilisateur. La réalisation de cette adaptation doit se faire dans le cadre d'une architecture fonctionnelle, prenant en compte non seulement le modèle cognitif mais aussi les prérequis déduits de l'analyse des mécanismes de conscience de groupe.

L'analyse de la place de l'outil de conscience de groupe a permis de spécifier un certain nombre de prérequis. L'outil doit ainsi :

- collecter les informations sur l'activité de l'utilisateur ;
- propager ces informations aux autres outils ;
- recevoir ces informations ;
- présenter ces informations à l'utilisateur ;
- présenter à l'utilisateur des informations non seulement sur les activités des autres membres, mais aussi sur les activités propres de l'utilisateur ;
- présenter les informations suivant deux modes :
  - un mode périphérique, pour présenter les informations globales ;
  - un mode actif, pour présenter, à la demande de l'utilisateur et de manière simple, le détail d'une information spécifique ;
- permettre à l'utilisateur de définir, pour chaque membre du groupe, son niveau d'accès aux différentes informations de son activité ;
- être dynamique et gérer les informations nouvelles ou changeantes.

Les quatre premiers prérequis (collecter, propager, recevoir et présenter) définissent l'architecture de base de tout outil de conscience de groupe. Cette architecture est divisée en deux parties : la partie émettrice et la partie réceptrice. La partie émettrice d'un outil collecte les informations et les diffusent aux membres du groupe. Ces informations sont

reçues par les parties réceptrices des autres outils, qui présentent ces informations à leur utilisateur. Les informations collectées sont aussi reçues par la partie réceptrice du même outil, de façon à représenter non seulement les informations concernant les autres membres du groupe, mais aussi celles concernant l'utilisateur lui-même, ce qui correspond à l'un des prérequis.

Il est possible d'établir un parallèle entre cette architecture et les trois étapes cognitives de la conscience de groupe : la collecte des informations correspond à l'étape de *Perception*, la diffusion à l'étape de *Construction*, et la représentation à l'étape d'*Action*.

Pour adresser le problème de la surcharge d'information, trois étapes sont ajoutées : une étape d'agrégation, une étape de filtrage et une étape d'adaptation. L'étape d'agrégation, située dans la partie émettrice après l'étape de collecte, a pour but, comme dans la construction de la conscience de groupe, d'agréger les informations de bas niveau en informations de haut niveau. Ces informations de haut niveau forment la représentation du contexte pour l'outil. L'étape de filtrage, située dans la partie émettrice juste après l'étape d'agrégation, a pour but de réaliser une partie du filtrage effectué dans la construction de la conscience de groupe, en ne diffusant au receveur que les informations *a priori* pertinentes pour son activité. Ce filtrage permet aussi à l'utilisateur de contrôler, en fonction du destinataire, le contenu des informations diffusées, ce qui correspond à un autre des prérequis. Enfin, l'étape d'adaptation, située dans la partie réceptrice entre l'étape de réception et l'étape de présentation des informations, permet de réaliser une autre partie du filtrage en indiquant quand et comment les informations doivent être présentées, suivant l'activité courante de l'utilisateur.

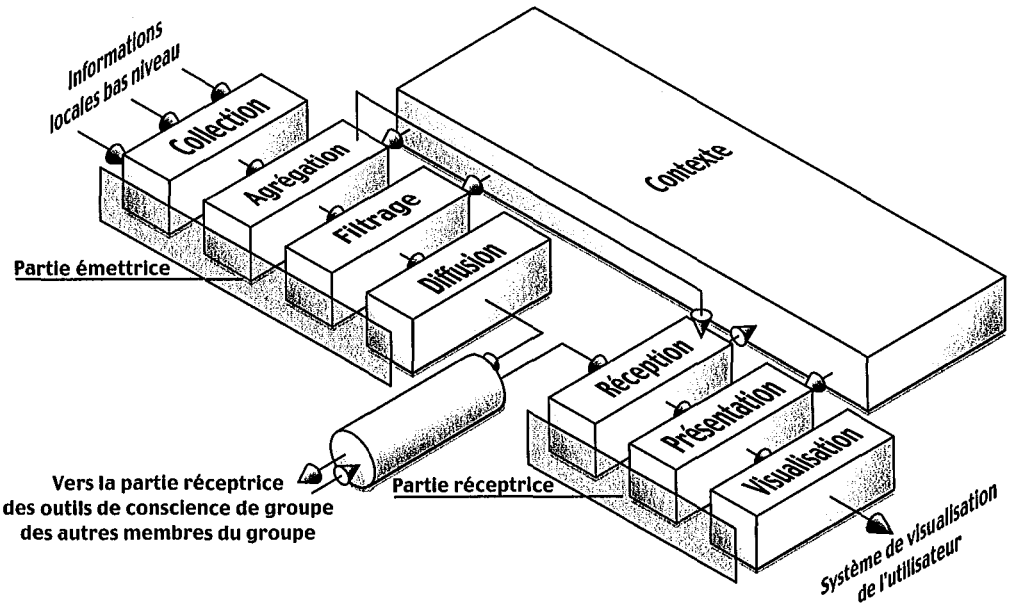

FIG. 2 – L'architecture fonctionnelle de l'outil de conscience de groupe.

Le résultat est l'architecture fonctionnelle suivante (cf. fig. 2) :

- le premier niveau du système collecte les informations de bas niveau sur l'activité

de l'utilisateur ;
- le second niveau agrège les informations de bas niveau en informations de haut niveau ; ces informations haut niveau sont envoyées au niveau suivant pour être filtrées, ainsi qu'au cinquième niveau de l'outil pour être représentées directement à l'utilisateur ;
- le troisième niveau filtre les informations non pertinentes et personnelles en fonction du contexte et des niveaux d'accès définis par l'utilisateur : les informations jugées pertinentes et diffusables sont envoyées au niveau suivant ;
- le quatrième niveau diffuse les informations aux autres outils ;
- le cinquième niveau est chargé de recevoir les informations envoyées par les outils des autres membres du groupe mais aussi par le second niveau : les informations reçues sont sauvées dans le contexte local, puis envoyées au niveau suivant ;
- le sixième niveau décide du moment et de la façon dont l'information va être représentée en fonction du contexte et des préférences de l'utilisateur ;
- le dernier niveau présente l'information à l'utilisateur. Ce niveau est aussi chargé des interactions avec l'utilisateur. La présentation des informations peut se faire suivant au moins deux modes : un mode périphérique et dynamique, présentant les changements et les nouvelles informations, et un mode actif permettant à l'utilisateur de rechercher une information particulière.

Les sept niveaux de l'architecture fonctionnelle sont répartis en deux parties, une partie émettrice et une partie réceptrice. La partie émettrice collecte les informations, les agrège en informations de haut niveau, filtre ces dernières et les diffuse. La partie réceptrice reçoit les informations haut niveau diffusées, les adapte à l'activité de l'utilisateur et les lui présente.

La communication d'informations entre les deux parties se fait de deux façons. Tout d'abord, les informations produites par le niveau deux sont directement envoyées au niveau cinq du même outil. Cette communication permet au système de représenter à l'utilisateur ses propres informations. Ensuite, les informations filtrées par le niveau trois sont transmises aux différents membres du groupe par le niveau quatre, en charge de la diffusion. Ces informations sont ensuite récupérées par le niveau cinq des personnes concernées. Les niveaux quatre et cinq assurent ainsi la communication entre les outils des différents membres du groupe (cf. fig. 3).

Cette architecture fonctionnelle permet à un outil de conscience de groupe non seulement d'utiliser le contexte de travail comme base des informations de la conscience de groupe, mais aussi de s'adapter au contexte de l'utilisateur, afin de réduire le problème de la surcharge d'information.

1.4 Utilisation du contexte

La proposition de cette thèse est donc d'utiliser l'architecture fonctionnelle définie précédemment pour construire un outil de conscience de groupe capable de s'adapter au contexte de travail de l'utilisateur. Dans cette architecture, le contexte se définit comme l'ensemble des informations de haut niveau reçues par l'outil. Ces informations de haut niveau proviennent à la fois des outils des autres membres du groupe et du propre outil de l'utilisateur.

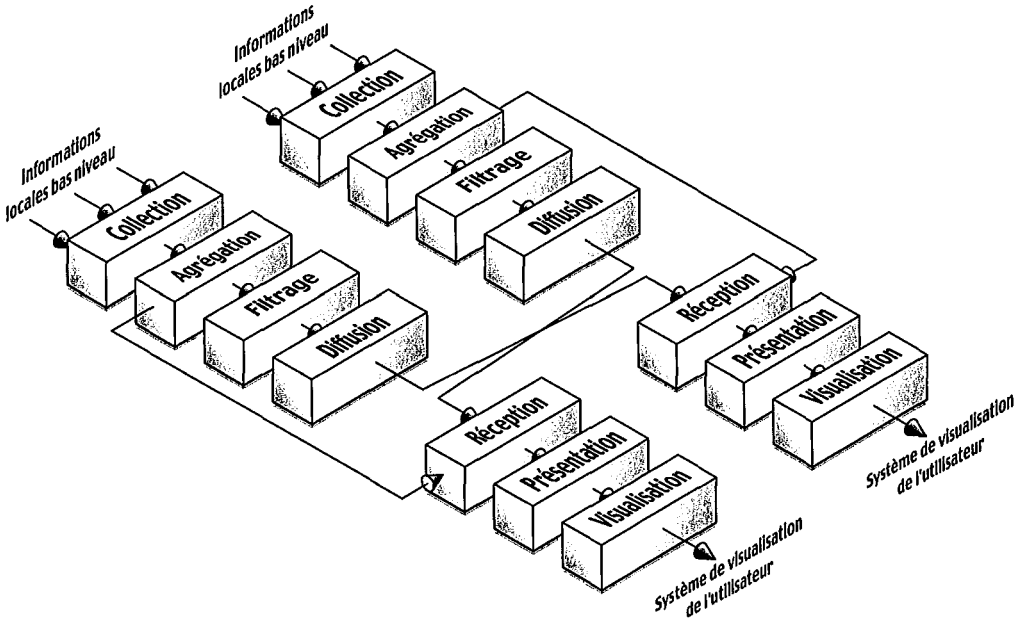

FIG. 3 – Communication entre plusieurs outils de conscience de groupe.

Ce contexte de travail est utilisé à plusieurs niveaux. En premier lieu, le contexte de travail est utilisé au niveau de l'étape d'agrégation. Dans cette étape, le contexte de travail de l'utilisateur est utilisé pour agréger les informations collectées (de bas niveau) en informations sémantiques de haut niveau. L'outil collecte, par exemple par l'intermédiaire du système d'exploitation, un ensemble d'informations sur l'utilisateur et son activité. Ces informations collectées sont des informations de bas niveau : elles décrivent les logiciels utilisés, les documents ouverts, la fréquence de la frappe clavier et des clics souris... L'agrégation a pour but de trouver, à partir d'un ensemble d'informations de bas niveau, l'information de haut niveau correspondante. Cette information de haut niveau décrit l'activité de l'utilisateur : par exemple, la lecture d'un courriel. Puisque les informations de haut niveau obtenues décrivent les activités de l'utilisateur, elles constituent le contexte de travail de l'utilisateur. Ainsi, l'agrégation permet de construire le contexte de travail de l'utilisateur.

Utiliser des informations de haut niveau comme constituant du contexte de travail au lieu des informations de bas niveau a plusieurs avantages. Tout d'abord, ce sont ces informations de haut niveau qui sont transmises au reste du groupe. Étant plus synthétiques et plus pertinentes que les informations de bas niveau, elles permettent de diminuer le nombre d'informations envoyées, tout en gardant (voire en augmentant) la pertinence de ces informations. Ensuite, c'est à partir des informations de haut niveau que l'outil va adapter la conscience de groupe. Cette adaptation est plus facile avec des informations sémantiques de haut niveau qu'avec des informations de bas niveau.

Le contexte de travail est ensuite utilisé au niveau de l'étape de filtrage. Dans cette étape, le contexte de travail de l'utilisateur est utilisé pour déterminer quelles informations doivent être diffusées et à qui. Les informations haut niveau, décrivant l'activité courante

de l'utilisateur, doivent être diffusées aux membres du groupe afin qu'ils puissent construire leur conscience de groupe. Le filtrage a pour but de déterminer, pour chaque destinataire, et à partir du type de l'information et du contexte de travail de l'utilisateur, le niveau de détail de l'information à envoyer.

Le filtrage a deux rôles : diffuser des informations plus pertinentes à chacun, et permettre à l'utilisateur de choisir les informations à diffuser aux autres membres du groupe. En effet, toutes les informations concernant l'activité de l'utilisateur ne sont pas forcément utiles et intéressantes pour tous les membres du groupe. De plus, l'utilisateur peut considérer que certaines parties de l'information sont personnelles. Il n'a pas forcément envie que tout le monde sache ce qu'il est en train de faire, ni avec autant de détails. Il peut accepter de partager certaines informations avec certaines personnes, mais pas avec d'autres. Ce contrôle des informations émises correspond au prérequis nécessaire pour garantir le respect de la vie privée de l'utilisateur, comme cela sera vu en détail dans la partie suivante.

Finalement, le contexte de travail est utilisé au niveau de la présentation des informations. La réception d'une information haut niveau indique un changement dans l'activité d'un membre du groupe. Pour permettre à l'utilisateur de mettre à jour sa conscience de groupe, l'outil notifie l'utilisateur de ce changement en lui présentant l'information reçue. Cette notification doit perturber le moins possible l'activité de l'utilisateur, car la construction de la conscience de groupe ne doit pas se faire au dépend de l'activité principale de l'utilisateur. Le but de l'étape d'adaptation est, comme son nom l'indique, d'adapter la présentation des informations haut niveau reçues au contexte de travail de l'utilisateur. Pour cela, l'outil détermine le niveau d'intrusion avec lequel l'information doit être présentée à l'utilisateur, en fonction du contexte de travail de l'utilisateur et de l'information elle-même.

L'adaptation de la présentation d'une information se fait suivant deux critères : l'importance de l'information pour l'activité de l'utilisateur, et le niveau d'intrusion que l'activité de ce dernier peut supporter. Si l'utilisateur est dans une activité peu interruptible (comme par exemple écrire ou coder), ou bien si l'information est peu pertinente pour l'activité de l'utilisateur, alors la notification est présentée d'une manière peu intrusive. Inversement, si l'information est primordiale pour l'activité de l'utilisateur (par exemple parce qu'elle indique un problème sur le fichier utilisé), alors la notification est présentée de manière à ne pas pouvoir échapper à l'attention de l'utilisateur.

Cette adaptation permet de filtrer les informations reçues en fonction de leur pertinence : si une information n'est pas pertinente, alors son niveau d'intrusion est minimal, et elle n'est pas présentée à l'utilisateur. Cela permet de réduire la surcharge cognitive de l'utilisateur. De plus, les informations présentées étant reliées à l'activité courante de l'utilisateur, ce dernier a plus de facilités à les comprendre et à les intégrer. Finalement, l'adaptation de la présentation des informations à l'activité de l'utilisateur permet de ne pas l'interrompre dans son activité plus que nécessaire.

Il est clair que le contexte de travail d'une personne est complexe et constitué d'un très grand nombre d'informations. Il est donc *a priori* impossible de le modéliser ou de capturer les informations le constituant de manière exhaustive. Cela est encore plus vrai si l'outil se limite dans sa collecte aux seules informations disponibles par le biais de l'ordinateur. Cependant, l'hypothèse de cette thèse est que même en utilisant seulement

une fraction du contexte de travail, des résultats concrets et appréciables sont obtenus en ce qui concerne la diminution de la surcharge cognitive.

Respect de la vie privée

La collecte, l'utilisation et la distribution d'informations sur le contexte de travail d'une personne posent le problème du respect de la vie privée de la personne. En effet, le contexte de travail d'un utilisateur contient des informations que ce dernier peut considérer comme personnelles. *A priori*, l'utilisateur n'a pas envie que tous les membres du groupe aient accès à toutes ses informations. Il n'a pas non plus envie que l'outil soit utilisé pour surveiller ses moindres faits et gestes, à la manière de « Big Brother ». Si l'utilisateur trouve que l'outil est un espion et non pas une aide, alors il ne l'utilisera pas. D'après les travaux de Bellotti [BS93], pour qu'un tel outil soit réellement adopté et utilisé par les utilisateurs, il faut absolument que l'utilisateur ait le sentiment de garder le contrôle de l'outil. En particulier, cela signifie que l'utilisateur doit pouvoir :

- avoir accès aux informations le concernant ;
- modifier ces informations et leurs destinataires ;
- avoir confiance dans l'outil.

Les deux premiers éléments permettent à l'utilisateur de garder le contrôle des informations manipulées par l'outil. Le dernier élément donne à l'utilisateur le sentiment de pouvoir garder le contrôle de l'outil.

Dans le cas de l'outil proposé par cette thèse, différentes stratégies de présentation et de modification des informations sont disponibles. L'accès et la modification des informations se font soit au niveau du filtrage des informations à diffuser (pour les informations à propager aux autres membres du groupe), soit au niveau de la réception et de l'enregistrement des informations (pour les informations concernant l'utilisateur lui-même). Il est intéressant de noter que le contrôle des informations est aussi l'un des prérequis définis dans la partie précédente : il correspond au contrôle de l'*externalisation* de l'activité, définie par la théorie de l'activité, ainsi qu'au contrôle de l'*horizon d'observation*, défini dans la cognition distribuée.

Pour que l'utilisateur ait confiance dans l'outil, celui-ci est proposé sous une licence « open-source » : l'utilisateur a accès au code source de l'outil — décrivant son fonctionnement interne — et peut ainsi vérifier qu'aucune information n'est cachée ou envoyée à son insu. De plus, la sûreté de la diffusion des informations (aux niveaux quatre et cinq) est assurée. La diffusion des messages se fait par un réseau chiffré d'« égal à égal » — sans serveur centralisant les messages — ce qui permet d'assurer à l'utilisateur que seul le destinataire d'un message a accès aux informations contenues dans ce message.

La suite du chapitre revient en détail sur chacun des niveaux de l'architecture de l'outil. Le chapitre se termine par un état de l'art sur l'utilisation du contexte dans la conscience de groupe. Mais auparavant, il est nécessaire d'introduire l'exemple utilisé tout au long de ce chapitre.

2 Exemple d'équipe coopérative

Afin d'illustrer la description des fonctionnalités de l'outil de conscience de groupe et son adaptation au contexte de travail des utilisateurs, il faut un exemple de travail coopératif assisté par ordinateur, et plus particulièrement d'équipe coopérative. L'exemple choisi est celui d'une équipe de développement de jeux vidéo.

La création d'un jeu vidéo est devenue une entreprise complexe, mettant en jeu des moyens conséquents ainsi que de nombreuses compétences. Les équipes de développement sont désormais multidisciplinaires, alliant la programmation, le graphisme 2D et 3D, l'animation, la musique...

La coopération entre toutes ces disciplines est essentielle pour l'aboutissement du projet global. Mais les membres de ces disciplines ont souvent des cultures et des méthodes de travail différentes, même si tous partagent globalement la culture du jeu vidéo. La méthode de travail d'un programmeur enchaînant les lignes de code est assez différente par exemple de celle d'un graphiste cherchant l'inspiration pour une illustration. De plus, il est fréquent que ces personnes soient réparties dans l'espace et le temps, principalement pour des raisons de coût et de compétences. Certaines parties spécifiques du projet peuvent par exemple être sous-traitées par une équipe de spécialistes se trouvant à l'autre bout du monde. Ces différences dans les cultures de travail ajoutées à la répartition dans le temps et dans l'espace des différentes personnes peuvent créer de sérieux problèmes pour la coopération. Les outils de conscience de groupe ont justement pour but de résoudre ces problèmes et de rendre la coopération plus facile et plus fluide. Prendre une équipe de développement de jeux vidéo comme exemple est donc tout à fait pertinent pour illustrer l'utilisation d'un outil de conscience de groupe s'adaptant à l'activité de l'utilisateur.

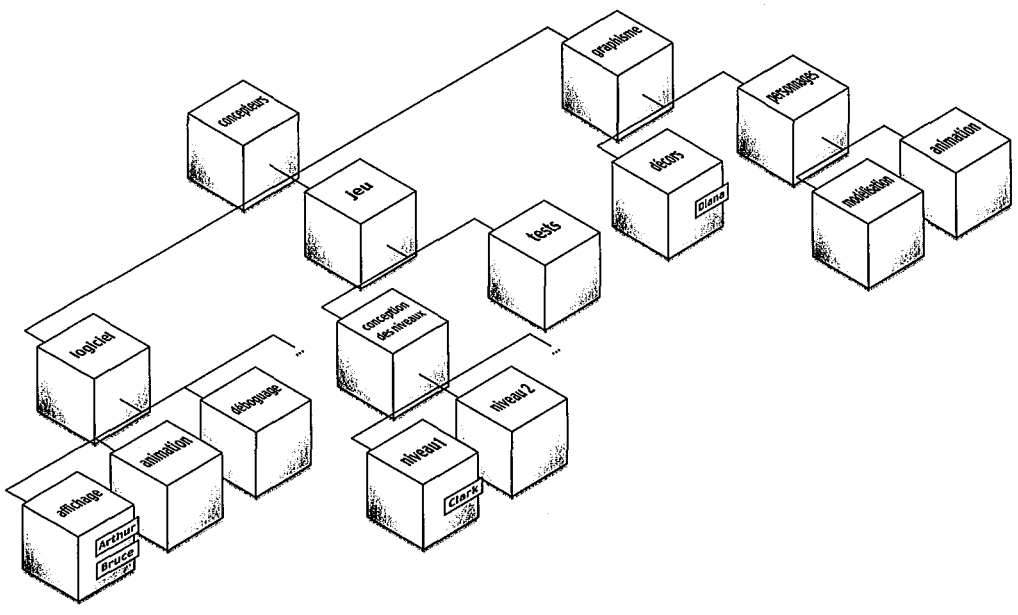

FIG. 4 – Une partie de la structure hiérarchique de l'équipe de développement de jeux vidéo et certains de ses membres.

L'équipe fictive de développement de jeux vidéo utilisée comme cadre des exemples de ce chapitre possède la structure suivante[6] (cf. fig. 4). Tout en haut de la hiérarchie se trouvent les concepteurs du jeu. En charge du concept du jeu, ils ont pour tâche de diriger le développement global du jeu, de décider des changements majeurs si besoin est, et d'assurer l'intégration et la cohérence du produit final. Ce sont les chefs du projet global. Le reste de l'équipe de développement est décomposé en cinq départements : le département « logiciel », le département « jeu », le département « graphisme », le département « musique », et enfin le département « cinématique ». Les départements plus administratifs, comme le département « financier », ne sont pas décrits dans l'exemple car ils ne participent pas directement au processus coopératif de développement.

Le département « logiciel » a pour mission de développer et d'améliorer le moteur de jeu (*game engine*) du projet. Un moteur de jeu est un ensemble de ressources logicielles en charge de la gestion, de l'affichage et de l'animation des objets du jeu vidéo. Il est aussi responsable de la musique, de la gestion des fichiers, et de l'interaction avec l'utilisateur. Le moteur de jeu est une partie importante du projet, car c'est lui qui détermine les limites techniques du jeu. Le département logiciel est divisé en sous-départements, chacun responsable d'une des fonctionnalités du moteur de jeu. Il y a ainsi un sous-département « affichage », un sous-département « animation »... Le sous-département « déboguage » est dédié uniquement à la correction des bogues.

Le département « jeu » a pour but de développer le jeu proprement dit, ainsi que d'assurer la jouabilité (*game play*) du produit. Il décide ainsi de l'enchaînement des étapes, des interactions entre les éléments, etc. Il est divisé en deux sous-départements, le département « conception des niveaux » et le département « tests ». Le sous-département « conception des niveaux » est responsable du développement des différents niveaux du jeu. Chaque niveau est géré par une sous-équipe spécifique, qui définit l'architecture du niveau, son but ainsi que les créatures et objets se trouvant à l'intérieur. Le département « tests » a pour mission de tester les différents niveaux, au fur et à mesure de leur développement. Le but est d'évaluer et d'affiner la jouabilité du niveau, ainsi que de traquer les bogues (défauts d'affichage, mauvais comportement des objets, etc.).

Le département « graphisme » est en charge de l'apparence visuelle du jeu. Il donne corps aux niveaux, objets et personnages conçus par le département « jeu ». Il est divisé en deux sous-départements, l'un pour les décors, l'autre pour les personnages. Le sous-département « personnages » est lui-même divisé en deux sous-équipes. L'une des équipes s'occupe de la modélisation des personnages, c'est-à-dire de leur création, de leur modélisation en 3D, et de leur habillage par des textures 2D. L'autre équipe s'occupe de l'animation des personnages. Chaque action et réaction d'un personnage est une animation, dont l'enchaînement sous les ordres de l'utilisateur ou de la machine définit le comportement du personnage.

Toutes ces personnes ne travaillent pas forcément en même temps et dans le même lieu. Cependant, elles travaillent toutes sur ordinateur, car leurs outils, que ce soit pour la programmation, le graphisme 2D ou 3D, l'édition musicale ou l'édition vidéo, sont tous informatiques. De plus, pour les besoins de l'exemple, on suppose que toutes ces personnes

[6] La structure de cette équipe fictive est inspirée de celles de réelles équipes de développement de jeux vidéo, telles que décrites dans la presse et les émissions spécialisées.

sont reliées par un réseau (intranet ou internet) qui leur permet de communiquer.

Cette équipe fictive de développement de jeux vidéo sert de cadre pour tous les exemples développés dans la suite du chapitre.

3 Application de l'architecture fonctionnelle

Cette partie revient en détail sur chacune des étapes de l'architecture fonctionnelle, ainsi que sur les liaisons entre ces différentes étapes.

3.1 Collecte des informations

La première étape dans le fonctionnement de l'outil de conscience de groupe est la collecte des informations bas niveau provenant du contexte de travail de l'utilisateur (cf. fig. 5). L'outil doit capturer le plus d'informations possible pouvant donner des indications sur l'activité de l'utilisateur. Ces informations sont représentées sous la forme d'un ensemble de trois champs :

- un champ « type », qui spécifie le type de l'information : un lancement ou une fermeture d'application, une ouverture ou une fermeture de document, la fréquence des clics souris pendant la dernière minute... ;
- un champ « valeur », qui spécifie la valeur de l'information : le nom de l'application ou du document concerné, la valeur de la fréquence des clics souris... ;
- un champ « date », qui indique le moment ou l'information a été reçue.

Par exemple, le lancement de l'application « Acrobat Reader » à une date donnée est représenté par l'information suivante :

type : Lancement d'une application
valeur : Acrobat Reader
date : Vendredi 29 août 2003, 15h30 GMT+2

C'est sous cette forme que l'information est collectée puis utilisée dans le reste de l'outil.

La collecte d'informations peut se faire en général de deux manières différentes. Certaines informations peuvent être capturées automatiquement par l'outil, sans l'aide de l'utilisateur. C'est le cas par exemple de la fréquence des frappes claviers ou des clics souris. Une autre manière de collecter une information est de la demander directement à l'utilisateur. L'outil peut, par exemple, demander à l'utilisateur la position de chaque personne dans la structure hiérarchique du groupe. Mais plus le nombre et la fréquence des demandes explicites à l'utilisateur augmentent, plus elles interrompent l'utilisateur et lui ajoutent une surcharge de travail. Un bon exemple de ce problème est fourni par le logiciel de messagerie instantanée et de téléprésence *ICQ*. Dans ce logiciel, l'état de l'utilisateur (« occupé », « ne pas déranger » ...) doit

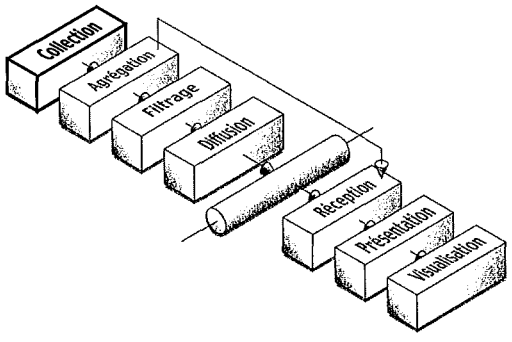

FIG. 5 – Place de l'étape de collecte dans l'architecture fonctionnelle.

être indiqué explicitement par ce dernier. Mais devoir indiquer au logiciel son état à chaque changement d'activité devient rapidement une surcharge de travail insupportable, et l'utilisateur abandonne vite. L'indication d'état perd alors tout son sens. Il est donc important de faire en sorte que le maximum d'informations puisse être collecté de manière automatique, de façon que le système ne donne pas une surcharge de travail à l'utilisateur.

La manière dont l'information est collectée dépend non seulement de l'information elle-même, mais aussi de la fréquence avec laquelle l'information est collectée, c'est-à-dire de la fréquence avec laquelle l'information peut changer. Certaines informations sont susceptibles de changer à tout moment, et doivent donc être capturées fréquemment. C'est par exemple le cas de la fréquence des clics souris. Pour ces informations fortement dynamiques, la capture doit impérativement se faire de manière automatique, sous peine de surcharger l'utilisateur. D'autres informations ne changent que très rarement, voire pas du tout, pendant l'exécution de l'outil. Dans ce cas, ces informations faiblement dynamiques sont collectées dès le premier lancement de l'outil, puis ne sont changées qu'à la demande de l'utilisateur. C'est par exemple le cas de la composition du groupe. Ces informations peuvent être demandées à l'utilisateur, car la demande ne se fait qu'une seule fois, au premier lancement. Il est aussi possible de fournir l'information directement pendant la phase de configuration.

Certaines informations sont disponibles directement par le biais de l'ordinateur de l'utilisateur. C'est par exemple le cas de sa frappe clavier, ou de son adresse IP. Sauf cas particulier, ces informations sont toujours disponibles, quel que soit l'équipement utilisé. D'autres informations en revanche ne sont disponibles qu'avec l'ajout d'un matériel spécifique. C'est par exemple le cas du regard ou du geste. L'outil peut aussi utiliser une infrastructure déjà présente, comme un système de *workflow* ou une infrastructure de gestion de connaissance. Cette infrastructure peut fournir des informations telles que la composition du groupe, le rôle de l'utilisateur, sa tâche en cours ou encore la liste de ses tâches à faire (particulièrement pour un workflow). L'ajout d'un matériel spécifique et la présence d'une infrastructure peuvent aider à collecter des informations inaccessibles autrement. Cependant, leur présence est un cas particulier. Dans le cas général, une équipe distribuée travaille sur des équipements standards. Cette thèse prend donc le parti de proposer un outil générique, utilisable dans n'importe quelle situation. En conséquence, afin de ne pas limiter la validité de l'outil, aucune hypothèse n'est faite sur la présence d'un matériel spécifique ou d'une infrastructure, et seules les informations capturables par un ordinateur « standard » sont considérées. La seule hypothèse demandée est que les membres de l'équipe soient tous connectés à un réseau leur permettant de communiquer. Si une infrastructure ou un matériel spécifique sont présents, l'outil peut les utiliser. Les informations ne changeant que rarement peuvent être soit demandées à l'utilisateur, soit fournies directement pendant la phase de configuration.

Ainsi, les informations bas niveau fortement dynamiques sont toutes fournies par le système d'exploitation, grâce à ses interfaces de programmation système. Ces informations sont collectées périodiquement et automatiquement. La liste de ces informations bas niveau automatiques et fortement dynamiques est décrite dans le tableau suivant.

Informations automatiques fortement dynamiques	
type	*valeur*
liste des applications en cours	liste des noms des applications en cours
documents en cours	couple (application, liste des documents)
lancement d'une application	nom de l'application
fermeture d'une application	nom de l'application
application courante	nom de l'application
ouverture d'un document	URL du document
sauvegarde d'un document	URL du document
fermeture d'un document	URL du document
document courant	URL du document
fréquence des clics souris	valeur pendant la dernière minute
fréquence des frappes clavier	valeur pendant la dernière minute
frappe clavier	valeur de la touche frappée
courriel reçu	nom de l'expéditeur et titre du message

Le système d'exploitation fournit aussi une partie des informations faiblement dynamiques. Ces informations sont collectées automatiquement mais une seule fois, au premier lancement de l'outil. Elles sont alors mises en cache, et comparées aux valeurs courantes à chaque nouveau lancement de l'outil. En cas de nouvelle valeur, la mise à jour est automatique. L'utilisateur peut aussi demander une vérification et une mise à jour pendant l'exécution de l'outil. La liste des informations bas niveaux automatiques et faiblement dynamiques est décrite dans le tableau suivant.

Informations automatiques faiblement dynamiques	
type	*valeur*
type de l'ordinateur	« bureau » ou « portable » ou « PDA »
type de connexion réseau	« ethernet » ou « modem »
adresse IP	adresse IP
nom de l'utilisateur	nom de l'utilisateur

Les autres informations bas niveau faiblement dynamiques que le système d'exploitation ne peut pas fournir se divisent en deux catégories : celles identiques pour tout le groupe, et celles spécifiques à l'utilisateur. Les informations faiblement dynamiques et identiques pour tout le groupe sont fournies directement pendant la phase de configuration de l'outil. C'est le cas de la structure hiérarchique du groupe, ainsi que de la liste de l'ensemble des membres du groupe. Bien sûr, l'utilisateur peut modifier ou compléter ces informations lorsqu'il le souhaite.

Informations fournies dans la phase de configuration	
type	*valeur*
membres du groupe	liste de l'ensemble des membres du groupe
structure hiérarchique du groupe	arbre de la structure du groupe

Les autres informations sont demandées directement à l'utilisateur, lors du premier lancement de l'outil. Ainsi, l'outil ne demande à l'utilisateur que des informations le concernant personnellement, et il ne fait cette demande qu'une seule fois, lors de son premier lancement. Par défaut, le système demande à l'utilisateur s'il veut désigner certains membres du groupe comme proches, de façon à ce que ces personnes reçoivent des informations plus détaillées de l'utilisateur. Bien sûr, l'utilisateur peut modifier ou compléter cette information lorsqu'il le souhaite.

Information demandée à l'utilisateur	
type	*valeur*
personnes proches	liste des noms des personnes proches

Il est possible lors de la phase de configuration d'ajouter, supprimer ou modifier le type, les valeurs possibles et la source de toutes les informations à collecter. Il est par exemple possible d'ajouter des informations ayant pour source une infrastructure existante ou d'un matériel spécifique ajouté. Sans modification, l'outil collecte les informations spécifiées ci-dessus.

Exemple

Dans l'équipe fictive de développement de jeux vidéo par exemple, la phase de configuration n'apporte aucune modification aux réglages par défaut de l'outil en ce qui concerne les informations collectées. L'outil est diffusé et installé sur les postes des utilisateurs préconfiguré avec la structure de l'équipe en départements et sous-départements, et les noms des différents membres de l'équipe. Lors de son premier lancement, le logiciel demande à l'utilisateur de préciser quelles sont les personnes proches. Pour Arthur par exemple, qui est un programmeur du sous-département « affichage » du département « logiciel », la liste des personnes proches contient entre autre :

- Bruce, un autre programmeur du département « affichage » ;
- Diana, une graphiste du sous-département « décors » du département « graphisme ».

Arthur connaît aussi Clark, le responsable de l'un des niveaux développé dans le sous-département « conception des niveaux » du département « jeu ».

Information données par Arthur	
type	*valeur*
personnes proches	(Bruce, Diana)

Arthur travaille au bureau sur un ordinateur PC standard, connecté par une prise ethernet au réseau de l'entreprise. Diana travaille aussi au bureau, mais sur un Macintosh, connecté aussi par ethernet au réseau de l'entreprise. Bruce, lui, travaille chez lui, sur un ordinateur portable, car il a des problèmes de dos en ce moment. Il est connecté au réseau de l'entreprise par modem. Clark est en voyage d'affaire, mais il a un *PDA* (*Portable Digital Assistant*)) qu'il peut connecter au réseau de l'entreprise avec son téléphone portable. Toutes ces informations sont collectées par l'outil lors de son lancement : type de l'équipement utilisé, type de la connexion réseau, adresse IP et nom de l'utilisateur.

Le travail d'Arthur consiste à ajouter de nouvelles fonctionnalités au moteur d'affichage du jeu. Pour le système de conscience de groupe d'Arthur, les applications ouvertes correspondent à ses applications de développement : éditeur, compilateur, débogueur. Il y a aussi un butineur web, affichant de la documentation technique utilisée par Arthur, ainsi que son application de gestion du courriel. L'application en cours varie principalement entre l'éditeur, le compilateur, le débogueur et le butineur web. Les documents ouverts sont, pour les outils de développements, les fichiers sources et exécutables ; pour le butineur web, la documentation technique ; et pour le gestionnaire de courriel, la boîte aux lettres d'Arthur. La fréquence des clics souris est toujours assez faible. Par contre, la fréquence des frappes claviers varie : elle est faible lorsqu'Arthur lit la documentation technique ou bien compile ; elle est forte lorsqu'Arthur code ou débogue.

Alors qu'Arthur est en train de coder, il reçoit un courriel de Clark. Dans ce courriel, Clark décrit une fonctionnalité intéressante qu'il vient de voir dans un jeu concurrent. Il aimerait qu'Arthur essaye de développer la même fonctionnalité, car cela rendrait le niveau dont Clark est responsable bien plus intéressant. Arthur lui répond que la fonctionnalité est intéressante et qu'il s'y met tout de suite. Il décrit la situation à Bruce pour que celui-ci lui donne un coup de main, puis se met au travail.

Le courriel reçu par Arthur donne naissance dans le système de conscience de groupe à une information bas niveau correspondante. Lorsque Arthur lit le mail, le gestionnaire de courriel devient l'application courante, et la frappe clavier devient faible. Lorsqu'Arthur répond à Clark, la frappe clavier devient forte. Finalement, lorsque Arthur envoie un courriel à Bruce, la frappe clavier est forte. Dans toutes ces activités, l'ensemble de ce qui est écrit au clavier par Arthur est collecté par le système.

Lorsque le système a collecté une information bas niveau, il la passe à l'étape suivante, l'étape d'agrégation.

3.2 Agrégation des informations

Le but de cette étape est d'agréger, en fonction du contexte, les informations bas niveau reçues de l'étape précédente, afin d'en déduire des informations haut niveau sur les activités de l'utilisateur (cf. fig. 6). Ces informations haut niveau servent à la représentation du contexte de travail de l'utilisateur, et sont propagées aux autres membres du groupe pour leur permettre de construire leur conscience de groupe.

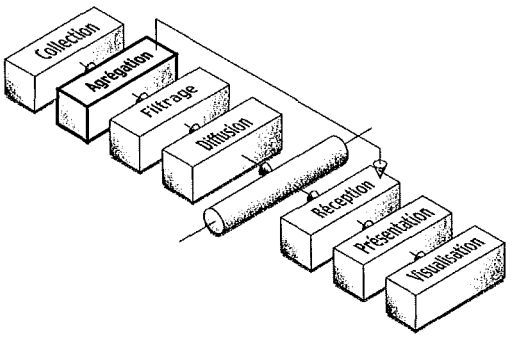

FIG. 6 – Place de l'étape d'agrégation dans l'architecture fonctionnelle.

Chaque information bas niveau reçue est un indice de l'activité de l'utilisateur. Chaque information haut niveau produite contient une partie de la description de l'activité courante de l'utilisateur. L'ensemble de ces informations, reçues et produites, représente une partie du contexte de travail de l'utilisateur. C'est par rapport à ce contexte de travail que les informations bas niveau arrivantes sont agrégées en information haut niveau. Ainsi, le contexte de travail participe à sa propre construction :

l'agrégation utilise le contexte de travail de l'utilisateur pour déduire les informations haut niveau constituant ce contexte.

Les informations haut niveau déduites de l'agrégation des informations bas niveau décrivent le contexte de travail de l'utilisateur, c'est-à-dire son activité courante. Pour simplifier la description du contexte, ces informations haut niveau sont classées en six catégories, répondant à six questions [7] :

quoi : quelle est le type de l'activité effectuée ?

qui : qui réalise cette activité, et en relation ou coopération avec qui ?

quand : quand cette activité a-t-elle commencé ?

où : où et sur quel ordinateur l'activité est-elle effectuée ?

comment : quels sont les objets et outils utilisés pour réaliser l'activité ?

pourquoi : dans quel cadre l'activité est-elle effectuée ?

L'agrégation des informations bas niveau est divisée en six canaux d'agrégations, un par type d'information. Chaque canal a la responsabilité d'un type spécifique d'information haut niveau. Le rôle d'un canal d'agrégation est de collecter les informations bas niveau reçues de l'étape précédente, de les agréger avec les informations précédemment reçues et avec le contexte de travail de l'utilisateur, et d'en déduire si possible une information haut niveau du type correspondant. Ainsi, lorsqu'une information bas niveau est reçue de l'étape précédente, elle est transmise aux six canaux d'agrégations, et chaque canal se charge de la partie correspondante de la description de l'activité de l'utilisateur.

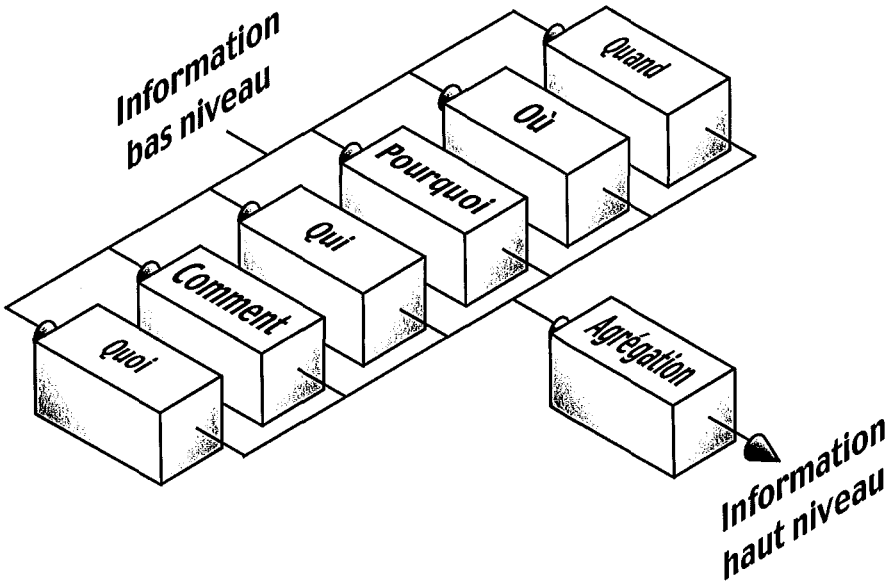

FIG. 7 – Diffusion de l'information bas niveau dans les six canaux d'agrégation.

[7] Cette structuration en six questions est classique, autant dans le domaine de la conscience de groupe [MB97] que dans la modélisation du contexte, que ce soit en intelligence artificielle [BP99, FF00] ou dans le domaine de l'« ubiquitous computing » [SDA99, Pas98].

Les différentes informations fournies par les canaux d'agrégation sont centralisées dans le canal général d'agrégation (cf. fig. 7). Le rôle de ce dernier canal est d'agréger l'ensemble des informations fournies par les différents canaux dans une seule information haut niveau contenant la description complète de l'activité courante de l'utilisateur. L'information agrégée résultante contient alors les six informations produites par les six canaux, classées suivant leurs catégories respectives.

Il n'y a pas forcément création d'une information haut niveau dans chacune des six catégories pour chaque réception d'une information bas niveau. Il est possible par exemple que l'information bas niveau n'apporte pas plus d'information que ce qui était déjà connu. C'est par exemple le cas si la fréquence des clics souris lors de la dernière minute est approximativement la même que lors de la minute précédente. Il est aussi possible qu'une information bas niveau ne contribue à la création que d'une information d'un type particulier. La création d'informations haut niveau dépend des informations bas niveau reçues et du contexte de travail courant de l'utilisateur.

Plusieurs techniques sont envisageables pour réaliser l'agrégation. La plus commune consiste à utiliser des règles, comme par exemple les règles *ECA* [Pat99, WC95] (*event-condition-action*), reliant l'apparition d'un ensemble d'informations bas niveau à la création d'une information haut niveau. Cependant, l'utilisation de règles pose plusieurs problèmes :

- la création, le codage et la relecture des règles n'est pas simple, et n'est pas à la portée de tous ;
- la gestion des règles, et principalement des conflits entre les règles, n'est pas une chose aisée ;
- les règles doivent être entièrement connues avant d'être codées.

Pour éviter ces problèmes, une autre technique est possible : la technique des réseaux bayesiens. Un *réseau bayesien* [Cha91, Nil98, Pea95] est une technique d'intelligence artificielle permettant de déduire la cause la plus probable d'un ensemble d'effets observés. Le réseau bayesien modélise la dépendance entre une cause et un effet par une relation affectée d'une probabilité conditionnelle. Cette probabilité conditionnelle correspond à la probabilité de l'apparition de l'effet sachant la cause. À partir d'une liste d'effets observés, un moteur d'inférence remonte l'arbre des probabilités et trouve la cause la plus probable. Cette technique a déjà prouvé son efficacité dans un grand nombre de cas [HBH+98].

Un réseau bayesien est spécifié par le graphe des relations entre causes et effets et par les probabilités conditionnelles attachées à ces relations. Comparé à des règles, la spécification du graphe des relations de causes à effets est simple et aisément compréhensible de tous. De plus, les règles à respecter pour avoir un réseau bayesien correct sont simples et peu nombreuses : le graphe doit être orienté et acyclique, et la somme des probabilités conditionnelles à un nœud doit être égale à 1. Finalement, certaines techniques permettent au réseau bayesien d'apprendre lui-même les probabilités conditionnelles au fur et à mesure de son utilisation [CG01, CH92, Nil98], ce qui évite d'avoir à les spécifier correctement *a priori*.

Certains canaux de l'outil de conscience de groupe utilisent un réseau bayesien pour réaliser leur agrégation. Dans ce cas, les effets observés proviennent des informations bas niveau reçues, et les causes sont les informations haut niveau que l'on cherche à trouver. Le fait d'écrire un courriel par exemple a entre autre pour effet une fréquence élevée de frappe

clavier. Le type de l'activité de l'utilisateur (« écrire un courriel ») est une information haut niveau qui peut être une cause probable de l'information bas niveau « fréquence élevée de frappe clavier ». Bien sûr, ce n'est pas la seule cause possible. Cependant, si en plus le réseau bayesien reçoit l'information bas niveau « l'application courante de l'utilisateur est un logiciel de courriel », alors cette information haut niveau (« l'utilisateur écrit un courriel ») peut devenir la cause la plus probable (cf. fig. 8). Le réseau bayesien peut aussi utiliser comme effet observé une information haut niveau déjà déduite, une information haut niveau provenant d'un autre canal, voire une information haut niveau reçue d'un autre membre et portant sur son activité.

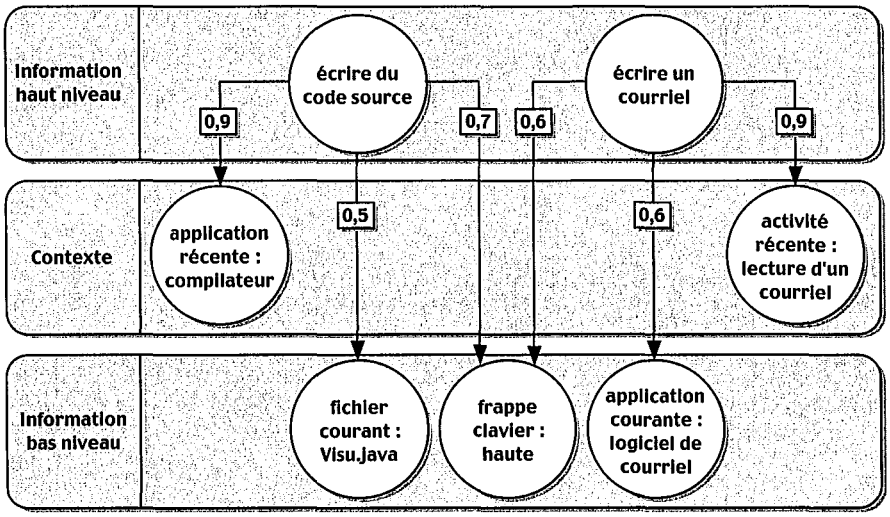

FIG. 8 – Un exemple d'utilisation d'un réseau bayesien.

À chaque fois qu'une information est reçue, le réseau bayesien la traite et met à jour les probabilités correspondantes à son apparition. Par exemple, lorsque l'information bas niveau concernant la frappe clavier est reçue, elle est traitée par le réseau bayesien, qui la compare à différents seuils pour décider s'il s'agit d'une fréquence de frappe élevée, moyenne, ou basse. S'il s'agit par exemple d'une fréquence élevée, le réseau bayesien passe à 1 la probabilité d'apparition d'une fréquence de frappe clavier élevée, et à 0 celles des fréquences moyenne et basse.

À chaque mise à jour des probabilités, le réseau bayesien remonte l'arbre des probabilités afin de trouver la cause la plus probable de l'observation de ces informations bas niveau. Cette recherche peut être infructueuse, par exemple parce qu'aucune cause n'est significativement la plus probable, ou bien parce que plusieurs causes sont majoritairement probables, avec égalité des probabilités. Dans ce cas, le canal fournit en sortie une information particulière, l'information « inconnu », signifiant qu'aucune information haut niveau ne peut être déduite des informations bas niveau observées. À l'inverse, en cas de réussite, le résultat de la recherche est une information haut niveau, du type du canal considéré, et qui est la cause la plus probable des informations bas niveau observées. Le canal fournit alors en sortie cette information haut niveau. Ainsi, pour chaque information

de bas niveau reçue, le canal fournit en sortie soit une vraie information haut niveau, soit l'information « inconnu ».

Lorsqu'une information bas niveau est reçue, elle est passée à chacun des canaux d'agrégation, dans l'ordre suivant : « quoi », « comment », « qui », « pourquoi », « où » et « quand » (cf. fig. 7). L'ordre est important car certains canaux utilisent les informations haut niveau produites par les canaux précédents.

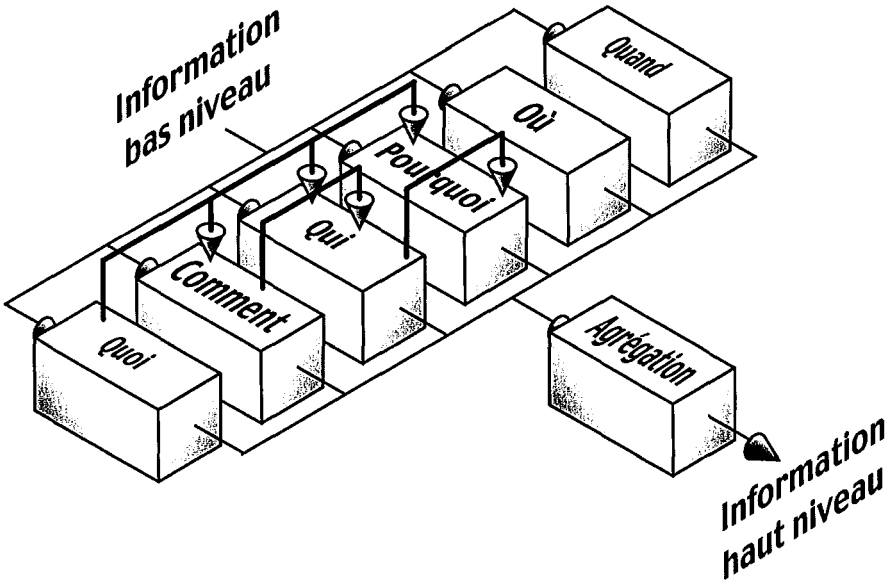

FIG. 9 – Diffusion de l'information bas niveau dans les six canaux d'agrégation.

Le premier canal d'agrégation à recevoir l'information bas niveau est le canal « quoi ». Ce canal utilise un réseau bayesien pour trouver le type de l'activité courante de l'utilisateur. Il est suivi par le canal « comment ». Ce canal utilise un réseau bayesien et les résultats du canal « quoi » pour trouver les outils et les objets utilisés par l'utilisateur pour effectuer son activité. Le canal suivant est le canal « qui », qui produit deux types d'informations : le nom de la personne réalisant l'activité, et le nom des personnes collaborant à cette activité. La première information est simple à obtenir car c'est le nom de l'utilisateur. Pour les noms des personnes coopérant, le canal utilise un réseau bayesien basé notamment sur les résultats des canaux « quoi » et « comment » ainsi que sur l'échange des courriels. C'est ensuite au tour du canal « pourquoi » de recevoir l'information bas niveau. Ce canal utilise un réseau bayesien pour trouver le cadre général dans lequel l'utilisateur effectue son activité, cadre qui peut être de deux types. L'utilisateur peut agir soit dans le cadre d'une demande d'une autre personne, soit dans le cadre de son activité générale. Par exemple, le fait de déboguer un programme peut être soit dû à une demande d'un autre membre du groupe, soit faire partie de l'activité générale d'ajouter une fonctionnalité si l'utilisateur est programmeur, soit faire partie de l'activité générale de corrections des bogues si l'utilisateur fait partie du département « déboguage ». Le réseau bayesien de ce canal observe particulièrement le type de l'activité de l'utilisateur (résultat du canal « quoi ») ainsi que ses échanges de courriels pour parvenir à son résultat.

Les deux derniers canaux n'utilisent pas de réseaux bayesiens. Le canal « où » donne deux types d'informations : le type de l'équipement utilisé (ordinateur de bureau, ordinateur portable ou *PDA* (*Portable Digital Assistant*)), et le type d'endroit où se trouve l'utilisateur (bureau, maison, ou en voyage). Pour trouver ces informations, le canal utilise des règles simples basées directement sur les informations bas niveau envoyées par le système d'exploitation sur le type de l'équipement (type de l'ordinateur, type de connexion réseau, et adresse IP). Finalement, le canal « quand » ajoute des informations temporelles aux informations haut niveau produites par les autres canaux. Le type des informations temporelles ajoutées dépend de l'information haut niveau. Par exemple, le canal attribue à une information haut niveau sur le début d'une activité la date de début de cette activité. À une information de fin d'activité, le canal attribue non seulement la date de fin de cette activité mais aussi la durée de l'activité. Le canal utilise pour cela des règles simples, en fonction des informations de haut niveau reçues et de la date système.

Les différentes informations fournies par les canaux d'agrégation sont centralisées dans le canal général d'agrégation (cf. fig. 9). Le rôle de ce dernier canal est d'agréger l'ensemble des informations fournies par les différents canaux dans une seule information haut niveau contenant la description complète de l'activité courante de l'utilisateur. Cette information n'est produite que si au moins une des informations fournie a changé. Ainsi, la production d'une information agrégée indique un changement dans l'activité de l'utilisateur. L'information agrégée résultante contient alors les six informations produites par les six canaux, classées suivant leurs catégories respectives. Certaines de ces informations peuvent être l'information « inconnu », si le réseau bayesien du canal correspondant n'a pas pu trouver la cause la plus probable. Lorsqu'une information agrégée est produite, elle est passée à l'étape suivante pour être diffusée aux membres du groupe.

La spécification, pour chaque réseau bayesien, des différentes informations haut niveau générables et du graphe de probabilité entre ces informations haut niveau et les informations bas niveau pouvant être collectées, ainsi que des règles pour les canaux « où » et « quand », est faite pendant la phase de configuration.

Exemple

Dans l'exemple de l'équipe fictive de développement de jeux vidéo, les informations haut niveau, les réseaux bayesiens et les règles des différents canaux d'agrégation ont été modélisés spécifiquement pour chaque type de profession. Ainsi, tous les programmeurs du département « logiciel » ont la même configuration d'outils. Par contre, cette configuration est différente de celle des graphistes du département « graphisme ». Dans ce département, les membres du sous-département « décors », du sous-département « modélisation des personnages » et du sous-département « animation des personnages » ont tous des configurations différentes, dues à l'utilisation de logiciels spécifiques dans chaque sous-département.

Lorsqu'Arthur travaille, les informations bas niveau collectées par le système sont principalement les suivantes :

- changement de l'application courante, avec comme application l'éditeur, le compilateur, le débogueur ou le butineur web ;
- ouverture, sauvegarde ou fermeture d'un document, avec comme document un fichier

source, un fichier exécutable ou de la documentation technique ;
- changement de document actif ;
- changement dans la fréquence de la frappe clavier ;

Dès qu'une de ces informations bas niveau est reçue, elle est transmise aux six canaux d'agrégations, dans l'ordre précédemment décrit :« quoi », « comment », « qui », « pourquoi », « où » et « quand » (cf. fig. 7).

Le premier canal à recevoir l'information est le canal « quoi ». Le réseau bayesien se base sur l'information actuelle et sur les informations précédentes pour trouver le type de l'activité de l'utilisateur. Par exemple, en recevant une information sur une haute fréquence de frappe clavier, et en ayant déjà comme information que l'application courante est l'éditeur et que le document courant est un fichier « Monfichier.c », le réseau bayesien en déduit que Arthur est en train d'écrire du code (cf. fig. 10). Avec les informations correspondantes, il peut déduire de la même manière qu'Arthur est en train de compiler, de déboguer ou de lire de la documentation technique.

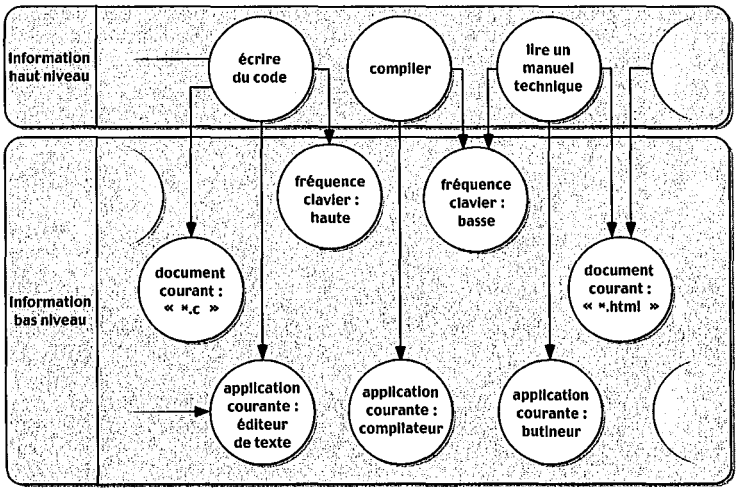

FIG. 10 – Une partie du réseau bayesien du canal « quoi » d'Arthur.

C'est ensuite au tour du canal « comment ». Celui-ci reçoit non seulement les mêmes informations bas niveau que le canal « quoi » mais aussi l'information haut niveau produite par ce dernier. Son réseau bayesien utilise ces informations pour trouver les applications et documents utilisés pour réaliser l'activité de l'utilisateur. Par exemple, en recevant du canal « quoi » l'information qu'Arthur est en train d'écrire du code, le réseau va déduire que l'éditeur et que le document courant « MonFichier.c » sont l'outil et l'objet utilisés pour cette activité.

Puis vient le tour du canal « qui ». Son premier résultat est d'indiquer que le réalisateur de l'activité est Arthur. Ensuite, son réseau bayesien utilise les informations fournies pour trouver les noms des personnes collaborant avec Arthur dans son activité. Il utilise pour cela non seulement les mêmes informations bas niveau que les autres canaux, mais aussi les informations produites par les canaux « quoi » et « comment », ainsi que les informations haut niveau reçues des autres membres du groupe. Par exemple, en recevant l'information

que le document « MonFichier.c » est utilisé, et en ayant déjà l'information que Bruce et Arthur sont dans le même sous-département, qu'Arthur et Bruce échangent beaucoup de courriels, et que Bruce utilise aussi le fichier « MonFichier.c », alors le réseau en déduit que Bruce et Arthur collaborent.

L'information bas niveau est ensuite transmise au canal « pourquoi ». Ce dernier utilise les informations bas niveau, mais aussi les résultats du canal « quoi » et du canal « qui » pour en déduire si Arthur effectue son activité dans le cadre normal de son travail (ajout de fonctionnalités) ou bien en réponse à une demande particulière. Ainsi, en recevant l'information qu'Arthur est en train de coder, et en ayant déjà l'information qu'Arthur et Bruce s'échangent des courriels mais qu'Arthur et Bruce collaborent, le réseau bayesien en déduit qu'Arthur effectue son activité dans le cadre général de son travail. Par contre, après avoir reçu le courriel de Clark, et après qu'Arthur ait fermé tous ses fichiers pour en ouvrir d'autres différents, le réseau bayesien en déduit qu'Arthur répond maintenant à une demande de Clark.

Le canal « où » est un peu particulier, car il ne s'intéresse qu'aux informations bas niveau automatique et faiblement dynamique : le type de l'ordinateur, le type de connexion réseau et l'adresse IP. L'information bas niveau indique si l'ordinateur est un ordinateur de bureau, un ordinateur portable ou bien un *PDA*. Cette information est généralement directement donnée par le système. Le type de connexion réseau et l'adresse IP permettent d'indiquer, avec quelques règles simples, si l'utilisateur est au bureau, chez lui ou bien en voyage. Pour Arthur par exemple, la réponse du canal « où » est qu'Arthur est au bureau, sur son ordinateur de bureau.

Finalement, le canal « quand » ajoute la date et l'heure aux informations reçues. Par exemple, lorsqu'il reçoit une information sur l'ouverture d'un document, il donne la date et l'heure de l'ouverture du document. Il stocke aussi cette information. Lorsqu'il reçoit une information sur la fermeture du document, il donne la date et l'heure de la fermeture, ainsi que la durée totale d'ouverture du document.

Une fois qu'une information bas niveau a fait le tour des six canaux d'agrégation, le canal global d'agrégation récupère l'ensemble des réponses des six canaux et l'agrège en une information haut niveau synthétique. Cette information n'est créée que si elle traduit un changement dans l'activité de l'utilisateur. Ainsi, si Arthur est toujours en train de coder, et qu'il n'a changé ni de documents, ni de cadre général, ni de personnes avec lesquelles il coopère, alors il n'est pas utile de créer une nouvelle information agrégée.

Dans le chapitre précédent, l'analyse cognitive des mécanismes de la conscience de groupe a montré que l'outil doit fournir à l'utilisateur non seulement les informations concernant les activités des autres membres du groupe, mais aussi celles concernant sa propre activité. L'outil de l'utilisateur reçoit les informations haut niveau des autres membres du groupe et les représente pour que l'utilisateur puisse construire sa conscience de groupe. Afin de répondre aux besoins fonctionnels établis dans le chapitre précédent, l'outil doit donc aussi recevoir ses propres informations haut niveau, représentant les activités de l'utilisateur, et les visualiser. Ainsi, lorsqu'une nouvelle information haut niveau agrégée est créée, elle est non seulement transmise à l'étape suivante pour être diffusée aux membres du groupe, mais aussi transmise directement à l'étape de réception du même outil afin d'être archivée, adaptée et représentée.

3.3 Filtrage des informations

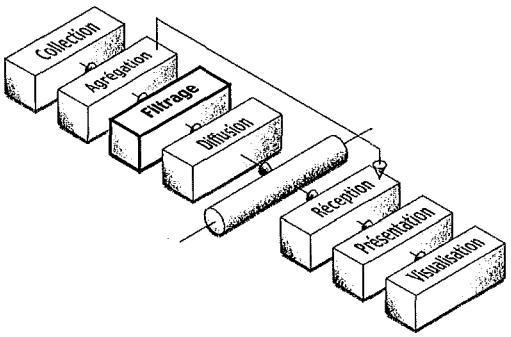

FIG. 11 – Place de l'étape de filtrage dans l'architecture fonctionnelle.

Le but de cette étape est de filtrer les informations à diffuser aux autres membres du groupe (cf. fig. 11). L'étape précédente a créé une information haut niveau décrivant l'activité courante de l'utilisateur. Cette information doit être diffusée aux membres du groupe afin qu'ils puissent construire leur conscience de groupe. Mais l'utilisateur peut considérer que certaines parties de l'information sont personnelles. Il peut cependant accepter de partager certaines informations avec certaines personnes, mais pas avec d'autres. De plus, toutes les informations concernant l'activité de l'utilisateur ne sont pas forcément utiles et intéressantes pour tous les membres du groupe. Le but de l'étape de filtrage est donc double : permettre à l'utilisateur de choisir les informations à diffuser aux autres membres du groupe, et diffuser des informations plus pertinentes à chacun. Le filtrage des informations doit donc être adapté non seulement en fonction de la personne à qui l'information est envoyée, mais aussi de l'information elle-même.

Cette étape de filtrage est composée de deux parties : une partie en charge de l'étiquetage des informations, et une partie en charge des décisions de filtrage. La partie d'étiquetage assigne à chacune des composantes de l'information à filtrer un niveau de détail. La partie de décision regroupe toutes les composantes de l'information avec leur niveaux de détail associés, et décide du contenu de l'information haut niveau envoyée.

Partie étiquetage

Plusieurs techniques sont envisageables pour réaliser la partie *étiquetage* du filtrage des informations. Une solution classique consiste à utiliser des règles, reliant le destinataire d'une information et le type de cette dernière à un niveau de détail prédéterminé. Cependant, l'utilisation de règles pose plusieurs problèmes :

- les règles sont difficiles à changer, et peu évolutives ;
- les règles doivent être entièrement connues avant d'être codées.

Pour éviter ces problèmes, une autre technique est possible : la technique des réseaux de neurones. Un *réseau neuronal* [Nil98], ou *réseau de neurones*, est une technique d'intelligence artificielle permettant de découvrir et d'implanter une relation *a priori* inconnue entre des entrées et des sorties. Le nom de cette technique vient du fait qu'elle est inspirée du fonctionnement des neurones biologiques. Un réseau neuronal est composé de neurones, reliés entre eux par des liaisons pondérées. Les liaisons pondérées servent à transmettre les signaux (des valeurs) entre les neurones. La valeur transmise est égale à la valeur émise pondérée par le poids de la liaison ayant effectué la transmission. Lorsqu'un neurone reçoit des signaux sur les liaisons pondérées qui arrivent jusqu'à lui, il émet un signal, dont la valeur est déterminée en fonction de la somme des signaux reçus. Ce signal est transmis

sur l'ensemble des liaisons pondérées partant de ce neurone, et est ainsi propagé à d'autres neurones. Un signal appliqué sur un neurone d'entrée crée ainsi dans le réseau neuronal un ensemble de signaux qui se diffusent de proche en proche jusqu'aux neurones de sortie. Les signaux émis par ces neurones de sortie forment la réponse du réseau de neurone.

Le but d'un réseau de neurones est d'implanter une fonction, au sens mathématique du terme, c'est-à-dire associant une valeur à une autre valeur. Pour cela, le réseau doit apprendre la fonction à partir de cas particuliers. Au départ, le réseau neuronal est entraîné avec en entrée des valeurs spécifiques pour lesquelles les valeurs de sorties sont connues. Pour obtenir les sorties voulues, le réseau neuronal modifie les poids de ses liaisons pondérées, et approxime ainsi la fonction souhaitée. Une fois l'entraînement terminé, le réseau neuronal peut non seulement calculer les valeurs de sortie de la fonction pour les valeurs données en exemple, mais il peut aussi, si l'apprentissage a été bien fait, calculer la sortie de la fonction pour n'importe quelle entrée.

Le double avantage de la technique des réseaux de neurones provient de cette capacité d'apprentissage. Tout d'abord, il n'est pas nécessaire de spécifier entièrement tous les niveaux de détail de chaque type d'information pour chaque destinataire, puisque le réseau neuronal apprend au fur et à mesure qu'apparaissent de nouvelles situations. De plus, le réseau neuronal est bien plus évolutif qu'un ensemble de règles : il suffit que l'utilisateur corrige une sortie du réseau neuronal pour que ce dernier apprenne la nouvelle valeur.

Dans l'outil de conscience de groupe, le réseau neuronal étiquete les informations à diffuser aux membres du groupe. Lorsqu'une information est reçue de l'étape précédente, le réseau neuronal est appelé une fois pour chaque membre du groupe, afin de déterminer quels détails de l'information doivent être transmis à cette personne. Pour cela, le réseau neuronal prend en entrée l'information à transmettre, la personne à qui l'information doit être transmise et le contexte de travail de l'utilisateur, et fait correspondre à ces entrées une sortie qui détermine le niveau de détail de l'information à diffuser à cette personne.

Le réseau neuronal reçoit trois catégories de valeurs d'entrées :

- des valeurs représentant l'information à diffuser ;
- des valeurs représentant le futur récepteur de la diffusion ;
- des valeurs représentant des informations sur le contexte de travail de l'utilisateur.

L'information à diffuser a été produite par l'étape précédente. Elle décrit l'activité courante de l'utilisateur, en regroupant des informations des six catégories « quoi », « comment », « qui », « pourquoi », « où » et « quand ». Pour pouvoir être passées en entrée du réseau neuronal, ces informations doivent être représentées par des valeurs entières. L'information de la catégorie « quoi » par exemple contient le type de l'activité réalisée par l'utilisateur. Lorsque l'ensemble des types d'activités possibles est spécifié, pendant la phase de configuration de l'outil, un entier est attribué à chacun des types. C'est cet entier qui sert à représenter l'information de la catégorie « quoi ». Le même principe est utilisé pour les informations des catégories « comment », « qui », « pourquoi » et « où ». L'information « quand » est représentée, comme d'habitude en informatique, par un nombre de secondes écoulées depuis une certaine date (le 1er janvier 1970 par exemple). Chacun des composants de l'information à diffuser est ainsi fourni, sous forme d'entier, en entrée du réseau neuronal.

Le même cas se présente pour passer au réseau neuronal le nom de la personne à qui l'information doit être transmise. Lors de la spécification de la structure du groupe,

pendant la phase de configuration de l'outil, un entier est associé à chaque membre du groupe. C'est cet entier qui est fourni en entrée du réseau neuronal pour représenter le futur récepteur de l'information.

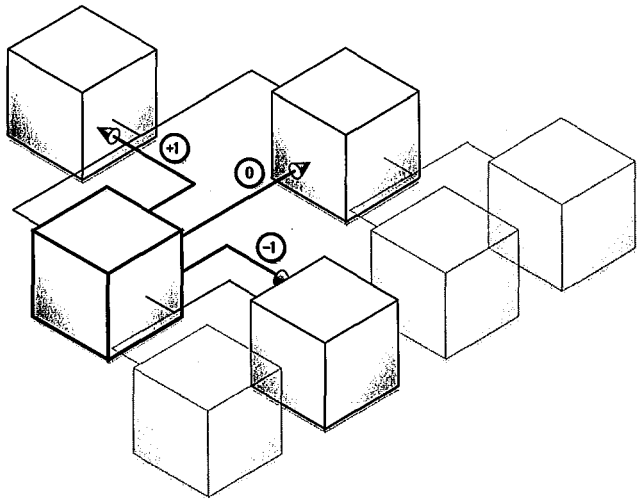

FIG. 12 – Le calcul de la distance suivant la place du récepteur par rapport à l'utilisateur.

Finalement, les dernières entrées du réseau neuronal sont des informations issues du contexte de travail de l'utilisateur et portant sur les relations entre l'utilisateur et le futur récepteur : le récepteur collabore-t-il avec l'utilisateur ? Le récepteur est-il dans la liste des personnes proches de l'utilisateur ? Quelle est la place du récepteur par rapport à l'utilisateur dans la structure hiérarchique du groupe ? Cette dernière information est calculée sous la forme d'une distance entre l'utilisateur et le récepteur. La distance est négative lorsque le récepteur est un subalterne de l'utilisateur, positive lorsque le récepteur est un chef, nulle lorsque le récepteur est au même niveau que l'utilisateur, et une valeur spécifique lorsque le récepteur n'est pas dans la même branche de la hiérarchie que l'utilisateur (cf. fig. 12).

Partie décisions

Les valeurs de sortie du réseau neuronal déterminent le niveau de détail de l'information à diffuser à la personne concernée. Il y a une valeur de sortie pour chacune des six catégories composant l'information à diffuser. Chaque valeur indique quelles sont les informations de la catégorie correspondante à transmettre. Les différentes valeurs possibles ainsi que leur signification dépendent de la catégorie :

Catégorie « quoi »

valeur	*signification*
vide	aucune information n'est transmise
restreinte	seul un indicateur d'activité (*Actif/Inactif*) est transmis
complète	l'information complète (type de l'activité) est transmise

Catégorie « comment »

valeur	*signification*
vide	aucune information n'est transmise
complète	l'information complète (outils et objets utilisés) est transmise

Catégorie « qui »

valeur	*signification*
restreinte	seul le nom du réalisateur de l'activité (l'utilisateur) est transmis
complète	l'information complète (réalisateur et coopérateurs) est transmise

Catégorie « pourquoi »

valeur	*signification*
vide	aucune information n'est transmise
complète	l'information complète (cadre global de l'activité) est transmise

Catégorie « où »

valeur	*signification*
vide	aucune information n'est transmise
complète	l'information complète (type du matériel et de l'endroit) est transmise

Catégorie « quand »

valeur	*signification*
restreinte	seule la date du début de l'activité est transmise
complète	l'information complète (début et durée) est transmise

Il est techniquement possible de définir plus de valeurs et plus de niveaux de détails. Cependant, l'utilisateur doit pouvoir contrôler et modifier les choix du réseau neuronal. Plus le nombre de choix possibles est élevé, plus cela complique la tâche de l'utilisateur. Il est donc important, afin de ne pas surcharger l'utilisateur inutilement, de ne lui présenter que des choix pertinents. Par exemple, donner le choix à l'utilisateur de ne transmettre que l'outil utilisé mais pas les objets ou l'inverse n'est pas utile dans la majorité des cas. Supprimer ces choix, qui ne sont utiles que dans des cas très particuliers, limite la flexibilité du filtrage mais le rend ainsi plus compréhensible et plus utilisable par l'utilisateur.

Les six valeurs produites par le réseau neuronal sont utilisées dans la partie décision pour filtrer l'information haut niveau. La première valeur utilisée est celle de la catégorie « quoi ». Si cette valeur vaut *vide*, aucune information, même des autres catégories, n'est transmise, et cela quelles que soient les autres valeurs de sortie. Aucun message n'est alors envoyé à la personne concernée. Si la valeur vaut *restreinte*, les seules autres informations

transmises sont les informations *restreintes* des catégories « qui » et « quand », et cela quelles que soient les autres valeurs de sortie du réseau neuronal. Le message envoyé indique donc seulement si l'utilisateur est actif ou inactif, et depuis quand. L'information indiquant si l'utilisateur est actif ou inactif est construite à partir de l'information de la catégorie « quoi ». Si cette information indique que l'activité de l'utilisateur est *inactif*, alors c'est cette information qui est envoyée. Dans tous les autres cas, c'est l'information *actif* qui est envoyée. Enfin, si la valeur de la catégorie « quoi » vaut *complète*, alors les autres valeurs de sorties du réseau neuronal sont utilisées comme spécifié.

Contrôle utilisateur

Le but général de l'outil est de réduire la surcharge de l'utilisateur. Pour arriver à ce but, les actions de l'outil doivent être le plus automatiques possible. Cependant, avec un outil trop automatique, l'utilisateur peut se sentir pris au piège d'un système qu'il ne comprend pas et qu'il ne maîtrise plus [Hor99, Shn97]. De plus, l'utilisateur doit garder le contrôle des informations diffusées. Le système doit donc toujours fournir à l'utilisateur assez de retour sur ses actions, afin que ce dernier puisse apporter les corrections nécessaires si besoin est, et ainsi garder son sentiment de contrôle sur le système.

Dans le cas de l'outil et des choix du réseau neuronal, les messages sont présentés à l'utilisateur avant d'être envoyés à l'étape suivante (l'étape en charge de la diffusion). L'utilisateur a le choix entre trois modes :

- le mode « confiance » ;
- le mode « contrôle » ;
- le mode « secret ».

Dans le mode « confiance », les messages sont présentés à l'utilisateur pendant un laps de temps réglable. Pendant ce délai, l'utilisateur peut modifier ou annuler le message. Si au bout de ce délai le message n'a pas été annulé, il est diffusé. Ce mode a l'avantage de ne demander qu'un minimum de travail à l'utilisateur, dans la plupart des cas une simple vérification visuelle, tout en permettant un contrôle adéquat sur les informations diffusées. C'est le mode normal de fonctionnement de l'outil. Dans le mode « contrôle », les messages sont présentés à l'utilisateur et attendent son accord pour être diffusés. L'utilisateur peut ainsi contrôler et modifier quand et comme il le veut toutes les informations diffusées, en échange d'une surcharge de travail. Ce mode est principalement prévu pour les premières utilisations de l'outil. Enfin dans le dernier mode, le mode « secret », aucun message n'est envoyé. Ce mode permet à l'utilisateur de garder un contrôle complet de sa vie privée, mais aux dépens de la conscience de groupe des autres membres. Dans ce mode, l'utilisateur continue cependant à recevoir les messages des autres membres du groupe, ainsi que ses propres messages. Ce mode ne devrait être utilisé que dans des cas spécifiques et peu fréquents. L'utilisateur peut passer d'un mode à l'autre quand il le souhaite.

L'utilisateur peut modifier un message en changeant le niveau de détail de n'importe quelle catégorie. Par exemple, un utilisateur auquel est présenté un message contenant, dans la catégorie « pourquoi », une information complète indiquant que l'utilisateur répond à une demande d'une certaine personne, peut décider de baisser le niveau de détail dans cette catégorie de *complète* à *vide*. Le message est alors diffusé, mais sans information dans la catégorie « pourquoi ». L'utilisateur peut ainsi modifier le niveau de détail d'autant

de catégories qu'il le souhaite. L'utilisateur peut aussi décider d'annuler purement et simplement le message.

Lorsque l'utilisateur modifie un message, il modifie un choix fait par le réseau de neurones. Cette modification, renvoyée au réseau de neurones, permet à ce dernier d'apprendre le choix de l'utilisateur et de modifier les poids de ses liaisons en conséquence. À la prochaine occasion similaire, la valeur fournie par le réseau de neurones reflétera le choix de l'utilisateur. Ainsi, au fur et à mesure de son utilisation, le réseau neuronal s'adapte aux préférences spécifiques de l'utilisateur concernant la propagation d'information.

Les valeurs acceptables en entrée du réseau neuronal sont spécifiées pendant la phase de configuration, car elles dépendent des valeurs que peuvent prendre les informations haut niveau. Toujours dans la phase de configuration, une fois les entrées spécifiées, le réseau neuronal est entraîné avec des données définies à partir de règles simples déterminant *a priori* les informations à échanger suivant les relations entre les personnes, comme par exemple :

- une personne collaborant avec l'utilisateur reçoit toutes les informations concernant l'activité coopérative ;
- une personne proche de l'utilisateur reçoit toutes les informations sur les activités de l'utilisateur ;

Ces règles ne servent que pour l'entraînement initial du réseau neuronal. Cet entraînement permet de donner au réseau neuronal un comportement par défaut. Ensuite, lors de son utilisation, les choix de l'utilisateur complètent son apprentissage et l'adaptent aux préférences de l'utilisateur. Les règles utilisées sont donc importantes pour les premières utilisations du réseau neuronal. Plus ces règles seront pertinentes, moins l'utilisateur aura besoin de modifier les choix initiaux du réseau neuronal. Mais après un certain temps d'utilisation, le réseau est adapté aux préférences de l'utilisateur, et son comportement n'a plus de relation avec les règles utilisées pour son apprentissage initial.

Exemple

Dans l'exemple de l'équipe fictive de développement de jeux vidéo, lorsqu'Arthur travaille sur la demande de Clark, l'étape précédente produit l'information haut niveau suivante :

CATÉGORIE	INTITULÉ	INFORMATION
quoi	*activité*	coder
comment	*outil*	ProjectBuilder
	objet	« MonFichier.c »
qui	*réalisateur*	Arthur
	coopérateur	Bruce
pourquoi	*cadre*	demande de Clark
où	*équipement*	ordinateur de bureau
	lieu	bureau
quand	*début*	29 août 2003, 15h30
	durée	2 heures

Cette information haut niveau doit être diffusée aux différents membres de l'équipe, notamment Bruce, son collègue de bureau, Clark, le responsable de la conception d'un niveau, et Diana, la graphiste du département « décors ». Le réseau neuronal est appelé une fois pour chaque personne.

Pour Bruce, le réseau neuronal reçoit, en plus de l'information haut niveau et du nom « Bruce », les informations suivantes :

- Bruce collabore avec Arthur pour l'activité en cours ;
- Bruce est une personne proche d'Arthur ;
- Bruce est dans le même sous-département qu'Arthur.

Bruce a ainsi trois bonnes raisons de recevoir un maximum d'informations sur l'activité d'Arthur. En se basant sur ces entrées, les sorties du réseau neuronal indiquent toutes que l'information complète doit être diffusée à Bruce. L'étape suivante reçoit donc l'information ci-après :

CATÉGORIE	SORTIE	INTITULÉ	INFORMATION
quoi	*complète*	*activité*	coder
comment	*complète*	*outil*	ProjectBuilder
		objet	« MonFichier.c »
qui	*complète*	*réalisateur*	Arthur
		coopérateur	Bruce
pourquoi	*complète*	*cadre*	demande de Clark
où	*complète*	*équipement*	ordinateur de bureau
		lieu	bureau
quand	*complète*	*début*	29 août 2003, 15h30
		durée	2 heures

Clark ne collabore pas avec Arthur. Ce n'est pas non plus une personne proche d'Arthur. Enfin, Clark travaille dans un autre département, situé dans une branche différente de la hiérarchie. Cependant, Clark est à l'origine de l'activité courante d'Arthur, comme l'indique le champ « quoi » de l'information haut niveau. L'information haut niveau fournie à Clark n'est donc pas complète, mais possède assez de détails pour que ce dernier puisse savoir ce que fait Arthur. Le réseau neuronal décide ainsi d'envoyer à l'étape suivante l'information ci-après :

CATÉGORIE	SORTIE	INTITULÉ	INFORMATION
quoi	*complète*	*activité*	coder
comment	*vide*	*outil*	*vide*
		objet	*vide*
qui	*complète*	*réalisateur*	Arthur
		coopérateur	Bruce
pourquoi	*complète*	*cadre*	demande de Clark
où	*vide*	*équipement*	*vide*
		lieu	*vide*
quand	*complète*	*début*	29 août 2003, 15h30
		durée	2 heures

Diana n'est pas une collaboratrice d'Arthur. Elle n'est pas non plus dans la même branche. Mais elle est notée comme personne proche. De plus, il y a quelques mois, Arthur a commencé à changer les propositions du réseau neuronal : il décidait toujours d'envoyer l'information complète à Diana. Le réseau s'est donc adapté à ce choix d'Arthur, et il décide ainsi d'envoyer l'information complète à l'étape suivante :

CATÉGORIE	SORTIE	INTITULÉ	INFORMATION
quoi	*complète*	*activité*	coder
comment	*complète*	*outil*	ProjectBuilder
		objet	« MonFichier.c »
qui	*complète*	*réalisateur*	Arthur
		coopérateur	Bruce
pourquoi	*complète*	*cadre*	demande de Clark
où	*complète*	*équipement*	ordinateur de bureau
		lieu	bureau
quand	*complète*	*début*	29 août 2003, 15h30
		durée	2 heures

Finalement, pour les autres membres de l'équipe qui ne sont ni proches, ni collaborateurs, ni dans le même département, l'information envoyée est minimale :

CATÉGORIE	SORTIE	INTITULÉ	INFORMATION
quoi	*restreinte*	*activité*	*Actif*
comment	*vide*	*outil*	*vide*
		objet	*vide*
qui	*restreinte*	*réalisateur*	Arthur
		coopérateur	*vide*
pourquoi	*vide*	*cadre*	*vide*
où	*vide*	*équipement*	*vide*
		lieu	*vide*
quand	*restreinte*	*début*	29 août 2003, 15h30
		durée	*vide*

Arthur étant en mode « confiant », toutes ces informations lui sont présentées à l'écran pendant un temps configurable. Arthur vérifie du coin de l'œil que les informations correspondent bien à ces choix. En particulier, il vérifie que les informations envoyées à Clark ne sont pas trop personnelles, et qu'au contraire les informations envoyées à Diana sont bien complètes. Au bout du temps imparti, si l'information n'a pas été annulée, elle passe à l'étape suivante, qui se charge de sa diffusion à la personne concernée.

3.4 Diffusion des informations

Le but de cette étape est de diffuser l'information reçue de l'étape précédente à la personne concernée (cf. fig. 13). En effet, cette information peut être vue comme une

description à un instant précis du contexte de travail de l'utilisateur. Associées aux informations reçues des autres membres du groupe, elle permet à la personne qui la reçoit de construire sa conscience de groupe.

La diffusion de l'information se fait à l'aide d'un réseau d'« égal à égal » (*peer-to-peer*). Dans ce type de réseau, toutes les communications se font directement d'un poste à l'autre, sans passer par un serveur. L'absence de serveur permet d'éviter que les informations transmises ne transitent par une autre machine que celles de l'émetteur et du diffuseur. Cette caractéristique est importante, car c'est une condition nécessaire pour assurer la protection de la vie privée de l'utilisateur. En effet, certaines des informations diffusées peuvent être considérées comme personnelles. Lorsque ces données personnelles sont diffusées aux personnes choisies, aucun autre membre du groupe ne doit pouvoir y avoir accès, même pas l'administrateur de l'outil. En effet, dans certains cadres d'utilisation, la pression exercée sur ce dernier pourrait être trop grande. Avec un réseau d'« égal à égal », les informations sont envoyées directement à l'utilisateur concerné, sans être sauvegardées ni même transiter sur un serveur central.

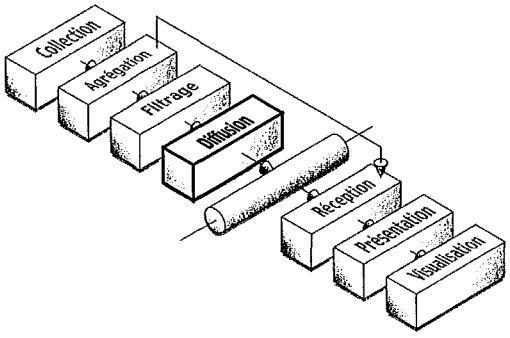

FIG. 13 – Place de l'étape de diffusion dans l'architecture fonctionnelle.

Si le destinataire de l'information est déconnecté, ou si son outil de conscience de groupe n'est pas accessible par le réseau, l'information est sauvegardée dans un tampon d'attente. L'outil gère un tampon d'attente par destinataire possible. Dès que l'utilisateur est de nouveau en ligne, toutes les informations en attente sont envoyées dans un message unique.

3.5 Réception des informations

Le rôle de cette étape est de recevoir des informations haut niveau et de les sauvegarder, puis de les passer à l'étape suivante pour leur représentation (cf. fig. 14).

Les informations haut niveau reçues dans cette étape peuvent provenir soit des outils des autres membres du groupe, soit du propre outil de l'utilisateur, comme vu à la fin de l'étape d'agrégation. Quelle que soit sa provenance, lorsqu'une information haut niveau est reçue, elle est sauvegardée dans le gestionnaire de contexte. Ce module contient toutes les informations haut niveau reçues par l'outil, classées suivant leur date de création. Les réseaux de neurones des étapes de filtrage et d'adaptation font appel à lui pour leur fournir les informations nécessaires. Ces informations sauvegardées constituent donc une partie du contexte de travail de l'utilisateur.

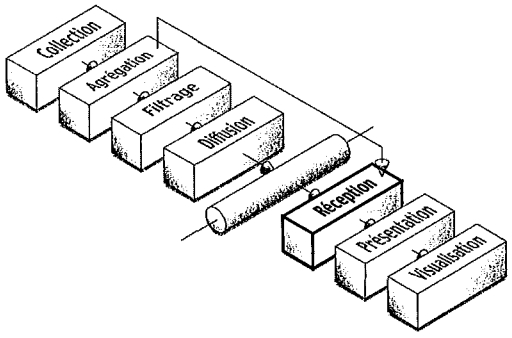

FIG. 14 – Place de l'étape de réception dans l'architecture fonctionnelle.

Certaines informations conservées par l'outil peuvent être considérées comme personnelles par l'utilisateur. Il est donc important que ce dernier puisse garder le contrôle sur toutes les informations conservées par l'outil. Pour cette raison, toutes les informations sont conservées dans un format lisible et compréhensible par un humain. Ainsi, l'utilisateur a la possibilité, à tout moment, d'accéder, de modifier ou d'effacer tout ou partie de ces données. Il faut de plus assurer l'utilisateur qu'il est le seul à avoir accès aux informations conservées par l'outil. Pour cela, l'utilisateur peut protéger ces informations des regards indiscrets d'autres personnes par un mécanisme de chiffrement. Ces deux fonctionnalités sont nécessaires pour garantir le respect de la vie privée de l'utilisateur, comme cela est décrit au début de ce chapitre.

Une fois sauvegardée dans le gestionnaire de contexte, l'information est envoyée à l'étape suivante pour être présentée. Lorsque l'utilisateur est déconnecté, il ne peut pas recevoir les informations haut niveau des autres membres du groupe. Ces informations sont alors sauvées du côté de l'émetteur dans un tampon d'attente. Dès que l'utilisateur est de nouveau accessible sur le réseau, toutes les informations en attente sont envoyées dans un message unique. Lorsque l'outil de l'utilisateur reçoit un tel message, il commence par en extraire les différentes informations contenues, qu'il classe suivant leur date de création. Toutes ces informations sont sauvées dans le gestionnaire de contexte, mais seule la plus récente est envoyée à l'étape suivante pour être représentée. La sauvegarde de l'ensemble des informations permet à l'utilisateur d'avoir accès à un historique de ce qui s'est passé pendant sa déconnexion. La représentation d'une seule information, la plus récente, permet de présenter à l'utilisateur l'état courant de l'activité de l'émetteur tout en évitant de surcharger l'interface de l'utilisateur par la représentation successive et rapide d'un trop grand nombre d'informations.

3.6 Adaptation de la présentation

Le but de cette étape est d'adapter au contexte de travail de l'utilisateur la présentation des informations haut niveau reçues de l'étape précédente (cf. fig. 15). Dans cette étape, l'outil détermine, en fonction de l'information reçue, de la personne l'ayant émise et du contexte de travail de l'utilisateur, le niveau d'intrusion avec lequel l'information doit être présentée à l'utilisateur.

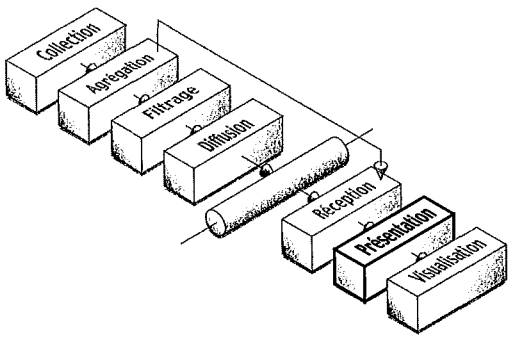

FIG. 15 – Place de l'étape de présentation dans l'architecture fonctionnelle.

L'étape d'adaptation n'est qu'une étape d'étiquetage. Elle assigne à chaque information reçue un niveau d'intrusion. C'est ensuite à l'étape suivante de décider *comment* représenter l'information en fonction du niveau d'intrusion qui lui a été assigné.

L'adaptation de la présentation d'une information se fait suivant deux critères : l'importance de l'information pour l'activité de l'utilisateur, et le niveau d'intrusion que l'activité de ce dernier peut supporter. Si l'utilisateur est dans une activité peu interruptible

(comme par exemple écrire ou coder), ou bien si l'information est peu pertinente pour l'activité de l'utilisateur, alors la notification est présentée d'une manière peu intrusive. Inversement, si l'information est primordiale pour l'activité de l'utilisateur (par exemple parce qu'elle indique un problème sur le fichier utilisé), alors la notification est présentée de manière à ne pas pouvoir échapper à l'attention de l'utilisateur. Cette adaptation permet aussi de filtrer les informations reçues en fonction de leur pertinence : si une information n'est pas pertinente, alors son niveau d'intrusion est minimal, et elle n'est pas présentée à l'utilisateur.

Plusieurs techniques sont envisageables pour réaliser l'adaptation de la présentation des informations. Une solution classique consiste à utiliser des règles, associant le type de l'information et le contexte de travail de l'utilisateur à un niveau d'intrusion prédéterminé. Cette solution souffre des mêmes problèmes que ceux expliqués dans la partie précédente. Pour les mêmes raisons, la technique des réseaux de neurones est utilisée. Le but du réseau neuronal est de déterminer le niveau d'intrusion à utiliser pour représenter l'information reçue. Pour cela, le réseau neuronal se base sur l'information à présenter et le contexte de travail de l'utilisateur. La sortie du réseau neuronal correspond au niveau d'intrusion à utiliser.

Le réseau neuronal ne détermine pas directement le type de représentation à utiliser : c'est le rôle de l'étape suivante. Le but du réseau neuronal est seulement de déterminer le niveau d'intrusion à utiliser pour représenter l'information reçue. Pour cela, le réseau neuronal se base sur l'information à présenter et le contexte de travail de l'utilisateur. La sortie du réseau neuronal correspond au niveau d'intrusion à utiliser.

Le réseau neuronal prend trois types d'information en entrée :

- l'information à représenter ;
- la dernière information décrivant l'activité de l'utilisateur ;
- des informations sur les relations entre l'utilisateur et l'émetteur de l'information à représenter.

La manière dont le contenu d'une information haut niveau est représentée par des valeurs entières a été expliquée dans l'étape de filtrage. Les deux premiers types d'information passées en entrée du réseau neuronal correspondent à des informations haut niveau représentées de cette manière. La deuxième information permet notamment de passer au réseau neuronal une description de l'activité courante de l'utilisateur. L'outil passe aussi au réseau neuronal des informations concernant les relations entre l'utilisateur et l'émetteur de l'information reçue : l'émetteur collabore-t-il avec l'utilisateur ? L'émetteur est-il dans la liste des personnes proches de l'utilisateur ? Quelle est la distance entre l'émetteur et l'utilisateur ? Cette distance est calculée de la même manière que dans l'étape de filtrage.

La sortie du réseau neuronal correspond à un niveau donné sous la forme d'une valeur entière comprise entre moins un (-1) et trois (3). Ce niveau représente le moment et la manière dont l'information doit être présentée. Plus le niveau est élevé, plus la présentation est intrusive :

valeur	*signification*
−1	information rejetée
0	information à réévaluer
1	présentation très peu intrusive
2	présentation normale
3	information à ne pas rater

Une valeur négative (−1) signifie que l'information n'est pas pertinente pour l'activité de l'utilisateur et n'a pas à lui être présentée. Cette information reste dans le gestionnaire de contexte de l'étape précédente, mais n'est pas passée à l'étape suivante.

Une valeur nulle (0) signifie que l'information est pertinente, mais que le moment n'est pas adéquat pour la présenter. C'est le cas, par exemple, lorsque l'utilisateur est en train de taper une phrase et ne doit pas être interrompu. Dans ce cas, l'information n'est pas présentée, mais est placée dans un tampon d'attente. À chaque changement dans le contexte de l'utilisateur, toutes les informations contenues dans le tampon d'attente sont réévaluées par le réseau neuronal. Puisque le contexte de travail de l'utilisateur est différent, la sortie du réseau neuronal peut l'être aussi. L'information est alors soit rejetée (avec une valeur négative (−1)), soit remise dans le tampon d'attente (avec une valeur nulle (0)), soit enfin présentée (avec une valeur strictement positive (1, 2 ou 3)).

Une valeur strictement positive (1, 2 ou 3) signifie que l'information doit être présentée maintenant à l'utilisateur. Une valeur de un (1) signifie que l'information doit être présentée de la manière la moins intrusive possible. Une telle information peut être représentée, par exemple, par un changement de couleur dans un coin de l'écran, en vision périphérique. Une valeur de deux (2) signifie que l'information doit être présentée d'une manière normale, par exemple par un mouvement dans la vision périphérique (la vision périphérique est très sensible aux mouvements [DFAB98]), ou bien par un son doux. Une valeur de trois (3) signifie que l'utilisateur ne doit manquer l'information sous aucun prétexte, et que celle-ci doit être représentée d'une manière intrusive si besoin est. Cela peut signifier, par exemple, l'utilisation d'une boîte de dialogue assortie d'un son d'alerte. Ce niveau d'interruption doit normalement être utilisé le moins souvent possible.

Lorsqu'une information est présentée à l'utilisateur, ce dernier peut vérifier et changer le niveau d'interruption utilisé. Comme dans l'étape de filtrage, le choix de l'utilisateur est renvoyé au réseau neuronal, qui l'apprend et ainsi s'adapte aux préférences de l'utilisateur.

Comme pour le réseau neuronal utilisé dans l'étape de filtrage, la spécification des valeurs d'entrées possibles et l'entraînement du réseau ont lieu pendant la phase de configuration de l'outil. Les données utilisées pour l'entraînement sont définies à partir de règles simples déterminant *a priori* le niveau d'interruption à utiliser, comme par exemple :

- une information portant sur le changement d'outil d'une personne sans aucun lien avec l'utilisateur ne doit pas lui être présentée ;
- une information sur un changement d'activité d'une personne doit être présentée de la manière la moins intrusive possible ;
- une information portant sur le même objet que celui sur lequel l'utilisateur travaille en ce moment mais dont le sujet est une personne collaborant avec l'utilisateur doit être présentée d'une manière normale ;
- la même information mais dont le sujet est une personne sans lien avec l'utilisateur

doit être présentée de manière à ce que l'utilisateur ne puisse pas la rater.
Comme dans l'étape de filtrage, ces règles déterminent principalement le comportement par défaut de l'outil. Après un certain temps d'utilisation, le réseau est adapté aux préférences de l'utilisateur, et son comportement n'a plus de relations avec les règles utilisées pour son apprentissage initial.

Exemple

Dans l'exemple de l'équipe fictive de développement de jeux vidéo, Arthur, lors de son travail sur la demande de Clark, reçoit un certain nombre d'informations, particulièrement de Bruce, Clark et Diana.

Diana, par exemple, est en train d'illustrer un décor 3D. Pour cela, elle change souvent d'outil, passant de l'un à l'autre suivant ses besoins. À chaque changement d'outil, une information haut niveau est générée. Arthur est dans la liste des personnes proches de Diana. Il reçoit donc des informations complètes de la part de l'outil de Diana. Mais ce qui intéresse Arthur, ce sont les informations concernant le type et la durée des activités de Diana, pas le détail des outils utilisés ou des objets ouverts. Lorsque l'information concernant le changement d'outil de Diana arrive sur l'outil d'Arthur, le réseau neuronal lui attribue un niveau d'interruption négatif (-1). L'information est donc sauvée dans le gestionnaire de contexte d'Arthur, mais ne lui est pas présentée.

Bruce, lui, est en train de coder sur le même projet qu'Arthur lorsque, pour résoudre un problème, il se met à lire de la documentation technique. Son outil diffuse alors une information indiquant le changement d'activité. Arthur, en tant que collaborateur et personne proche de Bruce, reçoit des informations complètes de la part de l'outil de Bruce. Cependant, lorsque l'information indiquant le changement d'activité de Bruce est reçue, Arthur est absorbé dans l'écriture d'un bloc de code. Le réseau neuronal estime que l'activité d'Arthur ne doit pas être interrompue, mais que l'information mérite quand même d'être présentée. Il lui attribue un niveau de zéro (0). L'information n'est pas présentée, mais est placée dans un tampon d'attente. Lorsqu'Arthur a fini son bloc de code, son outil génère une nouvelle information haut niveau indiquant le changement d'activité. L'information dans le tampon d'attente est alors réévaluée. Le réseau la considère comme peu importante pour l'activité d'Arthur. Il lui attribue donc un niveau de un (1). L'information doit être présentée de la manière la moins intrusive possible.

Clark, en voyage d'affaire, connecte son *PDA* dans un cyber-café de la région. Son outil, reconnaissant que la connexion réseau est rétablie, envoie un message aux autres outils pour signaler l'accessibilité de Clark sur le réseau. Cette information est notamment reçue par Arthur. Puisque Clark est à l'origine de l'activité en cours d'Arthur, le réseau neuronal de ce dernier considère que l'accessibilité de Clark sur le réseau est une information tout à fait pertinente pour l'activité en cours d'Arthur. En effet, ce dernier peut par exemple en profiter pour lui poser des questions concernant le travail à faire. Le réseau neuronal d'Arthur attribue donc un niveau de deux (2) à cette information. L'information doit être présentée normalement.

Clark, toujours connecté dans son cyber-café, décide de voir où en est Arthur. Il télécharge donc sur son *PDA* le fichier sur lequel Arthur est en train de travailler, et commence à le lire. En le lisant, il se rend compte qu'Arthur n'a pas du tout compris la

fonctionnalité qu'il souhaitait obtenir. Clark commence donc à ajouter ses commentaires au fichier d'Arthur, afin de le lui signaler. Lorsque Clark modifie le fichier d'Arthur, l'outil de Clark diffuse une information de haut niveau, ayant pour activité « annotation d'un fichier » et comme objet le fichier d'Arthur. Lorsque le système d'Arthur reçoit cette information, il la considère comme extrêmement importante. En effet, le fichier sur lequel Arthur travaille est en train d'être modifié par une autre personne, qui ne collabore pas avec Arthur, mais qui est à l'origine de l'activité d'Arthur. Le réseau neuronal attribue donc le niveau trois (3) à cette information : Arthur doit absolument voir cette information, même si cela interrompt son activité en cours.

Lorsque le niveau d'interruption retourné par le réseau est strictement positif (1, 2 ou 3), l'information concernée est envoyée, avec son niveau d'interruption, à l'étape suivante qui se charge de sa représentation sur l'équipement de l'utilisateur.

3.7 Visualisation des informations

Le but de cette étape est de présenter correctement les informations reçues de l'étape précédente (cf. fig. 16). Ces informations décrivent les activités des différents membres du groupe, utilisateur compris, et l'ensemble de ces informations forme le contexte de travail du groupe. La représentation de ces informations a pour objectif de permettre à l'utilisateur de construire sa conscience de groupe.

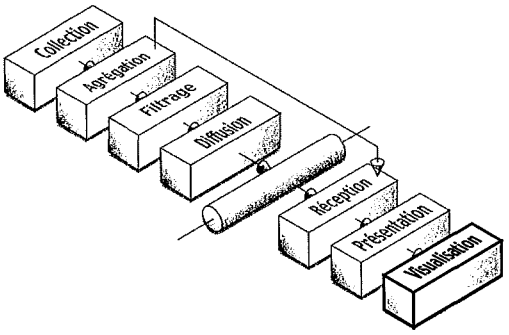

FIG. 16 – Place de l'étape de visualisation dans l'architecture fonctionnelle.

La présentation des informations doit se faire suivant deux modes : un mode périphérique, pour présenter les nouvelles informations d'une manière globale, et un mode actif, permettant à l'utilisateur d'accéder de manière simple mais détaillée à une information spécifique. Ces deux modes de présentation sont l'un des prérequis découlant de l'analyse cognitive de la conscience de groupe du chapitre précédent.

Dans le premier mode, lorsqu'une information haut niveau est reçue de l'étape précédente, l'outil choisit pour cette information la représentation la plus adéquate. Ce choix doit satisfaire plusieurs contraintes. Tout d'abord, la représentation doit respecter le niveau d'interruption fixé lors de l'étape précédente. Ensuite, elle doit être adaptée aux moyens disponibles sur l'équipement du receveur. Sur un *PDA* par exemple, la surface affichable étant bien plus petite que sur un écran d'ordinateur standard, une représentation sonore peut être plus adaptée qu'une représentation graphique. La représentation de l'information est donc différente suivant son niveau d'interruption et suivant le type d'équipement de l'utilisateur.

L'outil peut choisir entre plusieurs techniques de représentation. Une information peut par exemple être représentée par un son, par une parole, par une technique de visualisation (*treemap*, *hypertree*) ou bien par une interface graphique particulière. Chaque technique de représentation est réalisée par un module spécialisé, qui ne s'occupe que de son type de représentation. Le module reçoit l'information avec son niveau d'interruption, et se charge

de la représenter correctement. Chaque module possède trois niveaux de représentations, correspondant aux trois niveaux d'interruption qu'il peut traiter :

niveau 1 : l'information est présentée de manière très peu intrusive ;

niveau 2 : l'information est présentée de manière normale ;

niveau 3 : l'information est présentée de manière à ne pas pouvoir être ratée.

La façon réelle dont l'information est présentée dépend du module. Le découpage de la représentation en modules permet à l'utilisateur de choisir l'interface de son outil en fonction de ses goûts et des capacités de son équipement. L'installation des modules, le choix des modules disponibles, et, si besoin est leur configuration, se fait pendant la phase de configuration de l'outil.

Dans le second mode, l'interface de l'outil doit permettre à l'utilisateur d'accéder à l'ensemble des informations reçues. Cet accès à l'ensemble des informations se fait suivant trois vues : une vue globale, une vue de navigation et une vue en détail. La vue globale sur l'ensemble des informations permet d'obtenir une vision générale de la structure et de la répartition des informations, de faire des comparaisons globales entre les différentes informations, et de repérer des motifs ou des éléments particuliers. À partir de cette vue, l'utilisateur peut définir des sous-ensembles d'informations. Il peut ensuite invoquer sur ces sous-ensembles la vue de navigation. La vue de navigation permet de naviguer dans un ensemble (potentiellement très grand) d'informations et de trouver une information spécifique. L'utilisateur peut obtenir sur cette information une vue en détail pour accéder à la liste des détails composant l'information. La combinaison de ces trois vues respecte la désormais célèbre formule de Shneiderman [Shn97] :

> « D'abord la vue globale, zoom et filtrage, puis les détails à la demande. »[8]

Exemple

Dans l'exemple de l'équipe fictive de développement de jeux vidéo, Arthur, lors de son travail sur la demande de Clark, reçoit les trois informations décrites dans l'étape précédente :

- l'information indiquant le changement d'activité de Bruce, de niveau un (1) ;
- l'information indiquant l'accessibilité de Clark sur le réseau, de niveau deux (2) ;
- l'information indiquant la modification du fichier par Clark, de niveau trois (3).

Arthur travaille sur son ordinateur de bureau. Son outil de conscience de groupe utilise plusieurs modules graphiques comme modules de représentation. Il utilise notamment un module de visualisation en *treemap* (fig. 17). Dans cette visualisation, chaque membre du groupe est représenté par un rectangle, dont la couleur dépend de l'activité de la personne en question.

L'information indiquant le changement d'activité de Bruce est représentée par un changement de couleur du rectangle correspondant à Bruce. Cette représentation est très peu intrusive, et correspond à un niveau un (1).

L'information indiquant l'accessibilité de Clark sur le réseau est représentée par le clignotement d'une icône, pendant 10 secondes, dans le rectangle correspondant à Clark.

[8] *Overview first, zoom and filter, then details on demand.*

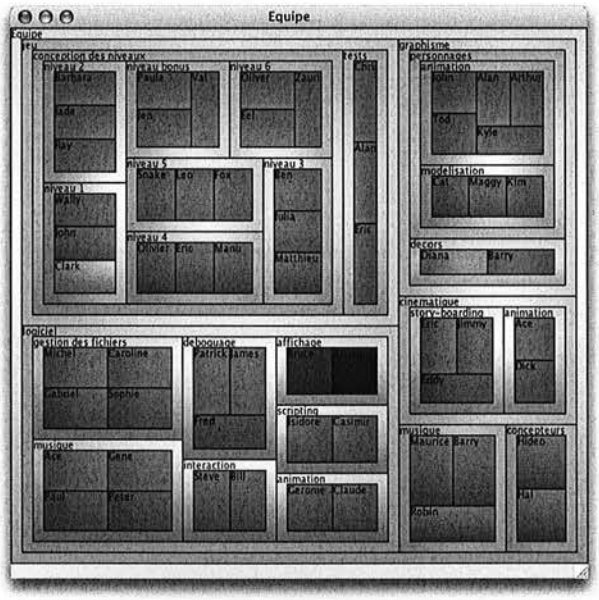

FIG. 17 – Le module de visualisation *treemap*.

L'apparition de l'icône est accompagnée d'un son doux. Cette représentation est une représentation normale, assez périphérique pour qu'Arthur puisse continuer à travailler mais assez intrusive pour qu'il la remarque du coin de l'œil. Elle correspond à un niveau deux (2).

L'information indiquant la modification du fichier par Clark est représentée par une ligne rouge dans la fenêtre des problèmes. Cette fenêtre, non bloquante, apparaît dans le coin de l'écran pour afficher les informations de niveau trois (3). La fenêtre reste dans le coin de l'écran jusqu'à ce qu'elle soit fermée par Arthur. L'apparition de la fenêtre ou d'une nouvelle ligne dans la fenêtre est accompagnée d'un son de cloche. Cette représentation est assez intrusive pour ne pas pouvoir être ratée. Elle correspond à un niveau trois (3).

Sur son *PDA*, Clark a choisi de n'utiliser qu'un unique module sonore. Chaque niveau d'information déclenche un signal sonore spécifique. Pour accéder au détail de l'information reçue, Clark utilise une interface graphique. L'utilisation d'un signal sonore pour la notification permet à Clark de garder tout son écran pour son activité principale.

Arthur peut aussi naviguer dans l'ensemble des informations qu'il a déjà reçues en utilisant le module de navigation *hypertree* (fig. 18). Les informations y sont classés sous la forme d'une hiérarchie, suivant les personnes émettrices et les catégories des informations. Il peut alors obtenir le détail d'une information spécifique.

3.8 Configuration de l'outil

Pour pouvoir s'adapter au contexte de travail de l'utilisateur, l'outil de conscience de groupe doit d'abord en posséder un modèle. Ce modèle est défini pendant la phase de configuration de l'outil, une phase préparatoire qui a lieu avant son installation et son

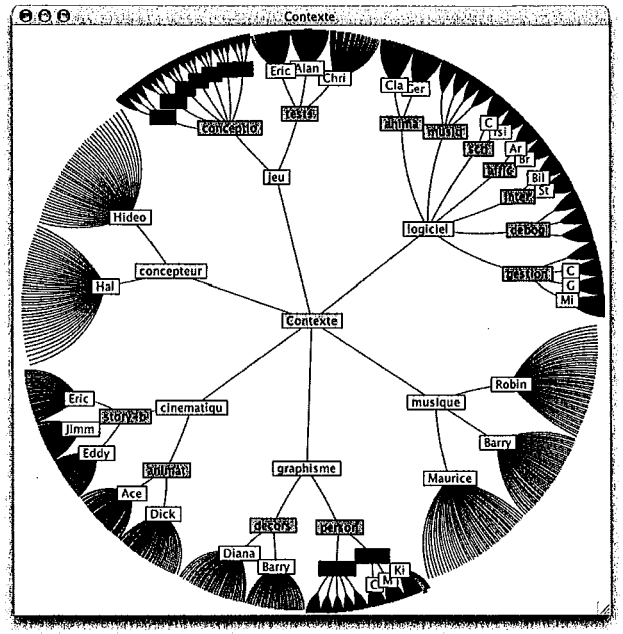

FIG. 18 – Le module de visualisation *hypertree*.

utilisation.

Le modèle d'un contexte de travail est défini par trois éléments : l'ensemble des informations de bas niveau pouvant être capturées, la structure des réseaux bayesiens et celle des réseaux de neurones. La spécification des informations de bas niveau définit les informations capturées par l'outil. La spécification des réseaux bayesiens définit les informations haut niveau pouvant être produites, ainsi que leurs relations avec les informations bas niveau. Enfin, la spécification des réseaux de neurones définit l'influence du contexte de travail en ce qui concerne la diffusion et la représentation des informations. La modélisation du contexte de travail d'une personne passe donc par la spécification de ces trois éléments : les informations bas niveau, les réseaux bayesiens et les réseaux de neurones. La spécification exacte de chacun de ces trois éléments est abordée plus en détail dans la partie correspondant à son utilisation.

Le modèle du contexte de travail d'une personne est spécifique au type d'activité de cette personne, ainsi qu'au groupe auquel elle appartient. Un programmeur n'a pas le même type d'activité qu'un graphiste, par exemple. De même, deux équipe de développement de jeux vidéo n'ont pas forcément la même structure. Dans ces deux cas, les modèles des contextes de travail résultants sont différents : le type des informations bas niveau et haut niveau, les relations entre ces informations, les adaptation possibles, tout cela est différent. Ainsi, le modèle d'un contexte de travail ne peut pas être générique. Il doit être créé spécifiquement pour chaque équipe et pour chaque type d'activité effectué dans l'équipe.

La phase de configuration a pour but de réaliser cette modélisation. Dans cette phase, les trois éléments modélisant le contexte de travail propre à un type d'activité sont spé-

cifiés. Cette phase a lieu avant que l'outil ne soit distribué et installé chez les membres du groupe. Il faut une phase de configuration par modèle, c'est-à-dire par type d'activité dans le groupe. Tous les programmeurs du département « logiciel » par exemple ont le même modèle. Par contre, dans le département « graphisme », les graphistes du sous-département « décors » n'utilisent pas les mêmes logiciels que ceux du sous-département « personnage - animation ». Leur modèle est donc différent, et demande une phase de configuration supplémentaire. Cette phase de configuration est la seule à être différente suivant les personnes. Par la suite, le fonctionnement de l'outil est identique, quel que soit le modèle.

4 État de l'art

Le travail de cette thèse, par son apport pluridisciplinaire, peut être mis en comparaison avec plusieurs domaines : le domaine du Travail Collaboratif Assisté par Ordinateur (*CSCW*) bien sûr, mais aussi celui de l'Informatique Pervasive (*Ubiquitous Computing*) et celui de l'Interaction Homme-Machine (*HCI*).

Dans le domaine du Travail Collaboratif Assisté par Ordinateur, il existe plusieurs travaux dont l'approche comporte des aspects similaires avec celle suivie par cette thèse, elle-même résumée dans un article de l'auteur [BC01].

Les travaux les plus proches sont ceux de Gross et Prinz [GP00, GP03]. Le but poursuivi est le même : proposer un outil de conscience de groupe capable de capturer et présenter les informations de conscience de groupe d'une manière adaptée au contexte. Leur proposition distingue deux types de contextes différents : le contexte d'origine de l'information, comparable au contexte de l'émetteur de l'information, et la situation de l'utilisateur, comparable au contexte du receveur. Le contexte est défini par un ensemble de couple attribut-valeur, définis dans l'étape de configuration de l'outil. Le contexte d'origine est attaché à un événement bas-niveau lors de l'émission de celui-ci. Le contexte de l'utilisateur est lui déduit de ses activités (la proposition ne donne pas de précision sur la réalisation de cette déduction). L'utilisateur précise dans ses préférences, en fonction de son contexte, les événements pour lesquels il veut être notifié, et comment (principalement avec quel outil). Lorsqu'un événement est reçu, les champs de son contexte sont comparés aux champs du contexte de l'utilisateur ainsi qu'à ses préférences, et l'événement est traité en conséquence.

L'approche proposée par Gross et Prinz est différente de celle de cette thèse à plusieurs égards. Tout d'abord, cette thèse ne propose pas seulement un outil, mais une architecture fonctionnelle permettant d'adapter un outil de conscience de groupe au contexte. Cette architecture fonctionnelle est basée sur un modèle cognitif de la conscience de groupe, déduit d'une analyse cognitive des mécanismes de construction de la conscience de groupe. De plus, dans notre approche, il n'y a qu'un seul contexte, unifié, celui de l'utilisateur, que celui-ci soit émetteur ou bien récepteur. Cela permet notamment de simplifier la spécification du contexte, mais aussi sa création. En effet, le contexte est construit directement à partir des événements envoyés aux membres du groupe, grâce à l'agrégation réalisée par les réseaux bayesiens. Il sert ensuite non seulement à sa propre mise à jour, mais aussi à adapter le filtrage et la représentation des informations. Enfin, leur proposition

permet une réduction de la surcharge d'information mais au prix d'une augmentation de la surcharge de travail, puisque l'utilisateur doit paramétrer ses préférences en fonctions de tous les contextes possibles. Cette surcharge de travail est fortement réduite dans notre approche grâce aux capacités d'apprentissage des réseaux neuronaux qui s'ajustent ainsi automatiquement aux préférences de l'utilisateur au fur et à mesure de l'utilisation de l'outil. Finalement, les niveaux de présentation proposés sont très spécifiques au cadre de l'espace de travail partagé.

Le travail de Rauschenbach [Rau96] propose, dans le cadre d'un mécanisme de conscience de groupe à base de notification d'événements dans un espace de travail partagé, de filtrer les événements en fonction de leur intérêt. L'intérêt d'un événement est modélisé par cinq champs, répondant à cinq questions : le type de l'événement (correspondant à *quoi*), l'instant de sa création (*quand*), les objets affectés dans le cadre de l'espace de travail partagé (*où*), la personne à l'origine de l'événement (*qui*), et enfin un champ contenant des paramètres optionnels. L'utilisateur spécifie, dans ses préférences, le niveau d'intérêt à apporter à un événement en fonction des valeurs des différents champs. Lorsqu'un événement arrive, ses champs sont comparés aux préférences de l'utilisateur, et un niveau d'intérêt en est déduit. La présentation de l'événement à l'utilisateur dépend de ce niveau d'intérêt. L'événement peut ainsi soit ne pas être présenté du tout, soit être présenté de différentes manières dans l'espace de travail partagé, soit enfin être présenté dans une fenêtre externe, modale ou non.

L'approche proposée par Rauschenbach souffre d'un problème principal : la surcharge de travail que doit supporter l'utilisateur pour configurer le système. En effet, pour que le système soit effectivement utile, l'utilisateur doit spécifier tous les niveaux d'intérêt en fonction d'une majorité des valeurs des champs possibles. De plus, les événements envoyés ne sont que des événements bas niveau, ce qui augmente le nombre d'informations envoyées. Enfin, le filtrage dépend uniquement du contexte de l'événement, pas du contexte de l'activité de l'utilisateur. Finalement, les niveaux de présentations proposés sont très spécifiques au cadre de l'espace de travail partagé.

L'approche de Schwabe [Sch96] est assez similaire à celle de Rauschenbach. Elle concerne aussi la conscience de groupe dans le cadre d'un espace de travail partagé. La proposition aborde le problème du passage à l'échelle dans le support de la collaboration d'un grand groupe, contenant de nombreux membres et pouvant manipuler un grand nombre d'artefacts. La proposition de Schwabe est de collecter des informations sur le contexte de la collaboration. Ces informations sont regroupées en cinq catégories, répondant à cinq questions : qui travaille sur un artefact particulier, où sont situés les participants, quand une action a-t-elle eu lieu, qu'a fait une personne spécifique (*quoi*), comment le changement a-t-il eu lieu. Ces informations bas niveau sont collectées à travers l'espace de travail partagé et mis dans une base de données, interrogeable par formulaire.

Cette approche se limite à la collecte d'information bas niveau, et n'aborde ni l'agrégation d'informations, ni leur filtrage, ni enfin leur présentation. C'est à l'utilisateur d'aller chercher les informations dont il a besoin ; il n'est pas notifié en cas de changement.

Enfin, le travail de Chen et Gellersen [CG99] propose une utilisation originale du contexte pour la conscience de groupe. Il distingue deux types de contextes : le contexte « boîte noire », interprétable uniquement par un humain, et directement transmis vers l'utilisateur (comme une image ou une vidéo), et le contexte « boîte blanche », interpré-

table par le système et utilisé pour ses déductions (comme une fréquence de frappe clavier). Le système proposé collecte des informations de différents capteurs, principalement vidéo et audio. Les informations provenant du contexte « boîte blanche » sont utilisées pour modifier la présentation des informations provenant du contexte « boîte noire ». Le résultat est une présentation des informations multimédia de conscience de groupe sous la forme d'un *story-board*.

Les résultats de Chen et Gellersen, tant au niveau de l'utilisation du contexte qu'au niveau de la présentation des informations, sont fortement spécifiques à la nature multimédia des informations collectées. Le contexte utilisé n'est pas modélisé, pas plus que son utilisation. Enfin, les différentes déductions sont spécifiques à l'exemple étudié. Notre proposition, par son architecture fonctionnelle et son modèle cognitif, est plus générale.

L'approche de cette thèse peut aussi être comparée à celle utilisée dans le domaine de l'Informatique Pervasive (*Ubiquitous Computing*), et plus particulièrement des applications adaptées au contexte (*Context-Aware Application*). Le cadre général de l'Informatique Pervasive est celui des systèmes d'information embarqués dans les objets courants ou mobiles, comme les *PDA* (*Portable Digital Assistant*).

Un bon exemple d'application adaptée au contexte dans ce domaine est fourni par les travaux de Sawhney et Schmandt [SS99, SS00]. Il s'agit d'un système de notification auditive adaptée au contexte de la situation de l'utilisateur pour un système de messages auditifs *wearable*, c'est-à-dire embarqué dans les vêtements. Dans ce cadre, le contexte de la situation de l'utilisateur est déduit d'un ensemble de capteurs, principalement audio. Lorsqu'un nouveau message arrive, ce contexte sert à choisir un moment et une représentation auditive adéquate pour en notifier l'utilisateur.

Les travaux de Abowd, Dey et Salber [DA01, SDA99] expriment bien le but poursuivi dans ce domaine : améliorer le comportement des applications en les informant de leur contexte d'utilisation. Pour cela, ils définissent le contexte comme étant :

> « toute information pouvant être utilisée pour caractériser la situation d'entités (que ce soit une personne, un endroit ou un objet) qui sont considérées comme pertinentes pour l'interaction entre l'utilisateur et l'application, ceci incluant l'utilisateur et l'application eux-mêmes. Le contexte est typiquement la localisation, l'identité et l'état des personnes et des groupes, et les objets concrets et abstraits. » [9]

Leur travaux proposent notamment un cadre conceptuel d'utilisation du contexte, contenant quatre étapes fonctionnelles : la collecte d'informations bas niveau, l'interprétation en informations haut niveau, l'agrégation d'informations haut niveau, et leur représentation.

Les approches utilisées dans le domaine de l'Informatique Pervasive sont similaires à celles de cette thèse, mais le but n'est pas le même. Comme exprimé plus haut, le but de l'Informatique Pervasive est d'incorporer les systèmes d'informations dans tous nos objets quotidiens, dans le but d'assister l'utilisateur. Dans ce cadre, il est clair que les applications tournant sur ces systèmes embarqués doivent se mettre à la portée de l'utilisateur et l'aider

[9] any information that can be used to characterize the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves. Context is typically the location, identity and state of people, groups and computational and physical objects.

dans ses tâches, et non pas imposer leur fonctionnement à l'utilisateur. L'adaptation de ces applications au contexte de la situation de l'utilisateur est donc un point capital. Le but de la proposition de cette thèse, lui, est spécifique à la réduction de la surcharge d'informations dans le cadre de la conscience de groupe. Le but étant plus spécifique, les solutions aussi, comme par exemple le modèle cognitif de la conscience de groupe ou bien l'architecture fonctionnelle.

Enfin, l'approche de cette thèse peut être comparée à une approche similaire récente dans le domaine de l'Interaction Homme-Machine, celle des Interfaces Utilisateur Attentives (*Attentive User Interface*). L'article de Vertegaal [Ver03] explique cette nouvelle approche en partant du constat suivant : les systèmes d'information se multiplient dans le champ de travail de l'utilisateur. Et avec le courant de l'Informatique Pervasive, les systèmes d'information vont bientôt envahir notre vie quotidienne. Or, tous ces systèmes, prévus pour assister l'utilisateur, demandent tous la même ressource : l'attention de l'utilisateur. Cette ressource, limitée, devient donc une ressource importante à préserver. Le but des Interfaces Utilisateur Attentives est de minimiser l'utilisation de cette ressource, l'attention de l'utilisateur, en mesurant l'importance de leur information par rapport aux priorités de l'activité de l'utilisateur.

Un exemple dans ce domaine est fourni par les travaux de Shell, Selker et Vertegaal [SSV03], qui décrivent différents systèmes capables de filtrer l'information à fournir en fonction de l'activité détectée de l'utilisateur. Ces systèmes utilisent principalement des capteurs extérieurs spécifiques, comme par exemple des capteurs permettant de déterminer où se porte le regard de l'utilisateur.

Un autre exemple est celui décrit par Horvitz, Kadie, Paek et Hovel [Hor03], qui utilisent des réseaux bayesiens pour modéliser l'activité de l'utilisateur à partir d'informations bas niveau. Cette modélisation est utilisée pour estimer le coût d'un délai dans la fourniture d'une information par rapport au coût de l'interruption de l'utilisateur.

La proposition de cette thèse peut être vue comme l'utilisation de l'approche des Interfaces Utilisateur Attentives au problème de la conscience de groupe.

Visualisation

Le chapitre précédent a décrit l'architecture générale d'un outil de conscience de groupe capable de s'adapter au contexte de travail de l'utilisateur. Dans la dernière étape de cette architecture, l'étape de présentation, les informations de conscience de groupe sont présentées à l'utilisateur à l'aide d'un ou plusieurs types de visualisation, chacun réalisé par un module spécifique. Déléguer la visualisation à un module spécifique permet à l'utilisateur de choisir la visualisation qu'il préfère, simplement en changeant de module.

Il existe un grand nombre de techniques de visualisation, développées dans le cadre du domaine de recherche de la Visualisation d'Informations (*Information Visualization*). L'analyse cognitive de la conscience de groupe a permis de définir un certain nombre de propriétés que doivent avoir les visualisations utilisées par l'outil. Certaines visualisations sont donc plus adaptées que d'autres pour représenter les informations de conscience de groupe.

La deuxième proposition de cette thèse concerne le choix de deux techniques de visualisations particulièrement adaptées pour représenter les informations de conscience de groupe : les visualisations *treemap* et *hypertree*. La suite de ce chapitre explique ce choix, ainsi que la réalisation et l'utilisation de ces deux techniques de visualisation.

1 Visualisation de la conscience de groupe

L'architecture de l'outil de conscience de groupe proposée par cette thèse découple la collecte, la diffusion et la présentation des informations de conscience de groupe de leur représentation sur le système de l'utilisateur. L'utilisateur peut ainsi choisir, en fonction de ses préférences et des capacités techniques de son système, la technique utilisée pour représenter les informations.

Pour un système standard, n'ayant pas de matériel spécifique, cette représentation est généralement visuelle ou auditive (ou une combinaison des deux). Mais alors qu'il existe beaucoup de travaux sur les représentations visuelles [Spe01, War00], il en existe très peu sur les représentations auditives [CBL96, MBW+98]. Cette thèse s'intéresse donc plus particulièrement aux techniques de représentation visuelle.

L'utilisateur a le choix entre un grand nombre de techniques de visualisations possibles. Chaque technique de visualisation a ses propres caractéristiques, ses points forts et ses points faibles. Certaines techniques sont plus adaptées que d'autres pour représenter les informations de conscience de groupe, dans le cadre défini par cette thèse. Pour déterminer lesquelles, il faut pouvoir classifier l'ensemble des visualisations en fonction de leurs fonctionnalités.

1.1 Cadre général de la visualisation d'informations

Les grands ensembles de données sont présents dans un grand nombre de situations. Un disque dur normal de nos jours peut facilement contenir des centaines de milliers de fichiers. Un système de gestion de connaissances peut contenir des milliers d'éléments. Un moteur de recherche sur le web peut renvoyer des centaines de pages correspondant à une requête. Dans chaque cas, ce n'est pas l'ensemble des données qui est véritablement intéressant. La vraie valeur ajoutée est enfouie dans l'ensemble lui-même, dans les relations et les motifs qui peuvent y être trouvés, ou bien dans certaines données spécifiques qui doivent être recherchées et extraites de l'ensemble.

Face à ces données, l'utilisateur peut vouloir accomplir un grand nombre de tâches différentes : rechercher les relations existantes entre ces données, rechercher des motifs cachés, naviguer dans les données à la recherche d'une information spécifique, avoir une idée de la structure générale des données...

Le but du domaine de recherche de la Visualisation d'Informations (*Information Visualization*) est de fournir les outils nécessaires aux utilisateurs pour les aider à réaliser ces tâches en visualisant les données [BS03, CMS99, Spe01, War00]. Ce domaine combine des aspects de l'Interaction Homme-Machine [DFAB98, PRS+94, Shn97], du graphisme statistique [Ber98, Tuf90, Tuf97, Tuf01] et du design [DK01a, DK01b].

Il faut le différencier du domaine de la Visualisation Scientifique, dont le but est de représenter des données provenant de domaines scientifiques, comme par exemple des données de températures sur un fluide en écoulement dans un tube. La problématique de la Visualisation Scientifique est de représenter les données d'une manière compréhensible pour le scientifique, ce qui revient à présenter les données suivant la représentation physique modélisée. La problématique de la Visualisation d'Information est de trouver les représentations abstraites adéquates pour représenter à l'utilisateur des informations dont la correspondance physique n'a pas ou peu de sens. Ainsi, l'image d'un billet de banque aura peu d'importance pour représenter la fluctuation d'un ensemble de cours monétaires.

Les recherches dans le domaine de la Visualisation d'Information concernent la mise au point de théories et de cadres de travail permettant de comprendre les mécanismes cognitifs et perceptifs de l'être humain, la création de nouvelles techniques de visualisation tirant partie de ces théories et cadres de travail, ainsi que la mesure de l'efficacité de ces nouvelles visualisations. Deux critères sont importants à distinguer dans l'évaluation des techniques de visualisation d'informations : le type et la dimension des données à visualiser d'une part, et la tâche à accomplir d'autre part. C'est la Taxonomie des Types de Données par Tâche (*Data Type by Task Taxonomy*) [Shn97]. Cette taxonomie permet de classer les différentes techniques de visualisation.

Le premier critère de classification de cette taxonomie est le type et la dimension des données à visualiser. Les différentes catégories de données sont les suivantes :

- les données linéaires ;
- les données bi-dimensionnelles ;
- les données tri-dimensionnelles ;
- les données multi-dimensionnelles ;
- les données temporelles ;
- les données sous forme d'arbre ;

– les données sous forme de réseau.

Ainsi, les graphiques simples, les listes ainsi que les visualisations offrant la possibilité de se situer à l'intérieur d'un texte permettent de visualiser les données à une dimension (linéaires). Les graphiques statistiques, en ligne ou en barre, ainsi que les cartes sont des exemples de visualisation de données bi-dimensionnelles, alors que les environnements virtuels sont typiquement des visualisations de données tri-dimensionnelles. Les techniques de visualisation de données temporelles mettent l'accent sur les propriétés particulières de la dimension temporelle : c'est par exemple le cas des visualisations d'agenda ou de planification de tâches.

Les techniques de visualisation de données en arbre ou en réseau sont spécialisées dans la représentation non seulement des données mais aussi des relations entre ces données. Une visualisation permettant une représentation de données sous forme de réseau peut bien sûr visualiser aussi des données sous forme d'arbre, mais l'inverse n'est pas vrai. Et les visualisations spécialisées pour les arbres tirent parti de la structure simple d'un arbre par rapport à un réseau pour apporter un plus. C'est par exemple le cas des deux visualisations qui nous intéressent, la visualisation *treemap* et la visualisation *hypertree*.

Le deuxième critère de classification de la taxonomie concerne le type de la tâche que permet d'accomplir la visualisation. Les différentes catégories de tâches possibles sont les suivantes :

– avoir une vue globale ;
– zoomer ;
– filtrer ;
– avoir une vue en détail ;
– voir les relations ;
– extraire.

Une vue globale permet de se faire une idée de l'ensemble des données, c'est-à-dire de la structure de l'ensemble ainsi que du type des données contenues dans l'ensemble. Un zoom permet d'accéder, dans le contexte, à une donnée spécifique. Un filtre permet d'éliminer de la vue certaines données pour ne conserver que celles répondant à un ou plusieurs critères. Une vue en détail permet d'accéder à tous les détails d'une donnée spécifique. Voir les relations permet de se concentrer sur les relations entre les données et non pas sur les données elles-mêmes. Enfin, extraire permet de construire un nouvel ensemble de données à partir d'un sous-groupe de l'ensemble principal, et d'agir sur ce nouvel ensemble.

De nombreuses techniques de visualisation ont été proposées pour aider à gérer de grands ensembles de données. Chaque technique est spécialisée dans la visualisation d'une catégorie de données et dans la réalisation d'une tâche spécifique. Mais il n'existe pas de technique de visualisation qui serait efficace pour toutes les données et dans toutes les tâches. Une technique de visualisation ne pouvant être adaptée pour toutes les tâches, chaque visualisation possède sa spécificité. Il faut donc connaître le type des données à visualiser et les tâches que l'utilisateur va devoir accomplir afin de choisir la ou les techniques de visualisation appropriées.

1.2 Choix des visualisations pour la conscience de groupe

Pour choisir une technique de visualisation adaptée à la représentation d'informations de conscience de groupe, il faut déterminer le type de tâches que va accomplir l'utilisateur ainsi que le type des données à représenter.

Concernant les types de tâches à réaliser, l'analyse cognitive de la conscience de groupe a permis de spécifier un certain nombre de prérequis pour l'outil de conscience de groupe. En particulier, concernant la présentation des informations, l'analyse a permis de déterminer deux modes de représentation des informations de conscience de groupe :

- un mode périphérique, pour présenter les nouvelles informations d'une manière globale ;
- un mode actif, permettant à l'utilisateur d'accéder de manière simple mais détaillée à une information spécifique.

Ces deux modes correspondent à des tâches utilisateur différentes.

Dans le premier mode, la tâche principale de l'utilisateur est la vue globale, qui lui permet de se rendre compte de l'apparition de nouvelles informations et d'identifier les informations changeantes. Les tâches de zoom et de filtrage sont aussi importantes, lorsque l'utilisateur ne s'intéresse qu'à une partie des informations (par exemple à un groupe particulier de collaborateurs).

Dans le second mode, l'interface de l'outil doit permettre à l'utilisateur d'accéder à l'ensemble des informations reçues. Cet accès à l'ensemble des informations se fait suivant trois vues : une vue globale, une vue de navigation et une vue en détail. La vue globale sur l'ensemble des informations permet d'obtenir une vision générale de la structure et de la répartition des informations, de faire des comparaisons globales entre les différentes informations, et de repérer des motifs ou des éléments particuliers. À partir de cette vue, l'utilisateur peut définir des sous-ensembles d'information. Il peut ensuite invoquer sur ces sous-ensembles la vue de navigation. La vue de navigation permet de naviguer dans un ensemble (potentiellement très grand) d'informations et de trouver une information spécifique. L'utilisateur peut obtenir sur cette information une vue en détail pour accéder à la liste des détails composant l'information. La combinaison de ces trois vues respecte la désormais célèbre formule de Shneiderman [Shn97] :

> « D'abord la vue globale, zoom et filtrage, puis les détails à la demande. »[10]

Dans le cadre de l'outil proposé par cette thèse, les informations de conscience de groupe sont de plusieurs types, comme vue dans le chapitre précédent. Tout d'abord, les informations reçues sont des informations de haut niveau, composés de six catégories. Ce sont donc des informations multi-dimensionnelles. Ensuite, chaque catégorie a sa propre spécificité : il y a des informations unidimensionnelles, comme le type de l'activité (*quoi*) ; des informations multi-dimensionnelles, comme les objets et outils utilisés (*comment*), des informations temporelles (*quand*), etc. Enfin, chaque tâche a besoin plus particulièrement de certaines catégories d'information. Le choix des catégories à représenter suivant la tâche à accomplir détermine donc le type des données à représenter. Pour simplifier la représentation, les informations, quelles que soient leurs catégories, sont classées sous

[10] *Overview first, zoom and filter, then details on demand.*

forme d'arbre. Ainsi, une technique de visualisation adaptée sera une visualisation pouvant représenter des données arborescentes.

Dans le cadre de cette thèse nous avons choisi de porter notre attention sur deux techniques de visualisation particulières : la visualisation *treemap*, et la visualisation *hypertree*. Ces deux visualisations ont en commun le fait d'être dédiées à la représentation de données arborescentes, ce qui correspond au type des données à représenter. De plus, chacune est spécialisée dans une tâche spécifique : la visualisation *treemap* est excellente pour avoir une vue globale, et la visualisation *hypertree* est excellente pour la navigation. Ces deux visualisations répondent donc parfaitement aux critères des tâches à effectuer pour chacun des modes. La visualisation *treemap* est utilisée pour le premier mode, et la visualisation *hypertree* pour le deuxième mode. La suite du chapitre décrit tour à tour ces deux représentations, puis détaille leur utilisation combinée pour la représentation des informations de conscience de groupe.

2 Treemap

2.1 Présentation

La technique de visualisation *treemap* a été inventée par Ben Shneiderman, à l'université du Maryland, dans le Laboratoire d'Interfaces Homme-Machine (*HCIL*) en 1991 [JS91, Shn92]. La visualisation *treemap* représente une hiérarchie de données arborescentes comme un ensemble de rectangles emboîtés (cf. fig. 1). Chaque nœud de l'arbre est représenté comme un rectangle, dont la taille et le contenu dépendent directement des propriétés du nœud. Par exemple, pour la visualisation d'un système de fichiers, la taille du rectangle peut dépendre de la taille des fichiers représentés, et la couleur de remplissage de la date de dernière modification. L'information concernant la position du nœud dans la hiérarchie est donnée par la façon dont les rectangles sont imbriqués.

La visualisation *treemap* est construite par découpage récursif du rectangle de la vue. L'algorithme de découpage original divise chaque rectangle en sous-rectangles, proportionnellement aux tailles des nœuds, suivant une unique direction, horizontale ou verticale. La direction change à chaque changement de niveau dans la hiérarchie (cf. fig. 2).

Avec cet algorithme, seuls les nœuds feuilles sont représentés. Les nœuds conteneurs ne sont visibles que par la somme des rectangles de leurs nœuds fils. Pour rendre plus visibles les nœuds conteneurs, il est possible de leur associer un bord : il s'agit alors de *nested treemap* (cf. fig. 3).

Le principal avantage de la visualisation *treemap* est son efficacité quant à l'utilisation de l'espace. Puisqu'elle représente une hiérarchie entière dans un rectangle, elle permet d'avoir une vue globale de cette hiérarchie, même si cette dernière est très grande. Cette vue globale permet à l'utilisateur de comparer les nœuds, en se basant sur les caractéristiques de leurs rectangles respectifs : leurs tailles relatives et leur couleur. Ces deux caractéristiques peuvent être associées à n'importe quelle caractéristique des nœuds correspondants, à une restriction près : la caractéristique « taille » doit être cumulative, c'est-à-dire que la taille d'un nœud qui contient d'autres nœuds doit être au moins égale à la somme des tailles de ses enfants.

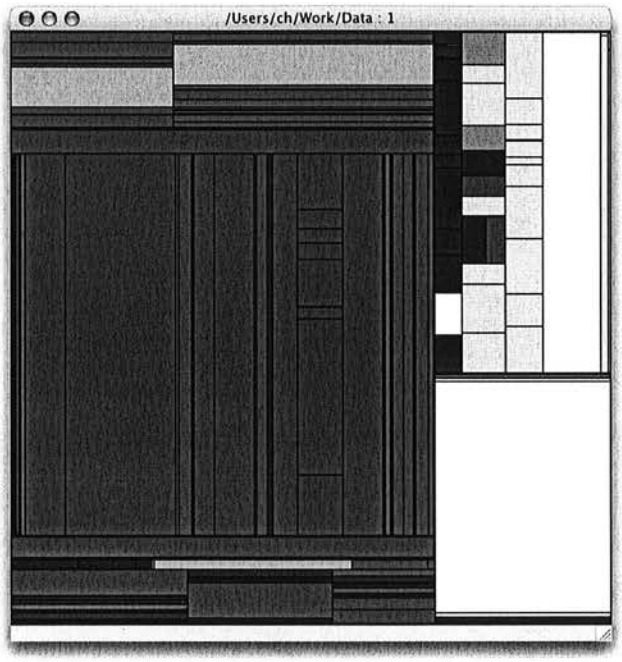

FIG. 1 – La visualisation *treemap*.

La visualisation *treemap* donne une vue globale sur un ensemble de données. Cette vue globale peut être décrite comme une « vue gestaltique » [Koh47] de l'ensemble de données, c'est-à-dire une vue qui a des propriétés que la somme des vues locales n'a pas. Cette vue gestaltique des caractéristiques de la hiérarchie complète peut être utilisée comme une carte (« map » en anglais, d'ou le nom de la visualisation) pour rechercher un nœud spécifique en fonction de ses caractéristiques, ou bien pour avoir une idée de la structure des données. Elle permet d'observer deux propriétés de l'ensemble : la relative prédominance d'une donnée par rapport aux autres, et la relative prédominance d'un critère par rapport aux autres. La détection de la prédominance d'une donnée se fait en repérant le plus grand rectangle dans la vue *treemap*. La prédominance d'un critère, représentée par la couleur des nœuds, est donnée par la couleur globale de la vue *treemap*. La répartition des différentes couleurs est aussi une information pertinente.

Par exemple, la figure 1 est une visualisation *treemap* d'un système de fichiers. Chaque rectangle — chaque nœud du *treemap* — représente un fichier. La couleur est associée à la date de dernière modification du fichier : blanc pour les plus récents fichiers, puis vert, jaune, orange, rouge, et finalement bleu pour les plus vieux. La taille est naturellement associée à la taille du fichier. Dans la vue *treemap*, il est facile de trouver le plus gros des fichiers de l'ensemble (le grand blanc en bas à droite). Il est assez récent (car de couleur blanche). Il est aussi facile de voir qu'une partie du disque à été remplie récemment, car les rectangles jaunes et blancs prennent une partie de la place à droite de la vue.

Plusieurs mécanismes existent pour apporter des capacités de navigation à la visualisation *treemap*. Il est par exemple possible de modifier dynamiquement les caractéristiques

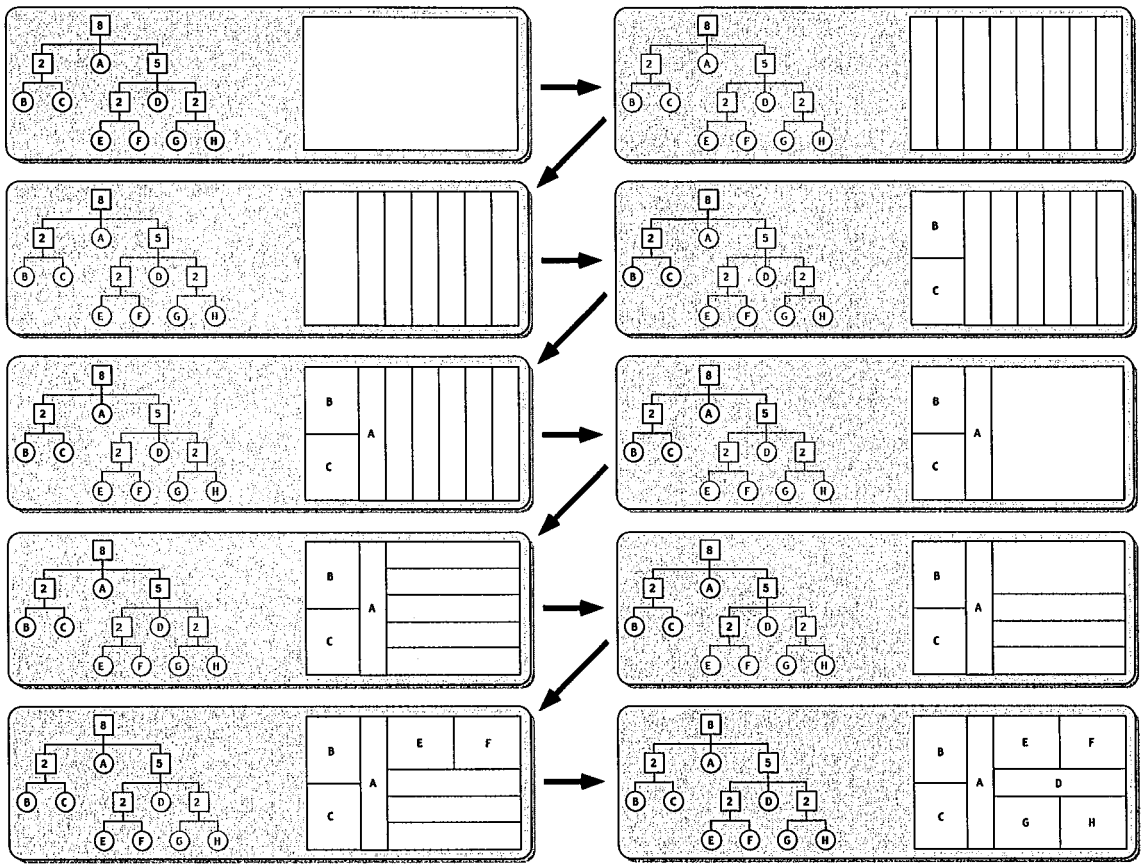

FIG. 2 – Construction d'un *treemap*.

représentées dans le *treemap*. Ainsi, l'association de la taille des rectangles et de leur couleur à certaines caractéristiques des données peut être changée dynamiquement, permettant ainsi une comparaison entre plusieurs critères. Il est aussi possible d'appliquer un filtre dynamique sur les caractéristiques pour que le *treemap* ne représente que les nœuds pertinents. Finalement, une fonction de zoom peut être implantée de façon à ce qu'un clic sur un nœud zoome le *treemap* à un niveau plus bas dans la hiérarchie. Il faut cependant reconnaître que la force du *treemap* n'est pas dans la navigation, mais dans la vue globale de la hiérarchie.

Le principal désavantage de la technique de visualisation *treemap* est qu'il ne s'agit pas d'une visualisation intuitive. Il faut environ de dix à quinze minutes pour une personne qui n'a jamais vu la visualisation *treemap* pour apprendre comment la lire. Mais une fois que l'apprentissage a été fait, la visualisation *treemap* est généralement trouvée utile et facile à comprendre. Un autre désavantage est que la visualisation *treemap* ne donne qu'une idée imprécise de la hiérarchie : il est quasiment impossible, rien qu'en regardant un nœud du *treemap*, de déterminer à quel niveau de la hiérarchie il se trouve. Il faut noter que la vue *nested treemap* améliore ce point. Enfin, la technique *treemap* est strictement limitée à la visualisation de données arborescentes, sans liens supplémentaires entre les nœuds. Il

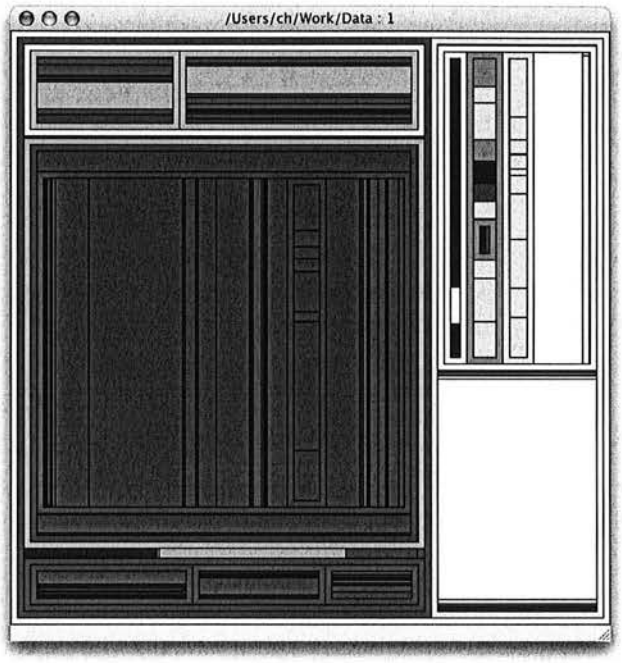

FIG. 3 – Exemple d'un *nested treemap* : un bord rend visible les nœuds conteneurs.

n'est pas possible de représenter un graphe, à moins de supprimer ou dupliquer certaines informations pour en faire un arbre.

2.2 État de l'art

Depuis sa première publication en 1991, beaucoup de travaux ont été publiés sur la visualisation *treemap*.

Certains décrivent des améliorations, comme les *cushions treemaps* de van Wijk et van de Wetering [vWvdW99]. Dans les *cushions treemaps*, les rectangles sont dessinés sous la forme de paraboloïdes, comme des coussins (d'où le nom). L'illumination de ces paraboloïdes donne un effet 3D qui donne une meilleure idée de la profondeur et de la différence entre les nœuds, ce qui permet de mieux voir la hiérarchie. Un grand nombre d'améliorations concerne l'algorithme de découpage. En effet, le principal défaut de l'algorithme original (*slide-and-dice*) est qu'il arrive fréquemment qu'une partie des nœuds soient très fins et très allongés, presque comme une ligne, ce qui les rend difficiles à voir et à distinguer. D'autres algorithmes ont été proposés pour résoudre ce problème, comme l'algorithme des *squarified treemap* de Bruls, Huizing, et van Wijk [BHvW00]. Cet algorithme de découpage a pour but d'essayer de toujours garder le rapport hauteur sur largeur des rectangles aussi près de 1 que possible, ce qui rend les rectangles plus « carrés », d'ou le nom de l'algorithme. L'algorithme de Vernier et Nigay [VN00] ressemble à l'algorithme précédent, à la différence qu'il permet d'assurer un rapport hauteur/largeur quelconque, fixé par l'utilisateur. L'avantage de ces algorithmes est qu'ils permettent d'avoir des nœuds

plus faciles à lire. Leur désavantage est qu'ils rendent l'aspect du *treemap* très sensible au moindre changement d'un seul nœud. Si la taille d'un rectangle change, c'est quasiment tous les rectangles du *treemap* qui changent de place. D'autres algorithmes de découpages, assurant un minimum de changement de place ou d'ordre, ont été développés au *HCIL* [BSW02]. Un algorithme de découpage adapté à des nœuds ayant une taille fixe a aussi été développé au même endroit, dans le cadre d'un visualisateur-organisateur de photos [Bed01, BSW02].

D'autres travaux portent sur les nouveaux domaines d'applications de la visualisation *treemap*. La visualisation *treemap* a ainsi été utilisée pour représenter les variations d'un portefeuille d'actions à la bourse [JT92], pour un outil de gestion de configuration de réseau satellitaire [KPTS94], pour un outil d'aide à la décision [ATS95], pour représenter l'ensemble des forums de discussion de Usenet [FS01], pour représenter des informations de divergences entre fichiers partagés dans le cadre d'une collaboration multi-synchrone [MSMB01], pour visualiser des données médicales [Bou02], pour analyser la structure de code source [RSB02], et enfin pour visualiser des données d'exécutions d'un programme [OJH03].

Enfin, certains travaux décrivent des comparaisons entre la visualisation *treemap* et d'autres visualisations ayant le même but. Une de ces visualisations est la visualisation *Sunburst*, de Stasko et al. [SZ00, SCGM00].

Signe de leur développement et de leur intérêt, la visualisation *treemap* a fait l'objet d'un workshop au *HCIL* en 2000 [Shn00].

2.3 Contribution

La visualisation *treemap* permet de représenter des données arborescentes et de donner une vue globale de l'ensemble des données. Elle est donc tout a fait adaptée pour la représentation périphérique des données de conscience de groupe.

Malheureusement, au début de cette thèse, il n'existait aucune implantation libre de cette visualisation. Une des contributions de cette thèse est donc la réalisation d'une implantation libre de la visualisation *treemap* : la Treemap Java Library[11] [Bou01]. Il s'agit, comme son nom l'indique, d'une bibliothèque écrite en Java. En tant que bibliothèque, son cadre d'utilisation est générique : elle peut être utilisée par n'importe quelle application pour représenter n'importe quel type de données arborescentes. Elle a par exemple déjà été utilisée pour visualiser des données d'exécutions d'un programme [OJH03], pour visualiser des données médicales [Bou02], ou encore pour représenter des informations de divergences entre fichiers partagés dans le cadre d'une collaboration multi-synchrone [MSMB01]. De plus, elle est distribuée sous une licence libre et open-source, ce qui la rend utilisable par tous et dans n'importe quelle application, sans aucune contrainte. Enfin, son code source étant accessible et modifiable par tous, elle peut être adaptée efficacement aux besoins de n'importe qui.

En ce qui concerne ses fonctionnalités, la TreeMap Java Library propose à l'utilisateur plusieurs choix d'algorithmes et de visualisations. Ainsi, différents algorithmes de découpages peuvent être utilisés, comme l'algorithme original (*slice and dice*) (cf. fig. 1)

[11] `http://treemap.sf.net/`

ou bien l'algorithme *squarified* (cf. fig. 4). L'utilisateur peut passer dynamiquement d'un algorithme à l'autre. L'utilisateur a aussi le choix entre une visualisation normale ou une visualisation *cushion*, dont l'illumination est paramétrable (cf. fig. 5). Finalement, il est possible de transformer toute visualisation en *nested treemap* par l'ajout d'un bord réglable dynamiquement (cf. fig. 6). La modification dynamique de ce bord aide à la vision de la hiérarchie.

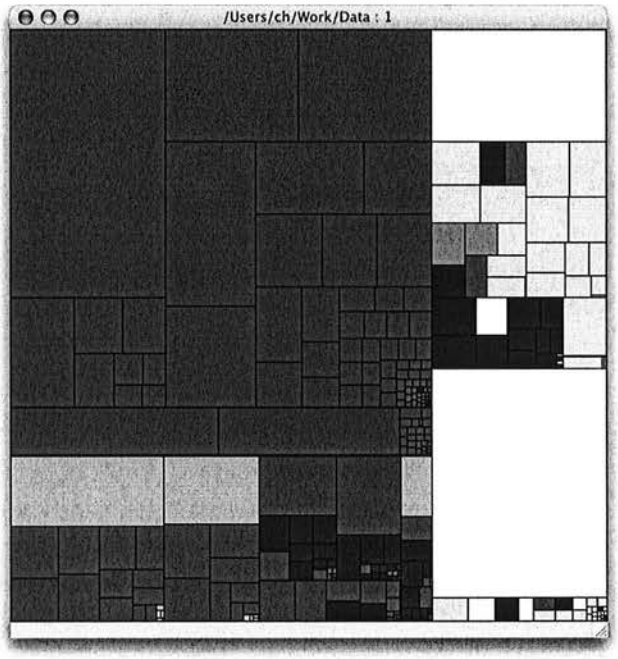

FIG. 4 – Un *treemap* utilisant l'algorithme *squarified.*

En ce qui concerne les interactions, la TreeMap Java Library propose un zoom permettant de descendre dans la hiérarchie : lorsque l'utilisateur sélectionne un rectangle, ce dernier devient la nouvelle racine de l'arbre affiché. De plus, les caractéristiques représentées dans le *treemap* par la taille ou la couleur des rectangles peuvent être changées dynamiquement. Finalement, les noms des nœuds peuvent être affichés ou masqués de manière dynamique.

Depuis l'implantation et la publication sur le web de la TreeMap Java Library, un certain nombre d'autres implantations de la visualisation *treemap* ont vu le jour sur Internet, comme l'*InfoVis Toolkit*[12], *Sequoia View*[13], ou *QTreeMap*[14]. L'implantation originale de Ben Shneiderman est même maintenant disponible[15] pour une utilisation non commerciale. Ces implantations sont pour la majorité réalisées dans le cadre spécifique d'un outil et ne sont pas réutilisables. À ce jour, la TreeMap Java Library reste la seule im-

[12] `http://ivtk.sf.net/`
[13] `http://www.win.tue.nl/sequoiaview/`
[14] `http://qtreemap.sf.net/`
[15] `http://www.cs.umd.edu/hcil/treemap/index.shtml`

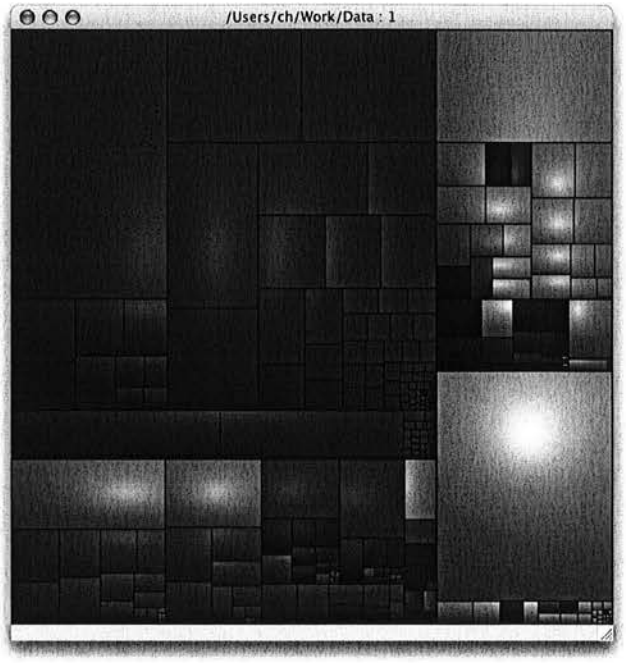

FIG. 5 – Un *treemap* avec la visualisation *cushion*.

plantation de la visualisation *treemap* qui soit libre, générique, et qui fournisse autant de fonctionnalités.

3 Hypertree

3.1 Présentation

La technique de visualisation en arbre hyperbolique, aussi appelée visualisation *hypertree*, a été inventée par Lamping et Rao [LR94, LR96a, LR96b, LRP95], au Palo Alto Research Center de Xerox (le fameux Xerox PARC) en 1994 (cf. fig. 7). L'histoire dit que l'idée leur est venue en regardant la gravure d'Esher intitulée « Circle Limit IV (Heaven and Hell) » de 1960, qui utilise les propriétés de la géométrie hyperbolique pour obtenir un effet de loupe (dit effet *fisheye*). Le but des arbres hyperboliques est de représenter et de manipuler de larges structures hiérarchiques dans l'espace hyperbolique pour profiter du même effet de loupe.

La géométrie hyperbolique et ses propriétés sont directement liées à la notion de parallélisme. Celle-ci repose sur le 5^{e} postulat d'Euclide[16]. C'est en cherchant à prouver ce dernier que Bolyai [Bol32] et Lobatchewsky [Lob37] mettent à jour une autre géométrie (cohérente, au sens du programme d'Erlangen [Hen01]). En supposant que l'on peut tirer une infinité de droites parallèles à une droite donnée passant par un point hors de cette

[16] « Par un point en dehors d'une droite donnée, il passe une et une seule droite parallèle à la droite donnée [Rat94]. »

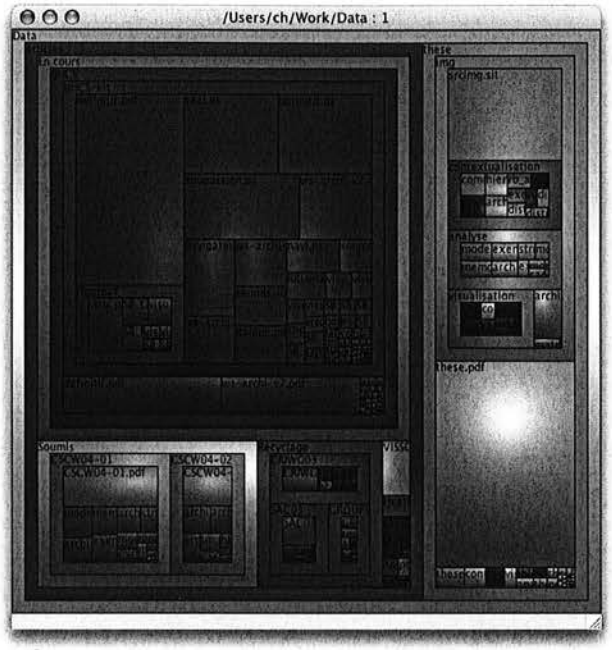

FIG. 6 – Un *nested cushion treemap*, avec affichage des noms des fichiers.

droite, ils révèlent la géométrie hyperbolique. En supposant que l'on ne peut tirer aucune parallèle, Riemann découvre la géométrie elliptique [Rie67]. Une représentation intuitive de la géométrie elliptique en dimension 2 dans l'espace euclidien est donnée par la surface de la sphère. Poincaré propose au 19^{e} siècle une représentation de l'espace hyperbolique (de dimension 2) dans l'espace euclidien sous la forme du disque unité [Hen01, Rat94].

La visualisation *hypertree* permet de représenter des données structurées sous forme d'arbre. L'arbre est dessiné de manière radiale — la racine au centre, les nœuds autour, les fils sur un cercle ayant pour centre leur père — dans le plan hyperbolique. Puis, une fois chacun des nœuds de l'arbre placé dans l'espace hyperbolique, l'arbre tout entier est projeté dans le disque de Poincaré. La racine est projetée au centre du disque unité. Les branches de l'arbre deviennent des géodésiques du modèle de Poincaré, c'est-à-dire soit des diamètres, soit des arcs de cercle (cf. fig. 8).

La visualisation *hypertree* est une technique de visualisation « focus+contexte » : elle donne une vue détaillée sur une petite surface (*focus*), tout en gardant une vue globale de la structure (*contexte*). Puisque l'ensemble du plan hyperbolique est projeté dans le disque unité, une distorsion de l'espace apparaît : plus on s'approche du bord du disque, plus les distances sont compressées. Ainsi, le centre du disque fournit une vue détaillée sur une partie des nœuds de l'arbre (*focus*), tandis que le bord du disque fournit une vue globale du reste de la structure de l'arbre (*context*).

La visualisation *hypertree* est une technique interactive. L'arbre peut être déplacé de façon à avoir une vue en détail au centre du disque de n'importe laquelle de ses parties. Un nœud peut être sélectionné pour être mis directement au centre (cf. fig. 18). Une

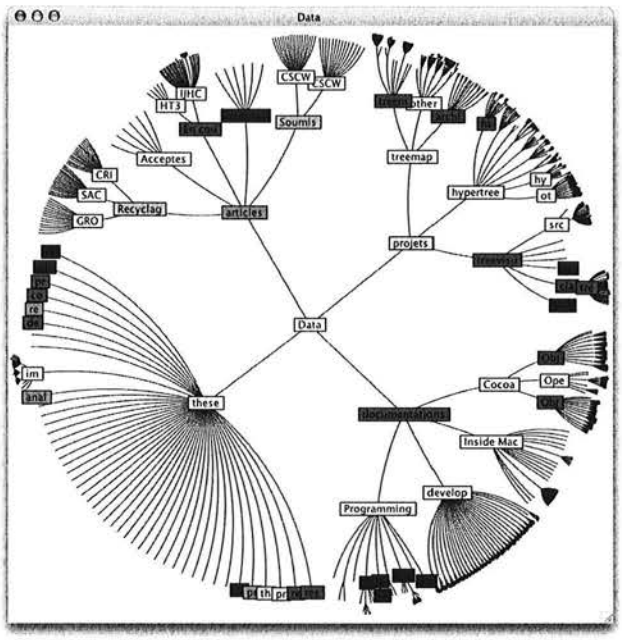

FIG. 7 – La visualisation *hypertree*.

animation de la transition permet à l'utilisateur de suivre le déplacement de l'arbre.

La visualisation *hypertree* est particulièrement adaptée pour la navigation dans des grandes hiérarchies. Plusieurs propriétés aident l'utilisateur pendant sa navigation. Tout d'abord, la visualisation *hypertree* présente l'ensemble de la hiérarchie dans le disque unité. Même si, pour une certaine distance, certains nœuds ne sont plus distinguables, écrasés par la distorsion hyperbolique, un grand nombre de nœuds restent visibles, donnant ainsi un contexte au focus local. Cela donne à l'utilisateur une idée de sa place dans la hiérarchie. La géométrie hyperbolique rend aussi la navigation plus efficace que dans la géométrie euclidienne, car la distance parcourue en un clic ou un mouvement est plus grande, et il y a plus de nœuds affichés à chaque déplacement. Enfin, l'utilisateur peut choisir son mode de navigation : soit une navigation libre en bougeant l'arbre lui-même, ou bien une navigation structurée en sélectionnant les nœuds les uns après les autres. L'animation de l'arbre aide à la visualisation du chemin de navigation, et fournit des informations additionnelles par rapport à la direction de navigation.

La technique de visualisation *hypertree* a été principalement créée pour représenter des données hiérarchiques arborescentes, mais elle peut aussi représenter des liens supplémentaires entre les nœuds. De plus, c'est une visualisation intuitive et directement utilisable par de nouveaux utilisateurs, même si la distorsion hyperbolique peut quelque peu désorienter l'utilisateur la première fois. Ensuite, c'est une visualisation esthétiquement plaisante. Enfin, les nœuds peuvent être colorés suivant un critère des données.

Par exemple, la figure 18 représente une vue *hypertree* d'un système de fichiers. Sur la gauche, le nœud *projects* est dans la zone de *focus*, au centre du disque. On peut voir que la structure globale des sous-répertoires *treemap*, *hypertree* et *treevisu*. Sur la droite, le

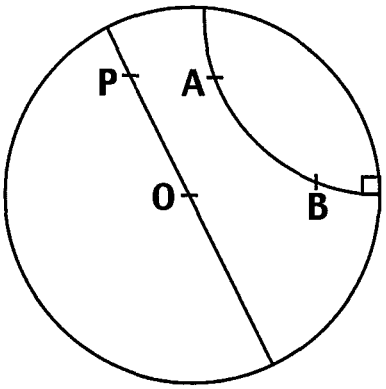

FIG. 8 – Deux types de géodésiques : un diamètre passant par O et P et un arc de cercle $\widehat{AB}$ orthogonal au cercle unité

nœud *treemap* a été déplacé au centre du disque et est maintenant dans la zone de *focus*. On a maintenant une vue en détail de ce nœud, tout en conservant en vue la structure globale des autres nœuds.

Le principal désavantage de la visualisation *hypertree* est que même si la structure est globalement visible, certains nœuds peuvent être trop près du bord du disque et donc impossible à distinguer à cause de la distorsion hyperbolique. Il est donc impossible de comparer en un coup d'œil tous les nœuds de l'arbre. La visualisation *hypertree* donne une vue du contexte, mais pas une vue globale ni même une vue uniforme des nœuds visibles. La visualisation *hypertree* ne permet pas la comparaison entre tous les nœuds, ni même entre certains nœuds et le reste de l'arbre. Elle ne permet la comparaison que des nœuds proches dans la hiérarchie, même s'il est possible d'assigner des couleurs aux nœuds pour aider la différentiation des sous-structures de la hiérarchie.

3.2 État de l'art

La compagnie Xerox a fortement breveté la visualisation en arbre hyperbolique aux États-Unis, ce qui a très fortement réduit la recherche sur cette visualisation, comme en témoigne le très faible nombre de travaux sur une technique pourtant si efficace. Les travaux les plus récents sont publiés par des personnes de Xerox, et décrivent une théorie expliquant la supériorité de la visualisation *hypertree* pour certaines tâches de navigation [PCVDW00, PCVDW01]. Il n'existe à notre connaissance que trois travaux parlant de l'implantation de cette technique de visualisation dans d'autres cadres, dans d'autres langages, ou avec des variations algorithmiques, dont un article co-écrit par l'auteur [BB02, Bul96, Miq00]. Finalement, il n'existe qu'un seul travail sur le passage en 3 dimensions de la visualisation *hypertree* : c'est celui de Munzner [Mun97, Mun98, MB95]. Son principe est identique à celui des *cones-tree* : les nœuds fils sont placés non pas sur un cercle ayant pour centre le nœud père, mais soit sur un disque, soit sur une demi-sphère. Le problème de placement des fils se ramène alors au problème éminemment difficile d'empilement de sphères. La solution proposée par Munzner passe par un placement non optimal

sur des bandes concentriques. De plus, la solution utilise non pas le modèle de Poincaré, mais le modèle de Klein, pour une facilité d'implantation et une question de performance. Il est à noter qu'il existe aussi une implantation différente, basé sur une variation de l'algorithme de placement [Hyu00]. Une autre solution, utilisant un algorithme différent basé sur celui de la visualisation *treemap*, est en cours d'implantation [BB04].

3.3 Contribution

La visualisation *hypertree* permet de représenter des données arborescentes et permet une navigation rapide et efficace dans un grand ensemble de données. Elle donne même une bonne idée de l'organisation générale des données. Elle est donc tout a fait adaptée pour la représentation active des données de conscience de groupe.

Malheureusement, au début de cette thèse, il n'existait aucune implantation libre de cette visualisation. Une des contributions de cette thèse est donc la réalisation d'une implantation libre de la visualisation *hypertree* : l'Hypertree Java Library[17]. Il s'agit, comme son nom l'indique, d'une bibliothèque écrite en Java. En tant que bibliothèque, son cadre d'utilisation est générique : elle peut être utilisée par n'importe quelle application pour représenter n'importe quel type de données arborescentes. Elle a par exemple déjà été utilisée pour analyser la structure de code source [RSB02], ou bien pour visualiser des données médicales [Bou02]. De plus, elle est distribuée sous une licence libre et open-source, ce qui la rend utilisable par tous et dans n'importe quelle application, sans aucune contrainte. Enfin, son code source étant accessible et modifiable par tous, elle peut être adaptée efficacement aux besoins de n'importe qui.

L'implantation de la visualisation *hypertree* est plus ardue que celle de la visualisation *treemap*. Non seulement les algorithmes et les formules mathématiques utilisées sont plus complexes, mais en plus les articles les décrivant sont moins nombreux, et contiennent parfois des erreurs dans les formules. C'est la raison pour laquelle nous avons décidé, avec un collègue docteur en mathématiques, Benjamin Bergé, de repartir des bases mathématiques pour retrouver les formules à utiliser ainsi que les algorithmes s'appuyant sur ces formules [BB02]. Cela a permis non seulement une amélioration de certaines formules et de l'algorithme, mais aussi l'ajout de certaines fonctionnalités inédites, et enfin une meilleure compréhension de la géométrie hyperbolique qui a débouché sur un passage à la troisième dimension [BB04].

Une autre contribution de ce travail est l'amélioration de l'algorithme de placement des nœuds de l'arbre dans le modèle de Poincaré. L'algorithme original est basé sur des calculs d'angles. Notre algorithme est basé sur des secteurs angulaires définis par trois points, dont deux sont sur le cercle unité. Il faut savoir que ce cercle, représentant l'infini, est globalement invariant pour les isométries hyperboliques. Cela signifie que lorsqu'une transformation est appliquée à un tel secteur angulaire, le résultat reste un secteur angulaire. Cela permet de simplifier les calculs en manipulant toujours les mêmes objets et les mêmes transformations tout au long de l'algorithme pour le placement des nœuds de l'arbre. De plus, l'algorithme devient ainsi plus générique. Il est notamment possible d'utiliser le même algorithme pour un passage en 3D, et cela quel que soit alors la forme

[17] `http://hypertree.sf.net/`

des « secteurs angulaires » 3D.

L'Hypertree Java Library propose aussi comme fonctionnalité la possibilité d'utiliser non plus le modèle de Poincaré, mais le modèle de Klein. Dans ce modèle, les géodésiques sont toutes des segments de droite. Par contre, les angles ne sont plus conformes. Récemment, un travail en collaboration avec Benjamin Bergé et le laboratoire mathématique de Neuchâtel, en Suisse, a permis de généraliser la méthode d'obtention des modèles, ce qui autorise un passage continu du modèle de Poincaré au modèle de Klein, et inversement. Cette fonctionnalité a été implantée dans la dernière version de la bibliothèque.

Depuis l'implantation et la publication sur le web de l'Hypertree Java Library, très peu d'autres implantations de la visualisation *hypertree* ont vu le jour sur Internet. De plus, dans la majorité des cas ces implantations n'utilisent pas véritablement les formules de la géométrie hyperbolique, et n'appliquent finalement qu'une simple distorsion des distances. À notre connaissance, la seule autre véritable implantation libre de la visualisation *hypertree* est la bibliothèque *Treebolic*[18] de Bernard Bou. Cette bibliothèque est d'ailleurs en partie basée sur les formules mathématiques et sur l'algorithme de l'Hypertree Java Library, comme cela est indiqué sur sa page des remerciements. À ce jour, l'Hypertree Java Library reste la seule implantation de la visualisation *hypertree* qui soit libre, générique, et qui fournisse autant de fonctionnalités.

3.4 Algorithmes

Au niveau algorithmique, la visualisation *hypertree* est composée de deux parties :
- un algorithme de placement, qui dessine l'arbre dans le modèle de Poincaré ;
- un algorithme de déplacement, qui permet à l'utilisateur d'interagir avec l'arbre.

L'algorithme de placement développé dans la présente implantation de l'Hypertree Java Library est différent de l'algorithme développé par Miquel [Miq00], qui est lui-même différent de celui de Lamping et Rao [LR96a]. Par contre, l'algorithme de déplacement est le même.

Pour pouvoir expliquer et comparer ces algorithmes, il faut d'abord revenir sur quelques bases mathématiques de la géométrie hyperbolique.

Bases mathématiques

Le but ici est de faire un rappel des formules utilisées dans le reste du texte. Le lecteur est invité à se référer à [BB02] pour l'explication et la preuve des formules et des théorèmes.

La première notion importante est celle de géodésique. Intuitivement, une géodésique est le plus court chemin entre deux points. Dans la géométrie euclidienne, les géodésiques sont les lignes droites. La deuxième notion importante est celle de parallélisme : deux géodésiques sont dites parallèles si leur seul point d'intersection est le point à l'infini. En terme de géodésique, le cinquième postulat d'Euclide devient alors : « par un point en dehors d'une géodésique donnée, il passe une et une seule géodésique parallèle à la géodésique donnée ».

[18] `http://treebolic.sortilege.net/`

Il existe deux types de géométries non-euclidiennes, c'est-à-dire qui ne respectent pas le cinquième postulat d'Euclide sur le parallélisme : la géométrie elliptique et la géométrie hyperbolique. Un exemple d'espace elliptique de dimension deux est donnée par la surface d'une sphère. Sur la surface d'une sphère, les géodésiques sont les grands arcs de cercles, comme l'équateur. Il est clair que deux grands cercles non confondus se coupent forcément en deux points, et donc ne peuvent être parallèles. La négation du cinquième postulat d'Euclide est donc l'absence d'une géodésique parallèle.

Pour la géométrie hyperbolique, la négation du cinquième postulat d'Euclide est l'existence d'une infinité (et non pas d'une seule) géodésique parallèle. Malheureusement, il a été prouvé qu'il n'existe pas de surface dans l'espace euclidien à trois dimensions qui puisse représenter un espace hyperbolique de dimension 2. D'ou la nécessité de passer par un modèle. Plusieurs modèles de la géométrie hyperbolique existent, notamment le modèle de Poincaré et le modèle de Klein.

Le modèle de Klein et le modèle de Poincaré représentent tous les deux l'ensemble de l'espace hyperbolique à l'intérieur du disque unité (de rayon 1). L'avantage du modèle de Poincaré est qu'il s'agit d'un modèle conforme : les angles mesurés euclidiennement dans le disque unité correspondent aux angles hyperboliques. Par contre, les géodésiques sont les diamètres du disque ainsi que les arcs de cercle orthogonaux au disque (cf. fig. 8). La visualisation *hypertree* utilise donc le modèle de Poincaré, et toutes les formules données par la suite s'appliquent sauf mention spécifique à ce modèle.

Les premières formules utiles concernent la distance hyperbolique. La distance hyperbolique entre deux points x et y appartenant au disque unité est donnée par la formule suivante :

$$d_H(x,y) = \operatorname{argch}\left(1 + \frac{2|x-y|_E^2}{(1-|x|_E^2)(1-|y|_E^2)}\right) \tag{1}$$

avec $|x|_E$ la norme euclidienne du vecteur Ox.

Lorsque l'un des deux points est l'origine O, la formule se simplifie pour donner

$$d_H(O,x) = 2\operatorname{argth}|x|_E. \tag{2}$$

Après la distance hyperbolique, ce sont les isométries hyperboliques, c'est-à-dire les transformations conservant les distances, qui nous intéressent, et plus particulièrement les translations et les rotations. Soit z l'affixe complexe d'un point du disque unité, et b l'affixe complexe d'un vecteur de norme inférieure à 1 ($|b|_E < 1$). Alors, la translation hyperbolique de vecteur b est donnée par la formule suivante :

$$t_b(z) = \frac{z+b}{1+\bar{b}z}. \tag{3}$$

En ce qui concerne les rotations hyperboliques de centre O, elles sont confondues avec les rotations euclidiennes de centre O. La formule d'une rotation hyperbolique de centre O et d'angle $\Theta \in [0, 2\pi[$ est donc de la forme

$$r_\Theta(z) = ze^{i\Theta} = \theta z \tag{4}$$

avec $\theta = e^{i\Theta}$.

Il faut noter que contrairement à ce qui se passe dans la géométrie euclidienne, la composée de deux translations hyperboliques n'est pas une translation hyperbolique, mais une transformation qui peut être exprimée comme la composition d'une rotation hyperbolique de centre O et d'une autre translation hyperbolique. Il faut donc s'intéresser à un type particulier de transformation : les transformations étant la composée d'une rotation hyperbolique de centre O suivie par une translation hyperbolique. Avec les mêmes notations que précédemment, une telle transformation s'exprime par la formule suivante

$$\tau_{\theta,b}(z) = \frac{\theta z + b}{1 + \bar{b}\theta z}. \tag{5}$$

On a alors le théorème suivant.

Théorème 1 *L'ensemble* $\mathcal{M}(B^2) = \{\tau_{\theta,b} \in I(B^2) : |\theta| = 1, |b| < 1\}$ *est un groupe pour la composition des applications. De plus, tout élément de* $\mathcal{M}(B^2)$ *laisse le cercle unité invariant.*

Cela signifie que la composée de deux transformations reste une transformation, c'est-à-dire que la composée d'un nombre quelconque de rotations hyperboliques de centre O et de translations hyperboliques, dans n'importe quel ordre, peut toujours s'exprimer comme le résultat d'une seule rotation hyperbolique de centre O suivi d'une seule translation hyperbolique. Le deuxième résultat du théorème signifie que l'application d'une transformation à un point sur le cercle unité (donc au bord du disque) donnera toujours un autre point sur le cercle unité.

Placement de l'arbre

L'algorithme de positionnement doit placer les différents nœuds de façon à optimiser l'utilisation de l'espace. Dans le cas de la visualisation *hypertree*, le placement est fait de manière radiale. Cela signifie que la racine de l'arbre est placée au centre O et que le reste de l'arbre (P_1, P_2, P_3 et leurs fils) s'étend dans toutes les directions à partir de cette racine (fig. 9).

Un premier algorithme de placement possible est de placer l'arbre dans le plan euclidien, et d'utiliser la formule (2) concernant la distance hyperbolique pour projeter les points de l'arbre dans le disque unité. Ceci n'est pas un bon algorithme. En effet, il n'utilise pas les propriétés de la géométrie hyperbolique, et notamment sa place supplémentaire par rapport à l'espace euclidien. Le résultat est un simple effet de loupe, mais sans le gain de place et l'animation propre à la géométrie hyperbolique. Cet algorithme est cependant souvent utilisé dans des visualisations se disant « en arbre hyperbolique ».

Pour pouvoir utiliser les propriétés de la géométrie hyperbolique, il faut construire le placement de l'arbre directement dans le modèle de Poincaré, en s'inspirant de l'algorithme de placement dans l'espace euclidien. La suite de cette partie explique l'algorithme de placement radial euclidien puis hyperbolique développé par Lamping et Rao [LR96a], car il s'agit le l'algorithme historique. Il s'agit aussi, pour des raisons historiques, du premier algorithme utilisé dans l'Hypertree Java Library et décrit dans [BB02]. L'algorithme de Miquel [Miq00] et l'algorithme présentement utilisé dans l'Hypertree Java Library étant des variations ou simplifications de cet algorithme, ils seront décrits à la suite.

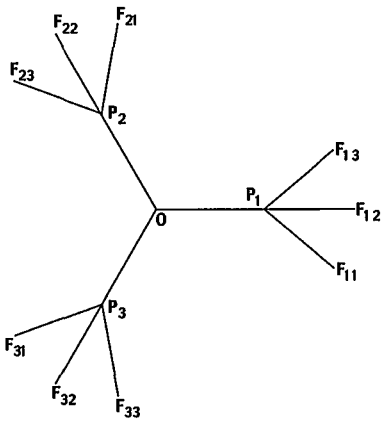

FIG. 9 – Un exemple de placement d'arbre radial dans le plan euclidien.

Placement radial dans le plan euclidien

L'idée principale est la suivante. Pour chaque génération ayant un nœud père et des nœuds fils, les fils sont placés sur un cercle ayant pour centre le père. Pour l'instant, nous allons considérer que la distance séparant un père de ses fils est une constante pour toutes les générations. Soit R cette distance.

L'algorithme commence par placer la racine de l'arbre à l'origine O. Soit n le nombre de nœuds de la première génération (les fils). Ces n points $P_1, \ldots, P_n$ sont placés sur le cercle de centre O et de rayon R. Pour avoir une occupation optimale de l'espace, les 2π radians du cercle sont répartis entre les n fils. Chacun des nœuds P_j de première génération reçoit un secteur angulaire θ_j. Posons :

- $\Theta_1 = 0$;
- $\Theta_j = \Theta_{j-1} + \theta_{j-1}$, $j = 2, \ldots, n$;
- Q_j le point de coordonnées $(\cos\Theta_j, \sin\Theta_j)$.

Chacun des n nœuds P_j est alors placé sur la bissectrice du secteur angulaire $\widehat{Q_jOQ_{j+1}}$, à une distance R de O (fig. 10). On a donc

$$P_j : \left(R\cos\left(\Theta_j + \frac{\theta_j}{2}\right), R\sin\left(\Theta_j + \frac{\theta_j}{2}\right)\right). \tag{6}$$

Une fois P_j placé, on s'intéresse à ses fils. L'algorithme pour placer les fils de P_j est le même que pour placer les fils de O, à la différence près que le secteur angulaire à répartir n'est plus 2π. Un algorithme simple est de fournir le même secteur angulaire que celui alloué à P_j, c'est-à-dire θ_j. Ce secteur angulaire va être réparti symétriquement autour de la droite (OP_j). Cette droite fait un angle $\Theta_j + \frac{\theta_j}{2}$ avec l'axe des abscisses. Le secteur angulaire à partager ira donc de $\left(\Theta_j + \frac{\theta_j}{2}\right) - \frac{\theta_j}{2} = \Theta_j$ à $\left(\Theta_j + \frac{\theta_j}{2}\right) + \frac{\theta_j}{2} = \Theta_{j+1}$ (fig. 11). Posons :

- n_j le nombre de fils de P_j ;
- $F_{j,k}$ le k^{e} fils de P_j ;
- $\theta_{j,k}$ le secteur angulaire alloué à $F_{j,k}$ et bissecté par $(P_jF_{j,k})$;

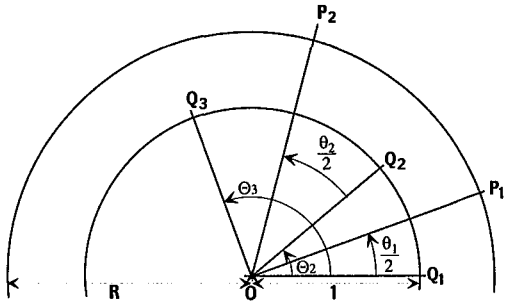

FIG. 10 – Placement de la première génération de nœuds.

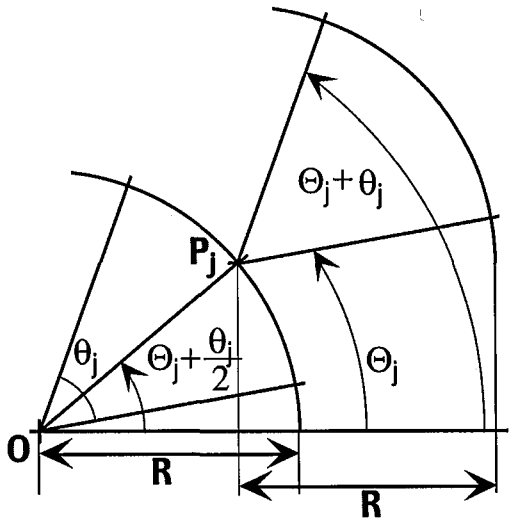

FIG. 11 – Le secteur angulaire à partager pour les fils de P_j

- $\Theta_{j,1} = 0$;
- $\Theta_{j,k} = \Theta_{j,k-1} + \theta_{j,k-1}$, $k = 2, \ldots, n_j$.

Les nœuds $F_{j,k}$ sont alors placés à une distance R de P_j, sur la bissectrice du secteur angulaire qui leur a été alloué (comme précédemment). On note $(\cdot,\cdot)_Q$ les coordonnées relatives dans le repère ayant pour origine le point Q. Pour trouver les coordonnées cartésiennes de $F_{j,k}$, nous changeons de repère, par translation, en prenant P_j comme nouvelle origine. Dans ce nouveau repère, les coordonnées relatives de $F_{j,k}$ sont

$$\left(R\cos\left(\Theta_j + \Theta_{j,k} + \frac{\theta_{j,k}}{2}\right),\ R\sin\left(\Theta_j + \Theta_{j,k} + \frac{\theta_{j,k}}{2}\right)\right)_{P_j}.$$

Il ne reste plus qu'à appliquer à ces coordonnées la translation de vecteur $\overrightarrow{OP_j}$ pour trouver les coordonnées absolues des $F_{j,k}$ dans le repère d'origine O. Si $P_j = \left(X_{P_j}, Y_{P_j}\right)$, alors les

coordonnées du vecteur $\overrightarrow{OP_j}$ sont (X_{P_j}, Y_{P_j}). Les coordonnées absolues de $F_{j,k}$ sont donc

$$\left(R\cos\left(\Theta_j+\Theta_{j,k}+\frac{\theta_{j,k}}{2}\right)+X_{P_j},\ R\sin\left(\Theta_j+\Theta_{j,k}+\frac{\theta_{j,k}}{2}\right)+Y_{P_j}\right).$$

Pour la troisième génération, on fait de même, à une nuance près : le secteur angulaire doit être réparti autour de la droite $(P_jF_{j,k})$. Cette droite fait un angle $\alpha=\Theta_j+\Theta_{j,k}+\frac{\theta_{j,k}}{2}$ avec l'axe des abscisses. Donc, dans le repère de centre $F_{j,k}$, les coordonnées relatives des fils $G_{j,k,l}$ de $F_{j,k}$ sont (avec $\Theta_{j,k,l}$ et $\theta_{j,k,l}$ définis par analogie)

$$\left(R\cos\left(\left(\alpha-\frac{\theta_{j,k}}{2}\right)+\left(\Theta_{j,k,l}+\frac{\theta_{j,k,l}}{2}\right)\right),\ R\sin\left(\left(\alpha-\frac{\theta_{j,k}}{2}\right)+\left(\Theta_{j,k,l}+\frac{\theta_{j,k,l}}{2}\right)\right)\right)_{F_{j,k}},$$

ou encore, plus simplement,

$$\left(R\cos\left(\Theta_j+\Theta_{j,k}+\Theta_{j,k,l}+\frac{\theta_{j,k,l}}{2}\right),\ R\sin\left(\Theta_j+\Theta_{j,k}+\Theta_{j,k,l}+\frac{\theta_{j,k,l}}{2}\right)\right)_{F_{j,k}}.$$

Enfin, on applique l'algorithme récursivement sur toutes les générations pour trouver les coordonnées de tous les nœuds de l'arbre :

- la racine de l'arbre est mise à l'origine O : (0, 0) ;
- pour la première génération P_j :
  - on répartit 2π radians entre les fils. Chacun reçoit θ_j comme secteur angulaire,
  - $\Theta_1=0$,
  - $\Theta_j=\Theta_{j-1}+\theta_{j-1}$, $j=2,\ldots,n$,
  - P_j : $\left(R\cos\left(\Theta_j+\frac{\theta_j}{2}\right),\ R\sin\left(\Theta_j+\frac{\theta_j}{2}\right)\right)$,
  - le vecteur $\overrightarrow{OP_j}$ fait un angle $\alpha_j=\Theta_j+\frac{\theta_j}{2}$ avec l'horizontale ;
- pour les générations suivantes :
  - soit P de coordonnées (X_P, Y_P) le père ayant reçu θ comme secteur angulaire,
  - soit $F_1,\ldots,F_n$ les fils de P,
  - soit M le père de P,
  - soit Θ l'angle que fait le vecteur $\overrightarrow{MP}$ avec l'horizontale,
  - on répartit θ entre les fils $F_1,\ldots,F_n$. Chacun reçoit θ_j comme secteur angulaire,
  - $\Theta_1=0$,
  - $\Theta_j=\Theta_{j-1}+\theta_{j-1}$, $j=2,\ldots,n$,
  - F_j : $\left(R\cos\left(\left(\Theta-\frac{\theta}{2}\right)+\left(\Theta_j+\frac{\theta_j}{2}\right)\right)+X_P,\ R\sin\left(\left(\Theta-\frac{\theta}{2}\right)+\left(\Theta_j+\frac{\theta_j}{2}\right)\right)+Y_P\right)$ $:=(X_{F_j}, Y_{F_j})$
  - le vecteur $\overrightarrow{PF_j}$ fait un angle $\Theta'=\left(\left(\Theta-\frac{\theta}{2}\right)+\left(\Theta_j+\frac{\theta_j}{2}\right)\right)$ avec l'horizontale.

Placement radial dans le plan hyperbolique

Dans le plan hyperbolique, l'algorithme de placement de l'arbre est le même que dans le plan euclidien. Par contre, les formules sont différentes, puisque l'on ne peut plus utiliser les formules de trigonométrie classique.

La première étape du placement dans l'espace hyperbolique est la même que celle du placement dans l'espace euclidien : la racine de l'arbre est placée à l'origine O de l'espace. Puis ses n fils $P_1, \ldots, P_n$ vont être placés sur le cercle hyperbolique de centre O et de rayon R. En effet, les cercles de centre O de l'espace hyperbolique restent des cercles de centre O dans le modèle de Poincaré. Comme dans l'espace euclidien, les 2π radians du cercle vont être répartis entre les n fils et chacun des nœuds de première génération P_j va recevoir un secteur angulaire θ_j.

Puisque la racine, qui est le nœud père, se trouve à l'origine, toutes les géodésiques reliant le nœud père à ses fils sont des diamètres, donc de « vraies » droites. Nous pouvons alors appliquer ici les formules de trigonométrie classique. Ainsi, les coordonnées des fils P_j de première génération sont

$$\left(R\cos\left(\Theta_j + \frac{\theta_j}{2}\right),\ R\sin\left(\Theta_j + \frac{\theta_j}{2}\right)\right)$$

comme dans l'espace euclidien (6), avec

- θ_j le secteur angulaire alloué à P_j ;
- $\Theta_1 = 0$;
- $\Theta_j = \Theta_{j-1} + \theta_{j-1}$, $j = 2, \ldots, n$.

Par contre, pour la génération suivante, les calculs sont différents. En effet, les points de la première génération n'étant plus au centre O du disque unité, les géodésiques reliant ces nœuds à leurs fils ne sont plus des droites mais des arcs de cercles orthogonaux au cercle unité (cf. fig. 8). Ici, les formules de trigonométrie ne vont plus s'appliquer. Puisque ces formules ne sont utilisables que lorsque le nœud père est placé à l'origine, nous allons faire un changement de repère : nous allons placer les nœuds fils par rapport à un nœud père centré à l'origine. Cela se fera en appliquant les formules de la section précédente. Puis nous appliquerons aux points obtenus la translation remettant le nœud père à sa véritable place. Cependant, utiliser cette méthode a un désavantage : lorsque l'on translate un fils, le diamètre (PF) reliant le père et ce fils se transforme en arc de cercle. Or, c'est de part et d'autre de cette géodésique que doivent être placés les fils de F. Cela signifie que l'angle autour duquel les fils de F doivent être répartis n'est pas l'angle euclidien entre le diamètre (PF) et l'axe des abscisses mesuré avant la translation. Plus exactement, soit H le point d'intersection du diamètre (PF) et du cercle unité lorsque l'on a P au centre du cercle. Si l'on pose Θ l'angle entre (PF) et l'axe des abscisses, l'affixe de H vaut $e^{i\Theta}$ (fig. 12).

Lorsque l'on remet P à sa véritable place par la translation hyperbolique t_{z_P} de vecteur $\overrightarrow{OP}$, le diamètre (PF) est transformé en un arc de cercle orthogonal au cercle unité. C'est autour de cette géodésique qu'il va falloir placer les fils de F. C'est donc l'angle hyperbolique que cette géodésique fait en F avec l'axe des abscisses qu'il faut trouver. Posons H' l'image de H par la translation hyperbolique t_{z_P} (fig. 13). Or, H' est sur le cercle unité.

Comme il nous faut l'angle en F par rapport à l'axe des abscisses, nous allons faire une translation t_{-z_F} de vecteur $-\overrightarrow{OF}$ qui va placer F à l'origine. Posons H'' l'image de H' par cette translation. Comme H' est sur le cercle unité, H'' est aussi sur le cercle unité. Son affixe est de la forme $e^{i\Theta''}$, avec Θ'' l'angle recherché.

C'est autour de cet angle que les fils de F vont être répartis. Donc, si :

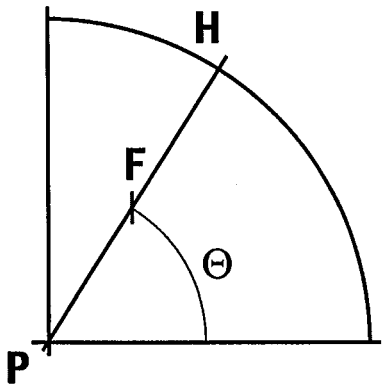

FIG. 12 – Placement du fils F avec le père P à l'origine.

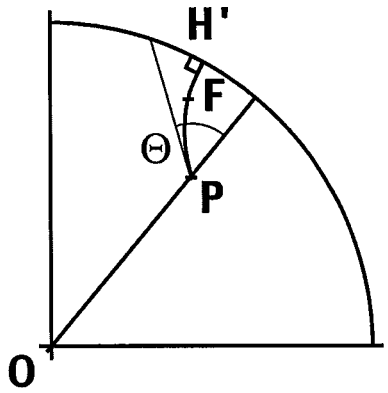

FIG. 13 – Image de la figure 12 après la translation t_{z_P}.

– P (d'affixe z_P) est le père de F (d'affixe z_F) ;
– le diamètre (PF) fait un angle Θ avec l'axe des abscisses lorsque P est à l'origine ;
– Θ'' est l'angle autour duquel les fils de F doivent être répartis ;
alors on a

$$e^{i\Theta''} = t_{-z_F} \circ t_{z_P}(e^{i\Theta}). \tag{7}$$

Une fois l'angle de la géodésique trouvé, on peut appliquer les formules valables dans l'espace euclidien. Ainsi, les coordonnées relatives des fils $F_{j,k}$ sont

$$\left(R\cos\left(\left(\Theta''_j - \frac{\theta_j}{2}\right) + \left(\Theta_{j,k} + \frac{\theta_{j,k}}{2}\right)\right), \sin\left(\left(\Theta''_j - \frac{\theta_j}{2}\right) + \left(\Theta_{j,k} + \frac{\theta_{j,k}}{2}\right)\right)\right)_{P_j}$$

$:= \left(X_{F_{j,k}}, Y_{F_{j,k}}\right)_{P_j}$ avec
– M le père de P_j ;
– Θ_j l'angle entre le diamètre (MP_j) et l'axe des abscisses lorsque M est à l'origine ;
– Θ''_j l'angle calculé à partir de Θ_j dans la formule (7) ;
– θ_j le secteur angulaire à répartir entre les fils de P_j ;

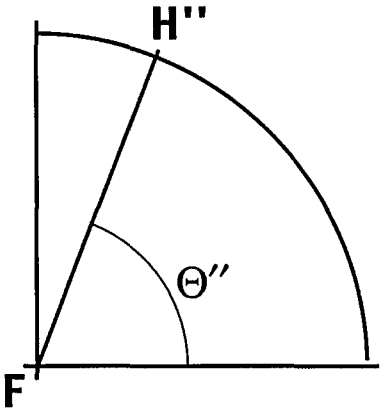

FIG. 14 – Image de la figure 13 après la translation t_{-z_F}.

- $\theta_{j,k}$ le secteur angulaire alloué à $F_{j,k}$;
- $\Theta_{j,1} = 0$;
- $\Theta_{j,k} = \Theta_{j,k-1} + \theta_{j,k-1}$, $k = 2, \ldots, n$.

Pour les fils de $F_{j,k}$, l'équivalent de Θ_j vaudra $\left(\Theta''_j - \frac{\theta_j}{2}\right) + \left(\Theta_{j,k-1} + \frac{\theta_{j,k}}{2}\right)$.

Il ne nous reste donc qu'à appliquer la translation hyperbolique de vecteur $\overrightarrow{OP_j}$. Or, une translation hyperbolique de vecteur b a pour expression analytique

$$t_b(z) = \frac{z+b}{1+\bar{b}z}. \tag{8}$$

Soit $p_j = X_{P_j} + iY_{P_j}$ (respectivement $f_{j,k} = X_{F_{j,k}} + iY_{F_{j,k}}$) l'affixe du père P_j dans le repère de centre O (respectivement l'affixe du fils $F_{j,k}$ dans le repère de centre P_j). Alors, l'affixe $z_{F_{j,k}}$ du fils $F_{j,k}$ dans le repère de centre O est donnée par

$$z_{F_{j,k}} = \frac{f_{j,k} + p_j}{1 + \overline{p_j} f_{j,k}}. \tag{9}$$

Ainsi, récursivement, cet algorithme permet de placer tous les points de l'arbre dans l'espace hyperbolique représenté par le disque de Poincaré.

Optimisation du secteur angulaire

Un tel placement ne tient pas compte des possibilités offertes par l'espace hyperbolique en ce qui concerne l'optimisation de l'espace utilisé. En particulier, les propriétés propres à l'espace hyperbolique permettent d'optimiser l'allocation du secteur angulaire accordé au père et à répartir entre ses fils. En effet, l'allocation et la répartition du secteur angulaire répondent à une contrainte précise : assurer que deux sous-arbres issus de branches différentes ne se croisent jamais. Pour cela, en géométrie euclidienne, on utilise la propriété qu'ont deux droites parallèles de ne jamais se couper pour délimiter des secteurs angulaires distincts (fig. 15). Le secteur angulaire à répartir entre les fils de P est donc identique au secteur alloué à P.

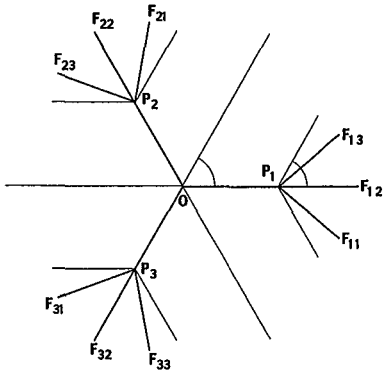

FIG. 15 – Secteur de répartition dans l'espace euclidien.

Dans le modèle de Poincaré, l'espace est confiné au disque unité. Cela permet une première optimisation de l'angle (fig. 16). Il faut de plus prendre en compte le fait qu'en géométrie hyperbolique, les géodésiques ne contenant pas O sont des arcs de cercles orthogonaux au cercle unité (fig. 17). Nous allons calculer le secteur d'angle optimal θ' à accorder au fils du point P auquel a été accordé l'angle θ.

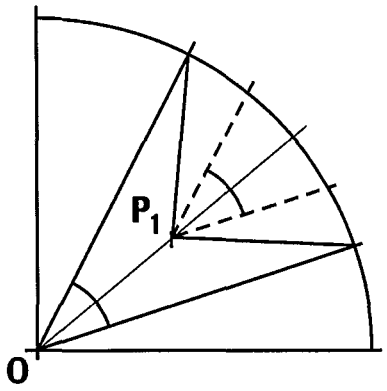

FIG. 16 – Optimisation de l'angle dans un domaine borné.

Commençons par remarquer que la valeur de θ' ne dépend pas de l'angle entre (OP) et l'axe des abscisses. On peut donc considérer pour le calcul que P est sur l'axe des abscisses, à une distance l de O. Soit (OA) le rayon du cercle unité faisant un angle θ avec l'axe des abscisses. L'affixe du point A vaut $e^{i\theta}$. On peut considérer que pour délimiter le secteur angulaire de P, on va utiliser une géodésique parallèle à (OA), passant par P. De par le principe même de la géométrie hyperbolique, il y a une infinité de telles géodésiques. À la limite, la géodésique parallèle à (OA) qui va maximiser l'angle alloué à P coupe (OA) à l'infini, c'est-à-dire dans le modèle de Poincaré sur le cercle unité, donc en A. Cette géodésique délimite le secteur angulaire le plus avantageux pour les fils de P (fig. 17). La translation hyperbolique de coordonnées $(-l, 0)$ translate P à l'origine. Soit A' l'image de A par cette translation. Or, A' est sur le cercle unité. L'image de la

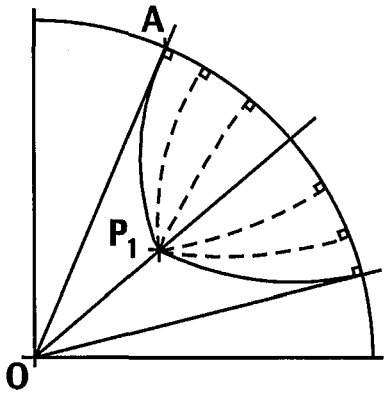

FIG. 17 – Différentes géodésiques parallèles à (OA) dans l'espace hyperbolique.

géodésique (PA) est le diamètre (OA'). Alors, puisque la translation hyperbolique est une transformation conforme, l'angle hyperbolique entre l'axe des abscisses et la géodésique (PA) est le même qu'entre l'axe des abscisses et la géodésique (OA'). Cet angle est l'angle θ' recherché. Puisque (OA') est un diamètre, A' a pour affixe $\mathrm{e}^{i\theta'}$. On a donc la relation

$$\mathrm{e}^{i\theta'} = t_{-l}(\mathrm{e}^{i\theta}) = \frac{\mathrm{e}^{i\theta} - l}{1 - l\mathrm{e}^{i\theta}}. \tag{10}$$

Un autre calcul (détaillé dans [BB02]) utilisant la formule d'Al Kashi [Lal87] permet de simplifier l'équation (10) en la formule

$$\theta' = \arccos\left(\frac{(1+l^2)\cos\theta - 2l}{(1+l^2) - 2l\cos\theta}\right). \tag{11}$$

Autres algorithmes de placement

L'algorithme de placement décrit par Miquel [Miq00] est, comme l'algorithme décrit ci-dessus, un algorithme récursif en deux étapes : d'abord un positionnement des nœuds fils par rapport à un nœud père ramené au centre, puis une translation hyperbolique qui ramène le nœud père à sa véritable place. La différence avec l'algorithme de Lamping et Rao est que l'algorithme de Miquel alloue un secteur angulaire constant (de 180 degrés) à chaque nœud fils. Par contre, la distance père-fils varie d'une génération à l'autre pour permettre aux secteurs angulaires des nœuds fils de ne pas s'intersecter. Cette variation de l'algorithme permet une simplification au niveau des calculs d'angle et se prête particulièrement bien à une implantation type *LOGO*, où le dessin est représenté comme le chemin d'une tortue sur l'écran. Cependant, il ne permet pas les optimisations sur le secteur angulaire ou sur la distance père-fils en fonction du nombre de fils, tels que ceux décrit ci-après.

L'algorithme présentement utilisé dans l'Hypertree Java Library est aussi un algorithme récursif avec placement des nœuds fils en fonction d'un nœud père ramené au centre du disque. Comme l'algorithme de Lamping et Rao, le secteur angulaire alloué à

un nœud fils dépend du secteur angulaire du nœud père et de son nombre de fils. La différence réside dans les calculs et dans les informations passées d'une génération à une autre. L'algorithme de Lamping et Rao attache à chaque nœud un secteur angulaire et une direction. En particulier, le secteur angulaire est défini comme un angle par rapport à la direction. Dans notre algorithme, le secteur angulaire est défini par ses points d'intersection avec le cercle unité. Un secteur angulaire n'est plus défini par un angle, mais par deux points. Comme ces deux points sont sur le cercle unité, le résultat de toute transformation appliqué à ces points reste sur le cercle unité. Il est donc possible d'utiliser ces deux points pour tous les calculs de répartition de secteurs et de placement des nœuds fils. Cela permet notamment de se passer du calcul de la correction de la direction, et de rendre l'algorithme plus générique.

Optimisation en fonction du nombre de fils

L'arbre à représenter est rarement équilibré. Il est possible que certains nœuds aient beaucoup plus de fils que d'autres. Il peut être intéressant d'adapter le placement des nœuds de l'arbre en fonction du nombre de fils, en particulier l'angle alloué à un nœud et la distance séparant le nœud de ses fils. En effet, si un nœud père a plus de fils que ses voisins, il est normal qu'il ait besoin d'un secteur angulaire plus grand pour placer tous ses fils qu'un nœud ayant peu d'enfants. De même, si un nœud père a de nombreux enfants, augmenter la distance père-fils pour ce nœud permettra d'obtenir plus d'espace entre les enfants.

La première adaptation porte sur le secteur angulaire alloué. L'idée consiste à donner un poids à chaque nœud. Ce poids servira à allouer à chaque nœud un secteur proportionnel au quotient de son poids par la somme des poids de ses frères. Par exemple, supposons que le nœud P a un secteur angulaire θ à allouer entre ses n fils $F_1, \ldots, F_n$. Chaque fils F_j a un poids w_j. Le secteur angulaire θ_j alloué au fils F_j vaudra alors

$$\theta_j = \theta \frac{w_j}{\sum_{k=1}^{n} w_k}.$$

En ce qui concerne le calcul du poids w d'un nœud P, remarquons qu'un nœud n'ayant qu'un seul enfant doit recevoir le même secteur angulaire qu'un nœud n'ayant pas d'enfant. Cela nous guide vers la formule suivante :

- $w = 1$ pour les nœuds feuilles ;
- $w_P = 1 + \ln\left(\sum_{\text{F fils de } P} w_F\right)$ pour les autres.

On notera que cette définition est récursive.

En ce qui concerne l'adaptation de la distance père-fils, plusieurs solutions sont possibles. L'Hypertree Java Library a recours à une optimisation ad-hoc. Cette distance père-fils $R(n)$, fonction du nombre n de fils, doit être comprise entre 0 et 1 puisqu'il s'agit de la distance euclidienne séparant le père de ses fils lorsqu'il est au centre du disque. Arbitrairement, cette distance pour un seul fils vaut $R(1) = 0.3$. L'optimisation de cette distance $R(n)$ doit répondre à deux contraintes paradoxales. D'une part, augmenter la distance apporte un plus large espacement, donc une meilleure lisibilité entre les fils. D'autre part, une trop grande distance écrase les données sur le bord du disque. Il faut donc trouver un compromis entre ces deux contraintes.

Pour un grand nombre de fils (par exemple plus de 50), la première contrainte l'emporte : la lisibilité entre les fils est primordiale. Ainsi, pour n grand, $R(n)$ doit être aussi grand que possible. Cependant, le respect de la deuxième contrainte implique un choix moindre que 1 pour $R(\infty)$. Par exemple, $R(\infty) = 0.95$.

Pour peu de fils, la deuxième contrainte prime. La lisibilité est apportée principalement par la fonction zoom du centre du disque de Poincaré. Par conséquent, $R(n)$ doit augmenter progressivement avec n à partir de $R(1)$. Nous donnons ci-dessous une fonction R qui satisfait à ces conditions.

$$R(n) = R + (0,95 - R)\cos\frac{10\pi}{n+19}.$$

Cette fonction, trouvée empiriquement, est très satisfaisante au niveau visuel ainsi qu'au niveau du temps de calcul. Nous aurions pu choisir une interpolation linéaire entre $R(1)$ et $R(\infty) \cong R(50)$, mais plutôt que de choisir une valeur particulière représentant l'infini, nous avons préféré une fonction trigonométrique permettant d'échelonner la distance.

Interactions

La représentation des données arborescentes dans le modèle de Poincaré apporte à l'utilisateur une vue en détail sur une partie des données, tout en gardant le reste de l'arbre visible. Pour qu'une telle visualisation soit utile, il faut que l'utilisateur puisse choisir la partie des données à voir en détail, c'est-à-dire qu'il puisse naviguer dans les données. Pour cela, il faut que l'arbre puisse être déplacé, d'une manière ou d'une autre, afin de mettre la partie de l'arbre sélectionnée par l'utilisateur au centre du disque unité.

L'algorithme de déplacement doit trouver la transformation hyperbolique permettant d'amener le point choisi à l'endroit adéquat. Malheureusement, une simple translation hyperbolique n'est pas appropriée. En effet, lors de la navigation, l'algorithme de déplacement est utilisé de nombreuses fois. Or, la composée de deux translations hyperboliques n'était pas forcément une translation hyperbolique mais une transformation s'exprimant comme la composée d'une rotation hyperbolique de centre O (éventuellement d'angle nul) et d'une translation hyperbolique. Cela signifie que la composition de plusieurs translations hyperboliques peut faire tourner l'arbre autour du centre du disque, ce qui risque de dérouter l'utilisateur. En effet, arriver à faire tourner une figure uniquement par des translations est impossible en géométrie euclidienne. Cet effet de rotation va donc contre le sens commun de l'utilisateur et peut le gêner dans sa navigation.

Pour éviter cette rotation, tous les mouvements sont calculés à partir de la position initiale de l'arbre, lorsque la racine se trouve à l'origine. De cette manière, il n'y a toujours qu'une seule translation appliquée. Cela permet en plus de réduire la propagation des imprécisions de calcul.

L'algorithme utilisé pour les interactions est le suivant. Considérons que l'arbre se trouve dans une position quelconque, avec :

- R la racine de l'arbre, d'affixe z_R ;
- S le point sélectionné par l'utilisateur, d'affixe z_S ;
- E le point correspondant à l'endroit indiqué comme destination, d'affixe z_E.

Il faut trouver la transformation hyperbolique $\tau_{\theta,p}$ telle que $\tau_{\theta,p}(z_S) = z_E$. Cette transformation doit être équivalente à une unique translation hyperbolique t_b appliquée lorsque l'arbre est en position initiale, avec R au centre du disque. Soit t_{-z_R} la translation hyperbolique qui remet R au centre du disque, et z'_S l'affixe de S dans cette position : $z'_S = t_{-z_R}(z_S)$. Il faut donc trouver t_b telle que $t_b(z'_S) = z_E$. Puisque $t_b(z) = \frac{z+b}{1+\bar{b}z}$, il faut trouver b tel que

$$z_E = \frac{z'_S + b}{1 + \bar{b}z'_S}.$$

Après calcul, il vient

$$b = \frac{z_E(1 - |z'_S|^2) - z'_S(1 - |z_E|^2)}{1 - |z_E|^2|z'_S|^2}.$$

La transformation hyperbolique $\tau_{\theta,p}$ cherchée est donc la composée des deux translations hyperboliques t_{-z_R} et t_b. Or, il vient pour $\tau_{\theta,p} = t_b \circ t_{-z_R}$,

$$\theta = \frac{1 - \overline{z_R}b}{1 - z_R\bar{b}}$$

et

$$p = \frac{b - z_R}{1 - z_R\bar{b}},$$

soit encore, pour tout z $\in B^2$,

$$\tau_{\theta,p}(z) = \frac{(1 - \overline{z_R}b)z + (b - z_R)}{\overline{(1 - \overline{z_R}b)} + \overline{(b - z_R)}z},$$

avec

$$b = \frac{z_E(1 - |z'_S|^2) - z'_S(1 - |z_E|^2)}{1 - |z_E|^2|z'_S|^2}.$$

4 Application à la visualisation de la conscience de groupe

La principale force de la visualisation *treemap* est sa vue *gestaltique* de l'ensemble de la hiérarchie, qui permet de comparer différentes caractéristiques des données. Ses deux principaux points faibles sont de ne pas être intuitive pour les personnes l'utilisant pour la première fois, et de ne donner qu'une idée imprécise de la structure de la hiérarchie. La visualisation *treemap* est généralement utilisée pour obtenir une compréhension globale d'un grand ensemble de données.

La principale force de la visualisation *hypertree* est de permettre une vue en détail sur une partie de la hiérarchie tout en gardant son contexte visible. Elle possède de plus une certaine esthétique. Son principal point faible est de ne pas donner une vue globale de l'ensemble des données, empêchant ainsi une comparaison entre toutes les données. La visualisation *hypertree* est généralement utilisée pour naviguer dans un grand ensemble de données.

Ces deux visualisations sont complémentaires. Utilisées sur la même hiérarchie de données, la visualisation *treemap* en donne une vue globale, et la visualisation *hypertree* permet d'y naviguer.

Il a été vu dans le chapitre précédent que la représentation des informations de conscience de groupe devait se faire suivant deux modes :

- un premier mode permettant de notifier l'utilisateur des nouvelles informations reçues (correspondant à des changements du contexte) ;
- un second mode permettant de naviguer dans l'ensemble des informations reçues

Les deux techniques de visualisations *treemap* et *hypertree* répondent parfaitement aux besoins de ces deux modes.

La visualisation *treemap* est une très bonne visualisation pour observer les changements du contexte. Elle permet en effet de visualiser l'ensemble des données, ordonnées suivant la hiérarchie, et de faire des comparaisons entre ces données. La visualisation *treemap* peut ainsi être utilisée pour visualiser les activités des membres du groupe. Chaque rectangle représente alors une personne, suivant sa place dans la hiérarchie. La couleur du rectangle représente l'activité de cette personne. L'utilisateur est notifié qu'un membre du groupe a changé d'activité par le changement de couleur du rectangle correspondant. La visualisation *treemap* peut aussi représenter le temps passé par chacun sur certaines activités, ou encore l'évolution des activités d'une personne spécifique au cours du temps.

La visualisation *hypertree* est une très bonne visualisation pour naviguer dans l'ensemble des informations reçues. En effet, grâce à sa vue « focus + contexte », elle permet de naviguer efficacement dans un grand ensemble de données. La visualisation *hypertree* peut ainsi être utilisée pour naviguer dans l'ensemble des informations haut niveau reçues. Ces informations peuvent être classées suivant les personnes les ayant émises, ou bien suivant une hiérarchie temporelle (année-mois-jour-heure-minute). Les couleurs des nœuds peuvent par exemple représenter la catégorie de l'information, ou encore sa date de réception.

Utilisées ensembles, les deux techniques de visualisations *treemap* et *hypertree* couvrent ainsi les besoins des deux modes.

Ces deux visualisations sont particulièrement adaptées à la représentation d'information de conscience de groupe pour plusieurs raisons. Tout d'abord, elles supportent toutes les deux très bien le passage à l'échelle et la représentation de larges ensembles de données, par exemple lorsque le nombre de participant ou d'artefacts manipulés augmente fortement. De plus, ces deux visualisations sont assez génériques pour être utilisées dans n'importe quelle situation, avec à peu près n'importe quel type d'informations de conscience de groupe. En particulier, elles ne sont pas limitées au cadre spécifique de la conscience de groupe dans les espaces de travail partagé. Enfin, une implantation libre et *open-source* existe pour chacune des visualisations, permettant leur utilisations dans n'importe quel cadre d'application.

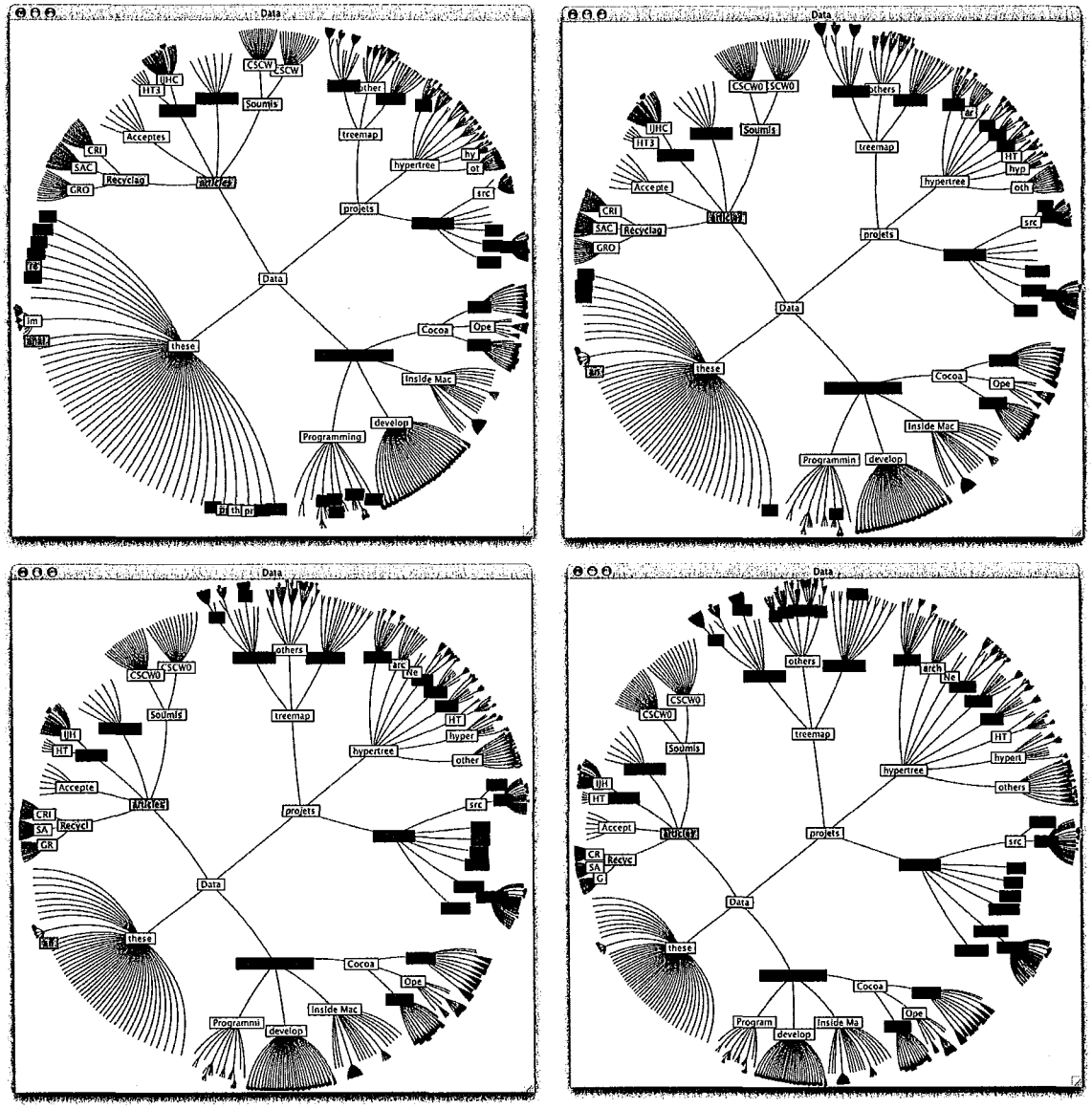

FIG. 18 – La visualisation *hypertree* en mouvement : le nœud « projets » en haut à droite est amené au milieu du disque.

Contributions logicielles

Cette thèse contient deux propositions. La première concerne la réalisation d'un outil de conscience de groupe s'adaptant au contexte de travail de l'utilisateur. La deuxième concerne l'utilisation de deux techniques de visualisation pour représenter les informations de conscience de groupe : la technique *treemap* et la technique *hypertree*.

Dans le cadre de cette thèse, trois réalisations logicielles ont été créées. La première, *CommanderMo*, est un outil de conscience de groupe générique, basé sur l'architecture fonctionnelle décrite précédemment, et capable de s'adapter au contexte de travail de l'utilisateur. Les deux autres, la *TreeMap Java Library* et l'*HyperTree Java Library*, sont deux implantations réutilisables des techniques de visualisation *treemap* et *hypertree*. La suite de ce chapitre détaille ces trois réalisations logicielles.

1 CommanderMo

CommanderMo est un prototype de système de conscience de groupe écrit en Java et implantant les travaux décrits dans le chapitre « Contextualisation ». C'est un outil générique, ne s'appuyant sur aucun outil de coopération spécifique, et qui peut être configuré pour être utilisé dans n'importe quelle situation de coopération. Il s'installe simplement sur les postes des différents participants, sans nécessiter de serveur central. Une fois lancé, *CommanderMo* exécute deux types d'activités. Premièrement, il collecte, agrège, filtre et diffuse les informations concernant l'activité de l'utilisateur à l'ensemble du groupe. Deuxièmement, il reçoit, adapte et présente à l'utilisateur les informations diffusées par les outils des différents participants et concernant les activités du groupe. Toutes ces activités sont réalisées de manière à s'adapter au contexte de travail de l'utilisateur. *CommanderMo* est disponible sous la licence *MIT*, une licence libre et *open-source*.

1.1 Architecture

L'architecture de *CommanderMo* comporte huit modules (cf. fig. 1). Les sept premiers modules correspondent aux sept niveaux de l'architecture fonctionnelle décrite précédemment, à savoir *Collection*, *Agrégation*, *Filtrage*, *Diffusion*, *Réception*, *Présentation* et *Visualisation*. Le huitième module, *Contexte*, est chargé du stockage et de l'accès aux informations reçues. Ces informations de haut niveau constituant le contexte de travail de l'utilisateur.

La collecte des informations bas niveau dans le module *Collection* se fait par l'appel périodique à des fonctions du système d'exploitation. Tous les systèmes d'exploitation

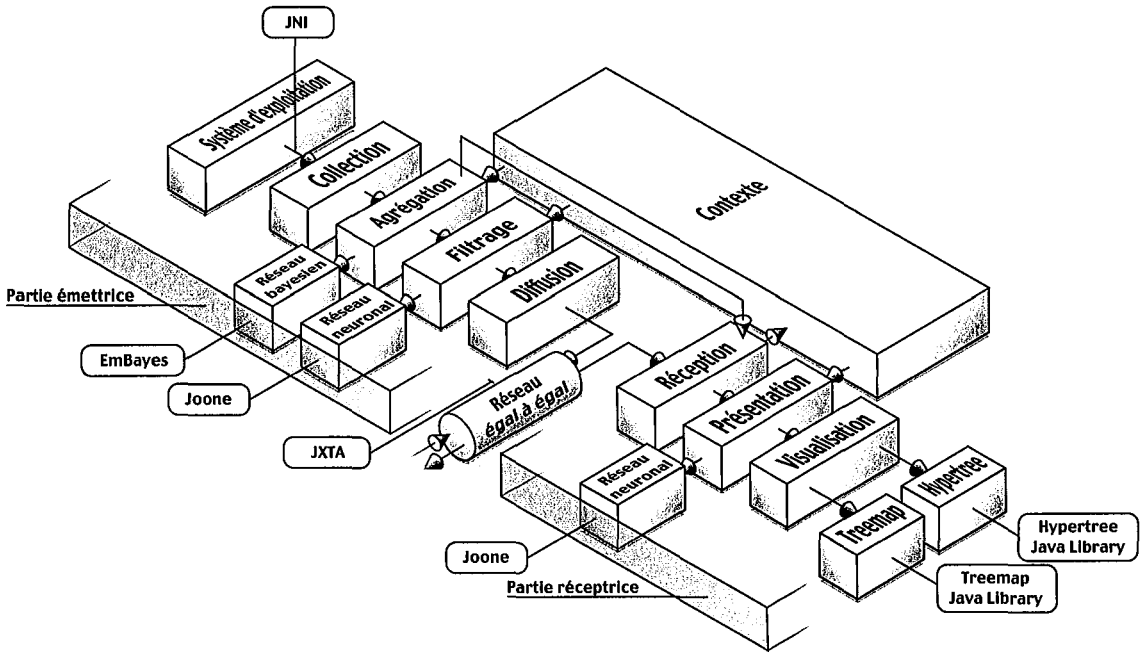

FIG. 1 – L'architecture de *CommanderMo*.

possèdent une interface de programmation permettant d'obtenir des informations comme l'application courante, les documents ouverts ... Généralement, les interfaces de ces fonctions sont écrites en C, obligeant les appels à ces fonctions à être également écrits en C. C'est par exemple le cas pour *Carbon* sous *MacOS*[19] ou *Win32* sous *Windows*. La liaison entre ces appels C et le module *Collection*, écrit en Java comme le reste de *CommanderMo*, est réalisée grâce à la technologie *JNI*[20] (*Java Native Interface*), qui permet à un programme Java d'appeler des méthodes C/C++ et de récupérer les résultats en tant que données Java. Cette technologie est fournie en standard avec l'environnement de développement Java (le *JDK* — *Java Developer Kit*), et ne nécessite aucune bibliothèque spécifique pour pouvoir être utilisée avec une machine virtuelle Java standard.

Le module *Agrégation* utilise, comme indiqué dans le chapitre « Contextualisation », des réseaux bayesiens pour réaliser l'agrégation des informations bas niveau collectées en informations haut niveau. Ces réseaux bayesiens sont implantés par la bibliothèque *EmBayes*[21], une bibliothèque Java libre et *open-source* de réseaux bayesiens. Cette bibliothèque a pour particularité d'avoir été spécifiquement conçue pour être embarquée dans des appareils petits et portables, comme des *PDA*. Les réseaux bayesiens fournis ont ainsi l'avantage d'être rapides et peu gourmands en ressources. La configuration de ces réseaux se fait à l'aide de fichiers *XML* décrivant le réseau lui-même, ses nœuds, ainsi que

[19]Le cas de *Cocoa* sous *MacOS X* est particulier, car les interfaces de *Cocoa* sont écrites en Objective-C et en Java. Il est donc possible d'accéder directement aux interfaces de *MacOS X* à partir de *CommanderMo*.

[20]`http://java.sun.com/j2se/1.3/docs/guide/jni/`

[21]`http://www.pmr.poli.usp.br/ltd/Software/javabayes/`

les probabilités conditionnelles attachées à chaque nœud. La bibliothèque *EmBayes* est disponible librement sur Internet sous une licence libre et *open-source*, la licence *LGPL* (*Lesser GNU Public License*, auparavant connue comme la *Library GNU Public License*), qui est compatible avec la licence *MIT* de *CommanderMo*.

Les modules *Filtrage* et *Présentation* utilisent des réseaux neuronaux pour assigner des niveaux respectivement de détail et d'intrusion aux informations qu'ils traitent. Ces réseaux neuronaux sont fournis par la bibliothèque *Joone*[22], une bibliothèque Java libre et *open-source* de réseaux neuronaux. Cette bibliothèque a l'avantage de fournir des réseaux neuronaux configurables, efficaces et peu gourmands en ressources. Les réseaux neuronaux sont spécifiés par des fichiers de configuration *XML*, décrivant le réseau et ses nœuds. La bibliothèque *Joone* est disponible librement sur Internet sous la même licence que la bibliothèque *EmBayes*, c'est-à-dire la licence *LGPL*.

L'entraînement initial des réseaux neuronaux peut se faire de deux manières. Il est possible d'assigner à chaque réseau un fichier d'entraînement, contenant des valeurs d'entrées spécifiques pour lesquelles les valeurs de sorties sont connues. Lors de son premier lancement, juste après l'installation de *CommanderMo*, le réseau va utiliser ce fichier pour s'entraîner. Cette solution a l'avantage de ne pas demander de travail de configuration spécifique. Par contre, elle ralentit le premier lancement du logiciel, car l'entraînement des réseaux neuronaux peut prendre un certain temps.

Une deuxième solution est d'utiliser le mécanisme de sauvegarde proposé par la bibliothèque *Joone*. En effet, un réseau neuronal ne cesse d'apprendre tout au long de son utilisation. Il faut que cet apprentissage soit sauvegardé d'un lancement du logiciel à l'autre. La bibliothèque *Joone* propose justement un tel mécanisme, qui permet d'enregistrer dans le même fichier le réseau neuronal et son apprentissage. La deuxième solution est donc de réaliser un entraînement initial plus ou moins générique dans la phase de configuration, de sauver le réseau neuronal avec l'apprentissage obtenu, et d'installer le fichier résultant comme fichier de configuration. Cette solution demande plus de travail dans la phase de configuration, mais permet un premier lancement du logiciel plus rapide.

Quelle que soit la solution choisie, les apprentissages des différents réseaux neuronaux sont toujours sauvegardés entre deux exécutions du logiciel. En effet, lorsque l'utilisateur modifie le résultat d'un réseau neuronal, ce dernier prend en compte le changement et s'adapte aux préférences de l'utilisateur. Cet apprentissage est sauvegardé lorsque l'utilisateur quitte le logiciel, afin que le réseau neuronal reste adapté aux préférences de l'utilisateur au prochain lancement de *CommanderMo*. Le travail fait par l'utilisateur pour indiquer ses choix aux réseaux neuronaux n'est pas perdu lorsqu'il quitte le logiciel.

La diffusion d'informations entre les modules *Diffusion* et *Réception* des différentes instances de *CommanderMo* se fait à travers un réseau d'égal à égal, fourni par la technologie *JXTA*[23]. Il s'agit d'un projet de Sun, le créateur de Java, qui offre une implantation libre, *open-source* et surtout standardisée d'un réseau d'égal à égal en Java. Cette technologie, composée d'un ensemble de bibliothèques Java, prend en charge à peu près tout ce qui concerne le réseau, depuis la découverte des autres nœuds du réseau jusqu'à l'émission et la réception des messages. En particulier, elle prend en charge la sécurisation des

[22] `http://www.joone.org/`
[23] `http://www.jxta.org/`

communications par chiffrement, ainsi que l'authentification des utilisateurs par mot de passe. La configuration de *JXTA* se fait soit dynamiquement par des fenêtres de dialogue lors de la première utilisation, après l'installation, soit par différents fichiers de configuration. La technologie *JXTA* est disponible librement sur Internet sous une licence libre et *open-source*, la licence *Apache*, compatible avec la licence *MIT* de *CommanderMo* et avec la licence *LGPL* des bibliothèques *EmBayes* et *Joone*.

Le module *Visualisation* propose initialement trois sous-modules de représentation des informations. D'autres modules peuvent être développés par l'utilisateur et ajoutés dynamiquement à *CommanderMo*. Les trois modules de base sont un module audio, basé sur les fonctionnalités audio de Java, et deux modules de visualisation, un module *Treemap* et un module *Hypertree*. Ces deux modules utilisent respectivement les bibliothèques *Treemap Java Library* et *Hypertree Java Library*, développées dans le cadre de cette thèse et décrites dans la suite du chapitre. Ces deux bibliothèques sont disponibles librement sur Internet sous la même licence libre et *open-source* que *CommanderMo*, c'est-à-dire la licence *MIT*.

Finalement, le module *Contexte* sauve les informations reçues dans des fichiers *XML* à l'aide de la bibliothèque *Xerces*[24], une bibliothèque libre et *open-source* permettant de lire et d'écrire facilement des fichiers *XML* en Java. Utiliser des fichiers *XML* pour sauver les informations reçues apporte plusieurs avantages. Tout d'abord, c'est un format directement lisible par l'utilisateur (contrairement à un format binaire), ce qui est nécessaire pour le respect de la vie privée de l'utilisateur. En effet, il a été vu précédemment que ce dernier doit pouvoir non seulement accéder aux informations stockées par l'outil de conscience de groupe et le concernant mais aussi les modifier. Ensuite, c'est un format standard, ce qui rend l'outil interopérable avec d'autres outils de conscience de groupe. Cela simplifie aussi l'accès aux informations pour les sous-modules de visualisation. Enfin, cela permet d'utiliser l'ensemble des techniques d'accès et de recherche d'informations disponibles pour le format *XML*, comme *XPath* ou *XQuery*. Finalement, un tel fichier se prête bien à une compression au vol, ce qui permet d'économiser la place disque. Ces deux dernières possibilités ne sont pour l'instant pas implantées dans la version courante de *CommanderMo*. La bibliothèque *Xerces* est disponible librement sur Internet sous la même licence que la technologie *JXTA*, c'est-à-dire la licence *Apache*.

1.2 Installation et configuration

CommanderMo est en majorité écrit en Java, ce qui le rend multi-plateforme. La seule partie de *CommanderMo* à être dépendante du système d'exploitation est la partie des appels systèmes du module *Collection*. Ces appels systèmes doivent être réécrits pour chaque système d'exploitation. Cependant, les informations bas niveau demandées sont assez génériques pour être disponibles sur tous les systèmes d'exploitation. *CommanderMo* peut ainsi être installé sur *MacOS*, *Windows*, *Linux*, mais aussi sur des *PDA*.

CommanderMo est un outil générique qui peut s'adapter à n'importe quelle situation de coopération, quelles que soit les applications utilisées par les membres du groupe pour coopérer. Pour adapter l'outil à une situation de coopération particulière, il suffit de

[24] `http://xml.apache.org/xerces-j/`

configurer de manière adéquate les réseaux bayesiens du module *Agrégation*. En effet, c'est dans ces réseaux que les informations bas niveau, comme le nom de l'application courante ou bien le type de l'activité de l'utilisateur, propres à la situation de coopération, vont être utilisées. Le reste de l'outil utilise des informations haut niveau. En particulier, l'entraînement initial des réseaux neuronaux des modules *Filtrage* et *Présentation* est basé sur des règles génériques. C'est ensuite l'utilisateur qui entraîne spécifiquement les réseaux neuronaux à sa situation, en fonction de ses choix.

L'installation de *CommanderMo* est très simple. Le logiciel se présente sous la forme d'un répertoire contenant les différentes bibliothèques nécessaires, ainsi que quelques utilitaires facilitant le lancement de l'application. Il doit être installé sur les systèmes de toutes les personnes souhaitant participer au partage d'informations. Grâce à son réseau d'égal à égal, *CommanderMo* ne nécessite pas la mise en place d'un serveur dédié.

En ce qui concerne la configuration, elle se fait à l'aide de fichiers de configuration présents dans le répertoire. Ces fichiers contiennent la spécification des réseaux bayesiens, les réseaux de neurones pré-entraînés, la configuration réseau du poste, ainsi que la configuration *JXTA* de l'utilisateur.

La phase de configuration est une phase importante qui doit être faite après analyse de la coopération afin de trouver les informations nécessaires comme la structure du groupe, le nom des participants et leur place dans le groupe, le type de coopération, les outils utilisés, les types d'artefacts manipulés ... La surcharge de travail induite par cette phase de configuration est limitée par le fait qu'elle ne doit être faite qu'une seule fois, avant la première utilisation du logiciel. De plus, ses résultats sont généralement valables pour toute une partie du groupe, voire tout le groupe, suivant la diversité des rôles et des activités des participants.

1.3 Utilisation

Pour être utilisé efficacement, *CommanderMo* doit être lancé dès le début de l'activité coopérative. En fait, il vaut même mieux le lancer automatiquement dès le démarrage du système. Cela permet d'avoir plus d'informations sur le contexte de travail de l'utilisateur. De plus, *CommanderMo* peut être utile même en dehors d'une activité coopérative. En effet, il donne accès à une trace des activités réalisées, ce qui peut permettre par exemple de dresser un bilan de la journée de travail, ou bien de reprendre plus facilement son activité après une interruption.

L'interface graphique de *CommanderMo* comprend un certain nombre de fenêtres. La première fenêtre de *CommanderMo* est sa fenêtre principale, qui permet de gérer le fonctionnement global de l'outil, comme sa configuration, la liste des différentes fenêtres ouvertes, ainsi que les modes de présentation des messages.

Le nombre de fenêtres et l'apparence générale de l'interface graphique est variable. Elle dépend des sous-modules de visualisations installés ainsi que des modes de confiance choisis pour la présentation des résultats des modules *Filtrage* et *Présentation*. En effet, dans ces deux modules, les résultats des réseaux neuronaux peuvent être présentés à l'utilisateur afin que ce dernier puisse les corriger et ainsi indiquer ses préférences. Du choix d'un tel mode dépend donc la présence d'une fenêtre spécifique pour la présentation des résultats. De même, chaque module de visualisation affiche sa ou ses fenêtres spécifiques.

Il existe deux types de fenêtres permettant de présenter les résultats des réseaux neuronaux : une fenêtre pour les résultats du module *Filtrage*, et une pour ceux du module *Présentation*. La première affiche les messages prêts à être envoyés après leur filtrage par le module *Filtrage*. L'utilisateur peut y voir le niveau de détail de chaque catégorie de l'information filtrée, l'information elle-même ainsi que le destinataire de l'information. Il peut choisir de modifier un ou plusieurs niveaux de détail, ainsi que d'envoyer ou pas le message. Dans la deuxième fenêtre sont affichées les informations reçues ainsi que le niveau d'intrusion qui leur a été attribué par le module *Présentation*. L'utilisateur a alors la possibilité de changer ce niveau d'intrusion. Dans les deux fenêtres, toute modification est transmise au réseau neuronal correspondant qui s'adapte alors au choix de l'utilisateur.

En ce qui concerne les sous-modules de visualisation, deux modules graphiques sont proposés en standard : le module de visualisation *Treemap* et le module de visualisation *Hypertree*.

Le module *Treemap* ouvre une fenêtre contenant une visualisation *treemap* représentant une vue globale des informations reçues. Chaque membre du groupe est représenté par un rectangle, dont la couleur indique le type de la tâche en cours. Le passage du pointeur de la souris sur un rectangle affiche une bulle d'aide contenant des détails sur l'activité en cours de la personne. Un clic sur le rectangle affiche la vue de navigation pour les informations de cette personne. Ce module est normalement utilisé par l'utilisateur pour avoir une vue globale de l'activité du groupe et pour pouvoir être notifié en cas de changement dans l'activité d'un membre.

Le module *Hypertree* ouvre une fenêtre contenant une visualisation *hypertree* utilisée pour la navigation dans les informations reçues. Les informations peuvent être classées suivant plusieurs hiérarchies : suivant une hiérarchie temporelle (mois-semaine-jour-heure-minute) par exemple, ou bien suivant la structure hiérarchique du groupe. Dans ce cas, les informations sont d'abord classées suivant leur expéditeur, avant d'être classées par date d'arrivée. Chaque membre du groupe est alors représenté par un nœud, contenant lui-même des nœuds représentant les informations haut niveau reçues de cette personne. Un clic sur un nœud représentant une information ouvre la vue en détail de cette information, c'est-à-dire une liste des valeurs de chacune des catégories de l'information sélectionnée. Ce module est normalement utilisé par l'utilisateur pour naviguer dans les informations reçues.

1.4 Conclusion

CommanderMo est un outil générique de conscience de groupe. Il est téléchargeable librement sur Internet avec son code source sous une licence libre et *open-source*, la licence *MIT*. Toutes les bibliothèques externes utilisées sont elles aussi téléchargeables librement sur Internet sous des licences libres et *open-source* compatibles avec la licence *MIT* (*LGPL* pour *EmBayes* et *Joone*, *Apache* pour *JXTA* et *Xerces*, *MIT* pour *TreeMap Java Library* et *HyperTree Java Library*).

CommanderMo est facile à installer, et ne nécessite pas la mise en place d'un serveur central. De plus, il est quasiment multi-plateforme, car écrit principalement en Java, avec seulement quelques appels en C vers le système d'exploitation. Ces appels peuvent facilement être réécrits car ils font appel à des fonctionnalités présentes sur n'importe quel

système d'exploitation. *CommanderMo* peut ainsi être porté facilement sur n'importe quelle plateforme.

La partie la plus difficile dans l'installation de *CommanderMo* correspond à la phase de configuration, pendant laquelle il faut spécifier les réseaux bayesiens du module *Agrégation* en fonction du type de la situation de coopération. Cela ne doit être fait qu'une seule fois, avant la première utilisation du logiciel.

Une fois installé, *CommanderMo* collecte, diffuse et représente les informations de conscience de groupe d'une manière adaptée au contexte de travail de l'utilisateur.

2 Bibliothèques de visualisation

TreeMap Java Library et *HyperTree Java Library* sont deux bibliothèques Java permettant de créer rapidement et facilement des visualisations *treemap* et *hypertree* de données arborescentes fournies par l'utilisateur.

Le point fort de ces deux bibliothèques est leur réutilisabilité. En premier lieu, les deux bibliothèques sont développées en Java, ce qui les rend multi-plateformes. De plus, elles sont fournies sous la forme d'un simple fichier « .jar », ce qui les rend faciles à utiliser et à intégrer dans un projet existant. Les visualisations retournées par ces bibliothèques sont fournies sous la forme d'un composant graphique (un objet dérivant de la classe `java.awt.Component`), ce qui les rend très facilement intégrables à n'importe quelle interface graphique Java. Enfin, de nombreuses options de configuration sont disponibles pour adapter les vues aux besoins des utilisateurs. Finalement, ces deux bibliothèques sont disponibles sous la même licence libre et *open-source* que *CommanderMo*, la licence *MIT*. Cela permet à toute personne le souhaitant de les réutiliser, les adapter et les améliorer, directement au niveau du code source.

Les données devant être visualisées peuvent être fournies soit dans un fichier (différents formats sont disponibles), soit directement sous la forme d'un arbre d'objet. Pour cela, il suffit que les objets représentant les données implémentent une interface spécifique (`TMNode` pour la visualisation *treemap*, `HTNode` pour la visualisation *hypertree*). Cette interface est très semblable à l'interface `javax.swing.tree.TreeNode`, disponible dans la bibliothèque graphique *Swing*[25] de Java. Cette interface est notamment utilisée pour obtenir une visualisation *JTree*, une représentation arborescente classique comme celle que l'on trouve dans les explorateurs de fichiers. Cette similarité permet à une application qui a déjà dans son interface graphique une représentation *JTree* de ses données, d'avoir facilement et rapidement des visualisations *treemap* et *hypertree* de ces mêmes données.

Une fois que les données ont été fournies, elles sont stockées dans un modèle. L'utilisateur peut ensuite demander autant de vues qu'il le souhaite de ces données. Chaque vue reste synchronisée avec les données du modèle : toute modification des données est automatiquement reportée dans toutes les vues. Cependant, chaque vue peut être paramétrée différemment. Cela permet par exemple d'observer différents critères ou bien différentes parties du même ensemble de données.

[25]Depuis la version 1.2 de Java, toute machine virtuelle Java est fournie entre autre avec deux bibliothèques graphiques, l'*AWT* et *Swing*. Ces deux bibliothèques fournissent les objets nécessaires à la création d'interfaces graphique en Java.

Chaque bibliothèque est fournie avec une application de démonstration, permettant de tester les différentes fonctionnalités de la visualisation. Cette application est, pour les deux bibliothèques, un visualisateur de système de fichiers, qui prend en paramètre un chemin vers un répertoire sur le système de fichiers de l'ordinateur. Le répertoire et ses sous-répertoires sont parcourus et représentés par la visualisation en question. L'application de démonstration n'est pas très puissante (elle ne permet pas d'interaction avec les fichiers, comme un déplacement par exemple), mais elle remplit son rôle, à savoir montrer les différentes fonctionnalités de la visualisation. Elle donne de plus un bon exemple du code nécessaire pour l'utilisation et l'intégration de la bibliothèque.

2.1 TreeMap Java Library

TreeMap Java Library est, comme son nom l'indique, la bibliothèque en charge des visualisations *treemap*. Elle implante les algorithmes de découpage et de représentation décrits dans le chapitre « Visualisation », comme l'algorithme *slice-and-dice*, l'algorithme *squarified*, la représentation *cushion* ...

Son architecture globale est décrite dans la figure 2. Les deux principales classes à connaître et à utiliser sont les classes `TreeMap`, qui représente le modèle de données à partir duquel les vues sont créées, et `TMView`, qui représente la vue *treemap*, c'est-à-dire le composant graphique.

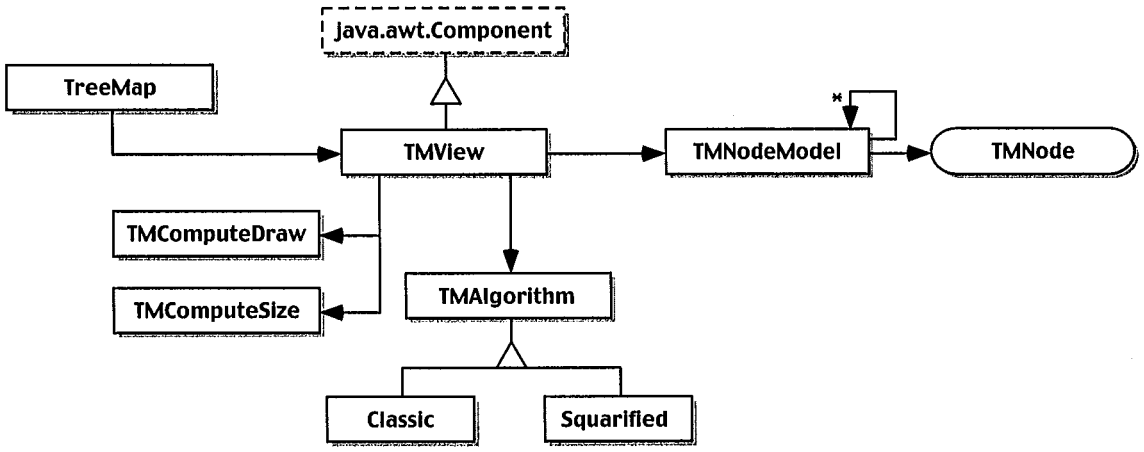

FIG. 2 – L'architecture de la *TreeMap Java Library*.

La première classe à instancier pour pouvoir obtenir une vue *treemap* est la classe `TreeMap`. Pour instancier cette classe, il faut lui passer en paramètre les données à représenter. Ces données sont fournies sous la forme d'un arbre d'objets. Chaque objet de cet arbre doit implémenter les méthodes de l'interface `TMNode`. Une fois la classe `TreeMap` instanciée avec les données correspondantes, l'utilisateur peut obtenir autant de vues *treemap* qu'il le souhaite sur ces données. Pour cela, l'utilisateur appelle la méthode `getView()`, et obtient en retour un objet `TMView`.

La classe `TMView` dérive de la classe `java.awt.Component`, ce qui signifie que ses

instances sont des composants graphiques capable de s'intégrer de manière standard dans toute interface graphique Java. L'objet `TMView` représente donc le composant graphique contenant la vue *treemap*. Il est en charge de la construction et de l'affichage du *treemap*, et est aidé dans sa tâche par plusieurs autres objets.

Premièrement, l'objet `TMView` n'accède pas directement aux objets `TMNode`, contenant les données à représenter. Il passe par des objets `TMNodeModel` qui encapsulent les objets `TMNode` en leur ajoutant des attributs correspondant aux différentes caractéristiques du rectangle représentant le nœud en question, comme par exemple ses coordonnées, sa taille ou sa couleur. Ainsi, à chaque rectangle du *treemap* correspond un objet `TMNodeModel` contenant l'objet `TMNode` représenté. Les objets `TMNodeModel` forment entre eux une hiérarchie similaire à celle existante entre les objets `TMNode`.

Certains objets utilisés par l'objet `TMView` permettent de paramétrer la visualisation *treemap*. C'est le cas des objets implémentant les interfaces `TMComputeDraw` et `TMComputeSize`. Ces deux interfaces permettent de spécifier l'algorithme à utiliser pour calculer la taille et la couleur de remplissage des rectangles de la vue *treemap*, en fonction des données des objets `TMNode`. Ces deux objets doivent forcément être définis par l'utilisateur, car ils dépendent complètement de la structure des objets utilisateurs implémentant l'interface `TMNode`. Les objets `TMComputeDraw` et `TMComputeSize` peuvent être changés en cours d'exécution, la vue étant alors dynamiquement mise à jour. Cela permet de visualiser différentes caractéristiques en passant dynamiquement d'une vue à l'autre.

Le dernier objet utilisé par l'objet `TMView` est une instance d'une sous-classe de la classe `TMAlgorithm`. La classe `TMAlgorithm` est une classe abstraite permettant l'implantation d'un algorithme de découpage. Tous les algorithmes de découpage sont spécifiés dans des classes dérivant de cette classe, qui fournit le cadre et les fonctionnalités nécessaires. Elle fournit en particulier les algorithmes de dessin, comme l'algorithme *cushion*. Ces algorithmes de dessin étant indépendants du type de l'algorithme de découpage, ils peuvent être appliqués quel que soit l'algorithme de découpage choisi.

Chaque algorithme est implanté sous la forme d'une sous-classe de la classe `TMAlgorithm`. Pour choisir un algorithme, il suffit de passer l'instance de la classe correspondante à l'objet `TMView`. Comme pour les objets `TMComputeDraw` et `TMComputeSize`, l'algorithme de découpage ainsi que l'algorithme de dessin peuvent tous les deux être changés en cours d'exécution, et la vue se met alors dynamiquement à jour.

Dans l'implantation courante, deux algorithmes de découpage et deux algorithmes de dessin sont fournis. Pour le découpage, il s'agit de l'algorithme originel de découpage (*slice-and-dice*), et de l'algorithme *squarified*, où les nœuds sont représentés non plus par des rectangles mais par des carrés. Pour le dessin, il s'agit de l'algorithme classique et de l'algorithme *cushion*, donnant un effet *3D* aux rectangles.

2.2 HyperTree Java Library

HyperTree Java Library est, comme son nom l'indique, la bibliothèque en charge des visualisations *hypertree*. Elle implante les algorithmes de placement et d'interaction décrits dans le chapitre « Visualisation ».

Son architecture globale est décrite dans la figure 3. Comme pour la *TreeMap Java Library*, les deux principales classes à connaître et à utiliser sont les classes `HyperTree`,

qui représente le modèle de données à partir duquel les vues sont créées, et `HTView`, qui représente la vue *hypertree*, c'est-à-dire le composant graphique. Ces deux classes ont le même rôle et s'utilisent de la même façon que les classes `TreeMap` et `TMView` dans la *TreeMap Java Library*.

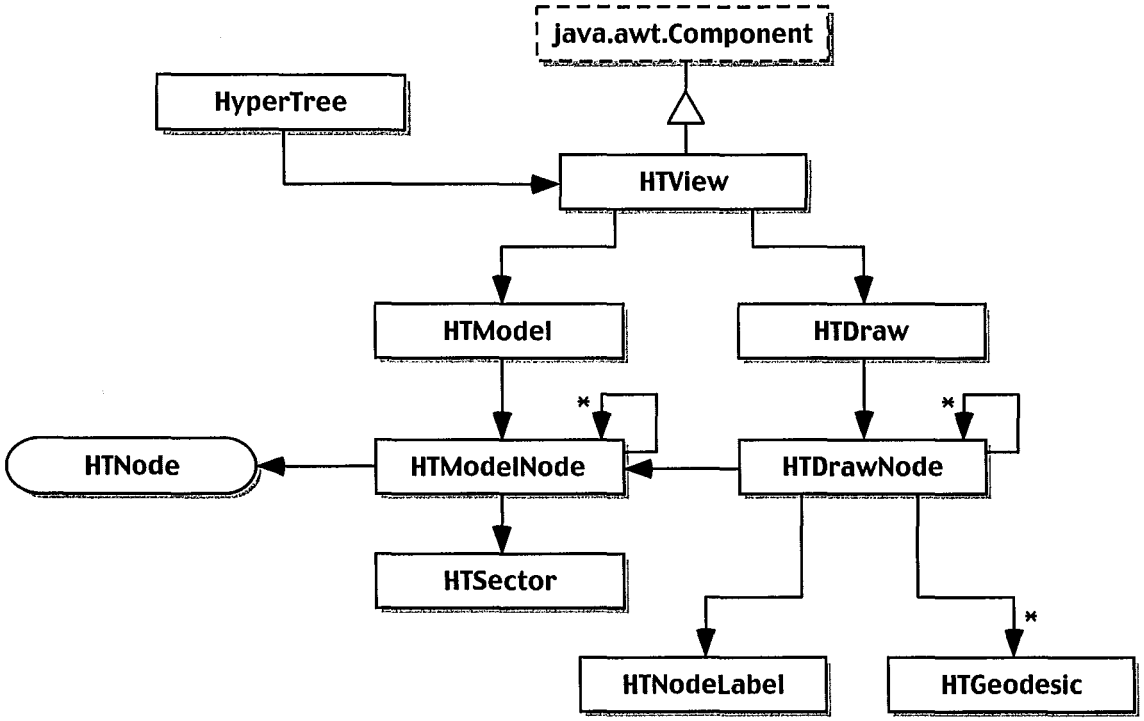

FIG. 3 – L'architecture de l'*HyperTree Java Library*.

La classe `HyperTree` est la première classe à instancier pour pouvoir obtenir une vue *hypertree*. Elle prend en paramètre les données à représenter sous la forme d'un arbre d'objets implémentant les méthodes de l'interface `HTNode`, elle-même très similaire à l'interface `TMNode`. Une fois la classe `HyperTree` instanciée avec les données correspondantes, l'utilisateur peut obtenir autant de vues *hypertree* sur ces données qu'il le souhaite en appelant la méthode `getView()`, qui retourne un objet `HTView`.

La classe `HTView` dérive de la classe `java.awt.Component`. L'objet `HTView` représente donc le composant graphique contenant la vue *hypertree*, et est en charge de la construction et de l'affichage de l'*hypertree*. En raison des différents calculs à réaliser, la classe `HTView` utilise deux modèles différents pour ses données. Ces deux modèles sont représentés par les classes `HTModel` et `HTDraw`.

La classe `HTModel` contient le modèle statique des données à représenter. Elle a pour responsabilité de placer l'ensemble des données, représentées par les objets `HTNode`, dans l'espace hyperbolique, et de conserver les coordonnées ainsi trouvées. Pour cela, elle passe par des objets `HTModelNode` qui encapsulent les objets `HTNode` en leur ajoutant des attributs correspondant aux coordonnées initiales des nœuds dans l'espace hyperbolique. Les objets `HTModelNode` forment entre eux une hiérarchie similaire à celle des objets `HTNode`.

Ils encapsulent un objet `HTNode`, et contiennent les coordonnées cartésiennes du nœud correspondant. Ils contiennent aussi un lien vers un objet `HTSector` qui représente le secteur angulaire alloué au nœud correspondant, et qui est utilisé au moment du placement de l'arbre dans l'espace hyperbolique.

La classe `HTDraw` contient le modèle dynamique des données à représenter. Elle est responsable des coordonnées de l'arbre en mouvement. Elle utilise pour cela une hiérarchie d'objets `HTDrawNode`, qui encapsulent les objets `HTModelNode` en leur ajoutant des attributs correspondant aux coordonnées courantes du nœud en question. Ces coordonnées sont différentes de celles contenues dans l'objet `HTModelNode` si l'arbre a bougé. L'objet `HTDrawNode` contient aussi un lien vers un objet `HTNodeLabel`, chargé de la représentation du nœud proprement dit (par exemple par un rectangle coloré avec du texte dans l'implantation courante), ainsi que des liens vers un certain nombre d'objets `HTGeodesic`, en charge de la représentation des géodésiques, c'est-à-dire des branches de l'arbre hyperbolique. Les objets `HTDrawNode` forment une hiérarchie similaire à celle des objets `HTModelNode` et à celle des objets `HTNode`.

La classe `HTGeodesic` contient le code nécessaire au tracé des géodésiques. Ces branches peuvent être soit des segments de droite, soit des arcs de cercles orthogonaux au cercle unité. Le tracé des arcs de cercle se fait soit en utilisant la classe `QuadCurve2D` du *package* `java.awt.geom` si celle-ci est disponible[26], soit avec une approximation par segments de droite pour les machines virtuelles Java plus anciennes. Ce choix permet d'allier performance et portabilité.

La classe `HTView` utilise deux modèles, l'un contenant les coordonnées initiales des nœuds et l'autre leurs coordonnées en mouvement, pour une raison pratique tenant au calcul des transformations dans l'espace hyperbolique. En effet, il a été vu dans le chapitre « Visualisation » qu'une solution pour éviter la rotation induite par la composition de translations hyperboliques était de toujours effectuer les translations à partir des coordonnées initiales de l'arbre, lorsque sa racine se trouve au centre du cercle unité. Une solution pratique pour cela est de conserver les coordonnées initiales dans un premier modèle, et les coordonnées en mouvement dans un second. Un avantage additionnel de cette solution est que la propagation des erreurs dues aux imprécisions de calcul est limitée car les calculs sont toujours faits à partir des coordonnées initiales. Enfin, cela permet de séparer clairement les données du modèle hyperbolique (`HTModel`, `HTSector`) des données utilisées pour le dessin (`HTDraw`, `HTNodeLabel`, `HTGeodesic`).

2.3 Conclusion

TreeMap Java Library et *HyperTree Java Library* sont deux bibliothèques Java de visualisation permettant d'intégrer facilement et rapidement à une interface graphique deux visualisations peu utilisées et pourtant très utiles, la visualisation *treemap* et la visualisation *hypertree*. Elles sont disponibles librement sur Internet sous une licence libre et *open-source*, la licence *MIT*.

Ces deux bibliothèques ont une architecture qui leur permet d'être facilement modifiées et adaptées aux besoins des développeurs. Cela a déjà donné lieu, depuis leurs publications

[26] La classe `java.awt.geom.QuadCurve2D` n'est disponible qu'à partir de la version 1.2 de Java.

sur Internet, à un certain nombre de réutilisations, améliorations et portages divers, non seulement par la communauté *open-source* mais aussi par certaines entreprises, grâce notamment à la liberté accordée par la licence *MIT*. Ces deux bibliothèques aident ainsi à faire connaître au public d'autres visualisations que celles habituellement trouvées dans les bibliothèques d'interfaces graphiques.

Bilans et perspectives

Une des difficultés principales du travail coopératif à distance réside dans le manque de *conscience de groupe.* Lorsque des personnes travaillent en même temps et dans le même lieu, elles ont conscience de leur appartenance au groupe et de leur place dans ce groupe. Ce sentiment, qui permet notamment une meilleure coordination entre les membres du groupe, est fortement diminué lorsque ces personnes sont distribuées dans le temps et/ou dans l'espace.

La conscience de groupe émerge de l'ensemble des informations, formelles et informelles, qui sont échangées entre les membres du groupe, de manière intentionnelle ou non. Lorsque les personnes ne sont plus ensemble en même temps, ces informations ne sont plus échangées, et la conscience de groupe est quasiment inexistante. Le but des outils de conscience de groupe est de permettre aux personnes distribuées dans le temps et l'espace de s'échanger à nouveau ces informations, d'une manière ou d'une autre.

Cette thèse aborde le problème de la surcharge d'informations dans les outils de conscience de groupe : comment trouver un compromis entre fournir assez d'informations pour permettre la construction de la conscience de groupe, mais pas trop pour ne pas surcharger l'utilisateur. La première proposition de cette thèse est de replacer les informations dans leur contexte, et d'utiliser ce contexte pour adapter et visualiser la conscience de groupe, ceci afin de rendre les informations transmises plus pertinentes et moins intrusives. La deuxième proposition est d'utiliser les visualisations *treemap* et *hypertree* pour représenter les informations de la conscience de groupe, afin d'adapter la représentation suivant la tâche à accomplir. La suite de ce chapitre détaille le bilan et les perspectives de ces deux propositions.

1 Bilan

La première contribution de cette thèse est un modèle cognitif des mécanismes de construction de la conscience de groupe. Ce modèle découle de l'analyse cognitive de la conscience de groupe à l'aide de plusieurs résultats, outils et cadres de travail de la psychologie cognitive comme la Gestalt, la théorie de l'Activité ou la théorie de la Cognition Distribuée. Dans ce modèle, la construction de la conscience de groupe se fait en trois grandes étapes :

Perception des informations. Les informations permettant de construire la conscience de groupe proviennent du contexte des activités des membres du groupe. Ces informations peuvent être classifiées suivant les six questions habituelles : *qui*, *quoi*, *quand*, *où*, *pourquoi*, *comment.* Elles sont transmises par différents canaux, chacun

ayant ses propriétés et son influence sur la communication, elle-même pouvant être de plusieurs types.

Construction de la conscience de groupe. Une fois les informations perçues, la conscience de groupe est construite en quatre étapes :
- Agrégation des informations élémentaires en informations complexes ;
- Filtrage des informations complexes ;
- Interprétation des informations ayant passé le filtrage ;
- Internalisation des informations interprétées dans le contexte.

Action de l'utilisateur. Enfin, lorsque la conscience de groupe est construite, l'utilisateur peut avoir deux types d'action. Il peut utiliser la conscience de groupe pour améliorer sa coordination avec les autres membres, ou bien il peut activement rechercher une information pour améliorer sa conscience de groupe.

L'analyse de la place de l'outil de conscience de groupe a aussi permis de spécifier un certain nombre de prérequis pour l'intégration d'un outil de conscience de groupe. L'outil doit ainsi :
- collecter les informations sur l'activité de l'utilisateur ;
- propager ces informations aux autres outils ;
- recevoir ces informations ;
- présenter ces informations à l'utilisateur ;
- présenter à l'utilisateur des informations non seulement sur les activités des autres membres, mais aussi sur les activités propres de l'utilisateur ;
- présenter les informations suivant deux modes :
  - un mode périphérique, pour présenter les informations globales ;
  - un mode actif, pour présenter, à la demande de l'utilisateur et de manière simple, le détail d'une information spécifique ;
- permettre à l'utilisateur de définir, pour chaque membre du groupe, son niveau d'accès aux différentes informations de son activité ;
- être dynamique et gérer les informations nouvelles ou changeantes.

De ce modèle et de ces prérequis est déduite une architecture fonctionnelle permettant à un outil de conscience de groupe de s'adapter au contexte de travail de l'utilisateur. Le but de cette adaptation est de rendre les informations fournies à l'utilisateur plus pertinentes, moins intrusives, et de les présenter d'une manière plus adaptée à son activité. Cette architecture fonctionnelle est divisée en sept niveaux :
- le premier niveau du système collecte les informations de bas niveau sur l'activité de l'utilisateur ;
- le second niveau agrège les informations de bas niveau en informations de haut niveau ; ces informations haut niveau sont envoyées au niveau suivant pour être diffusées, ainsi qu'au cinquième niveau de l'outil pour être représentées directement à l'utilisateur ;
- le troisième niveau filtre les informations non pertinentes et personnelles en fonction du contexte et des niveaux d'accès définis par l'utilisateur : les informations jugées pertinentes et diffusables sont envoyées au niveau suivant ;
- le quatrième niveau diffuse les informations aux autres outils ;
- le cinquième niveau est chargé de recevoir les informations envoyées par les outils

des autres membres du groupe mais aussi par le second niveau : les informations reçues sont sauvées dans le contexte local, puis envoyées au niveau suivant ;
- le sixième niveau décide du moment et de la façon dont l'information va être représentée en fonction du contexte et des préférences de l'utilisateur ;
- le dernier niveau présente l'information à l'utilisateur. Ce niveau est aussi chargé des interactions avec l'utilisateur. La présentation des informations peut se faire suivant au moins deux modes : un mode périphérique et dynamique, présentant les changements et les nouvelles informations, et un mode actif permettant à l'utilisateur de rechercher une information particulière.

Ces sept niveaux sont répartis en deux parties, une partie émettrice et une partie réceptrice. La partie émettrice collecte les informations, les agrège en informations de haut niveau, filtre ces informations haut niveau et les diffuse. La partie réceptrice reçoit les informations haut niveau diffusées, les adapte à l'activité de l'utilisateur et les lui présente.

Un outil de conscience de groupe basé sur cette architecture est finalement décrit. L'outil utilise le contexte de travail de l'utilisateur pour construire et adapter la conscience de groupe pour les membres du groupe. L'outil récupère et agrège les événements de bas niveau à l'aide de réseaux bayesiens de façon à reconstruire les informations sémantiques de haut niveau correspondantes. Ces informations sont alors utilisées pour adapter, à l'aide de réseaux de neurones, la propagation et la représentation de la conscience de groupe. En adaptant la propagation des informations de la conscience de groupe, l'outil filtre ces informations. Ainsi, seules les informations non-personnelles sont envoyées, et seules les informations utiles et pertinentes pour l'utilisateur sont reçues. Enfin, leur représentation est adaptée pour que le système soit aussi peu intrusif que possible, tout en faisant en sorte que les informations importantes ne puissent être manquées.

Comparé aux autres travaux existants, le système proposé construit et apporte une conscience de groupe qui est adaptée au contexte de travail de l'utilisateur. Ainsi, les informations de la conscience de groupe sont plus utiles, plus pertinentes et moins intrusives pour l'utilisateur. La modélisation du contexte d'un utilisateur permet en effet :
- de recueillir et diffuser assez d'informations aux autres membres du groupe afin que ceux-ci puissent construire leur conscience de groupe ;
- de déterminer, pour une information donnée, quelles sont les personnes intéressées et qui y ont accès ;
- de déterminer, suivant l'activité en cours, comment notifier l'utilisateur d'un changement de la manière la moins intrusive possible.

L'utilisation de réseaux bayesiens apporte une solution efficace pour l'agrégation d'événements bas niveau en information haut niveau. L'utilisation de réseaux neuronaux pour l'adaptation permet d'avoir un outil qui s'adapte automatiquement aux préférences de l'utilisateur au fur et à mesure de son utilisation.

Dans la dernière étape de cette architecture, l'étape de présentation, les informations de conscience de groupe sont présentées à l'utilisateur à l'aide d'un ou plusieurs types de visualisations, chacun réalisé par un module spécifique. L'analyse cognitive de la conscience de groupe a permis de définir un certain nombre de propriétés que doivent avoir les visualisations utilisées par l'outil. Certaines visualisations sont donc plus adaptées que d'autres pour représenter les informations de conscience de groupe.

La deuxième contribution de cette thèse est la réalisation, l'amélioration et l'utilisation de deux techniques de visualisation particulières, les visualisations *treemap* et *hypertree*, pour représenter les informations de la conscience de groupe. Ces deux visualisations sont dédiées à la représentation de données arborescentes, ce qui correspond au type des données à représenter. De plus, chacune est spécialisée dans une tâche spécifique : la visualisation *treemap* est excellente pour avoir une vue globale, et la visualisation *hypertree* est excellente pour la navigation. Ces deux visualisations répondent donc parfaitement aux critères des tâches à effectuer. La visualisation *treemap* est utilisée pour représenter une vue globale et périphérique des informations de conscience de groupe, permettant à l'utilisateur d'être notifié des nouvelles informations et des changements d'informations. La visualisation *hypertree* est utilisée pour représenter une vue de navigation de l'ensemble des informations disponibles, permettant à l'utilisateur de naviguer dans l'ensemble des données et d'obtenir les détails d'une information spécifique.

Deux bibliothèques libres et open-source, en Java, implantant les deux visualisations ont été réalisées et publiées sur Internet dans le cadre de cette thèse. Il s'agit de la *Treemap Java Library* et de l'*Hypertree Java Library*. Ces bibliothèques ont été utilisés dans plusieurs projets, industriels ou de recherche, et ont fait l'objet de différentes améliorations et portages de la part de la communauté open-source : l'*Hypertree Java Library* a été portée en *C++*, et les deux bibliothèques ont été portées pour *PDA* ainsi que sous forme d'applet pour le web. À ce jour, ces deux bibliothèques restent les seules implantations des visualisations *treemap* et *hypertree* qui soient libres, génériques, et qui fournissent autant de fonctionnalités.

Les travaux de cette thèse couvrent l'ensemble du spectre des fonctionnalités d'un outils de conscience de groupe, depuis la collecte des informations jusqu'à leur représentation, en passant par leur diffusion. Chaque étape est réalisée de manière à s'adapter à l'activité de l'utilisateur, afin de distraire le moins possible son attention. De plus, l'outil est générique. Il peut être utilisé dans n'importe quelle situation de coopération, et sur n'importe quel système. Enfin, le modèle cognitif et l'architecture fonctionnelle proposent un cadre théorique réutilisable pour l'étude de la conscience de groupe.

2 Perspectives

L'utilisation du contexte apparaît de plus en plus comme la prochaine étape importante de l'informatique, comme le montrent les derniers paradigmes à la mode : l'informatique située, l'informatique orientée contexte (*context-centered*), les applications adaptées au contexte (*context-aware*), les interfaces perceptives (*perceptual*), les interfaces attentives... L'approche suivie par cette thèse s'inscrit directement dans ce mouvement, en appliquant le concept d'utilisation du contexte au problème de la conscience de groupe.

Le système de conscience de groupe décrit dans cette thèse n'est qu'une première étape dans l'implantation des propositions de cette thèse. Pour être finalisé, le système doit être intégré à une situation de coopération réelle, avec de véritables utilisateurs. Cette intégration est nécessaire pour la spécification des réseaux bayesiens et pour l'apprentissage des réseaux neuronaux de l'outil. De plus, une utilisation du système dans un cadre réel de coopération permettra de valider les différentes propositions de cette thèse. En particulier,

il reste à répondre aux questions suivantes :

- le prototype a-t-il une bonne fiabilité en ce qui concerne l'inférence du contexte ?
- L'apport du contexte de travail améliore-t-il l'efficacité du travail du groupe ou la satisfaction subjective de l'utilisateur ?
- Comment le prototype est-il utilisé ? Notamment, est-il utilisé dans un cadre ou pour un but initialement non prévu ?
- Le prototype modifie-t-il la façon de travailler des utilisateurs ?

En ce qui concerne les perspectives de travaux futurs prolongeant ceux de cette thèse, le point principal concerne la notion d'interruption.

L'outil proposé limite l'interruption de l'utilisateur en adaptant le temps et la manière de la présentation de nouvelles informations. Cependant, cette adaptation doit être apprise au fur et à mesure de l'utilisation de l'outil. Un premier point intéressant serait la modélisation du degré d'« interruptabilité » de l'utilisateur. Ce degré représenterait une mesure du dérangement qu'occasionnerait une interruption de l'utilisateur dans son activité. Une modélisation du degré d'interruptabilité permettrait à l'outil d'être plus efficace dans la gestion des notifications d'informations.

Un deuxième point intéressant concerne l'utilisation de l'outil comme aide à la reprise du travail après l'interruption, que celle-ci soit due à l'outil de conscience de groupe ou pas. En effet, l'outil garde une trace du contexte de travail de l'utilisateur. Cette trace peut être utilisée par l'utilisateur pour reconstruire son contexte de travail et reprendre là où il en était. Enfin, l'outil peut lui-même sentir l'interruption (par l'étude du contexte), et proposer un résumé de l'activité en cours pour aider à la reprise du travail.

Ces perspectives sont possibles notamment grâce à un certain nombres de travaux qui existent déjà sur la modélisation des interruptions et leurs conséquences sur les activités collaboratives [DRW95, Jam96, Lat99, OF95, PQS96, RHRV94, SVV97]. Certaines études permettant aussi de mieux comprendre comment l'utilisateur choisit les activités à effectuer en premier, et comment l'utilisateur gère son travail [Lah00a, Lah00b] peuvent aussi être utilisées pour comprendre comment aider l'utilisateur à reprendre son travail après une interruption. Ces études sont faites dans le cadre du travail sur le *COS*, (*Cognitive Overflow Syndrome*) [Kir00, LLGZ97], dont la surcharge d'informations des outils de conscience de groupe fait partie. L'analyse de la place de l'outil de conscience de groupe dans les théories traitant du *COS*, notamment la théorie des Transactions Intellectuelles [Zac00] et sa suite, la théorie des Transactions Communicationnelles Symboliques [Zac04], est aussi une perspective à plus long terme.

Plusieurs perspectives sont envisageables dans le domaine de la visualisation d'informations de conscience de groupe, ainsi que dans l'évolution des bibliothèques libres de visualisation.

En ce qui concerne l'évolution des bibliothèques de visualisation, la première perspective concerne le passage à la troisième dimension. Par analogie, un modèle de cet espace peut être construit dans la sphère unité de l'espace euclidien à trois dimensions. Certains travaux ont déjà été faits dans cette direction [Mun97, Mun98, MB95, Hyu00]. Cependant, nous pensons que le point de vue pris n'est pas forcément le plus adéquat pour la visualisation d'informations, même s'il est juste d'un point de vue mathématique. Nous proposons notamment de continuer à utiliser le modèle de Poincaré, plus intuitif et plus adéquat pour la visualisation. De plus, nous proposons d'utiliser les algorithmes déve-

loppés dans le cadre de la visualisation *treemap* pour résoudre le problème de placement des fils. Finalement, notre principal apport est de proposer une vue immersive dans les données, en plaçant l'utilisateur au centre de la sphère unité. Ces travaux sont décrits dans un article en préparation [BB04].

Une autre évolution de la bibliothèque est l'utilisation de l'algorithme de placement en secteur pour tracer non pas des arbres mais des graphes, directement dans l'espace hyperbolique. L'idée est d'utiliser les algorithmes de placement de graphes existants, normalement prévu pour l'espace euclidien, et de regarder leur application dans l'espace hyperbolique. Ceci est possible grâce à l'algorithme de placement en secteur car la notion d'angle et notamment de correction de direction disparaît. Un nœud peut ainsi être placé par rapport à ses voisins dès que l'un de ces voisins est au centre du disque.

Concernant la bibliothèque de visualisation *treemap*, plusieurs nouvelles fonctionnalités sont à l'étude, notamment une fonctionnalité de déformation élastique des *treemaps* permettant non seulement de mieux apprécier la hiérarchie d'informations, mais aussi d'améliorer la vue en détail des nœuds de petite taille.

Une perspective en cours d'implantation vise à rassembler les deux bibliothèques en une seule bibliothèque de visualisation de données arborescentes, baptisée *TreeVisu*. Cette bibliothèque offre, à partir d'une même source de données, une visualisation *treemap* et une visualisation *hypertree*. De plus, ces deux visualisations peuvent interagir : la sélection d'un nœud sur l'une des visualisations se répercute sur l'autre. D'autres fonctionnalités sont disponibles, comme une fonctionnalité de loupe, ou une possibilité de lire les données depuis de nombreux formats dont XML. Enfin, la bibliothèque est ouverte et générique pour faciliter l'implantation d'autres visualisations de données arborescentes [HMM00], comme la visualisation SunSpot. Cette bibliothèque est en cours d'implantation.

Concernant la visualisation d'informations de conscience de groupe, plusieurs axes attirent notre attention. En premier, l'utilisation de la psychologie cognitive, et notamment de la psychologie de la perception, est un point particulièrement important si l'on veut créer une visualisation efficace. Un grand nombre de travaux existent sur l'utilisation de la psychologie dans le domaine de l'interaction homme-machine [CMN83, Car87, Car91, Car03], et de nombreux résultats et théories sont disponibles [Bag99, Gib86, Gor97, Jon02]. En particulier, la théorie de la Gestalt nous semble contenir un grand nombre de possibilités sous-utilisée [CdF+99, Gui79, Jim97, Koh47].

Les résultats de Norman [Nor93, Nor98a, Nor98b, Nor04] sont aussi une grande source d'inspiration, particulièrement concernant le design et l'esthétique des interfaces. Concernant le design, d'autres travaux, autant écrits [DK01a, DK01b, Jac00, LZ03, WB01, Wil04, Woo03] que programmés [Fry04, Ste03] montrent la voix de la nouvelle génération d'interfaces. La création de ces interfaces est basée sur l'expérience [Joh00, Lau90, MS95, Ras00, Tog92] ainsi que sur de nouveaux modèles et de nouvelles interactions [BLL00, BLM00]. L'utilisation de la librairie graphique 3D OpenGL pour représenter ces nouvelles interfaces est même apparue dans les logiciels grand public, puisque c'est la base de l'interface du dernier système d'exploitation d'Apple [App04], MacOS X. La prochaine étape de l'interaction homme-machine pourrait bien être la réalisation d'interfaces non seulement statiquement esthétiques [FRC+93, Tog93] mais aussi dynamiquement plaisantes [CU93, Las87]. Cette perspective de la nouvelle génération des interfaces graphiques offre un maximum de possibilités.

Bibliographie

[App04] Apple: *"Think Different"*. `http://www.apple.com`, 2004.

[ATS95] Asahi, Toshiyuki, David Turo et Ben Schneiderman: *Using Treemaps to Visualize the Analytic Hierarchy Process.* Information Systems Research, 6(4) :357–375, 1995.

[BAB+97] Bentley, Richard, Wolfgang Appelt, Uwe Busbach, Elke Hinrichts, Douglas Kerr, Klaas Sikkel, Jonathan Trevor et Gerd Woetzel: *Basic Support for Cooperative Work on the World Wide Web.* International Journal of Human-Computer Studies, 46(6) :827–846, 1997.

[Bag99] Bagot, Jean-Didier: *Information, sensation et perception.* Armand Colin, 1999.

[BB02] Bergé, Benjamin et Christophe Bouthier: *Mathematics and Algorithms for the Hyperbolic Tree Visualization.* En soumission, 2002.

[BB04] Bergé, Benjamin et Christophe Bouthier: *3D-Hyperbolic Tree Visualization : Mathematical Challenges and Algorithmic Solutions.* En préparation, 2004.

[BBF+95] Benford, Steve, John Bowers, Lennart E. Fahlen, Chris Greengalgh et Dave Snowdon: *User Embodiment in Collaborative Virtual Environments.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-95)*, pages 242–249. ACM Press, 1995.

[BC01] Bouthier, Christophe et Gérôme Canals: *Le contexte comme base de la conscience de groupe.* Dans *Coopération, Innovation et Technologie (CITE 2001), Troyes, France*, novembre 2001.

[Bed01] Bederson, Benjamin B.: *PhotoMesa : A Zoomable Image Browser Using Quantum Treemaps and Bubblemaps.* Dans *Proceedings of the ACM Annual Symposium on User Interface Software and Technology (UIST-01)*, pages 71–80, Orlando, 2001. ACM Press.

[Ber98] Bertin, Jacques: *Sémiologie graphique, 3e édition.* Éditions de l'École des Hautes Études en Sciences Sociales, 1998.

[BFH00] Budzik, Jay, Xiaobin Fu et Kristian J. Hammond: *Facilitating Opportunistic Communication by Tracking the Documents People Use.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-00)*, Philadelphia, 2000. ACM Press.

[BHST95] Bentley, Richard, Thilo Horstmann, Klaas Sikkel et Jonathan Trevor: *Supporting Collaborative Information Sharing with the WWW : The BSCW*

Shared Workspace System. Dans *World Wide Web Journal : The Fourth International WWW Conference Proceedings*, pages 63–74, Boston, 1995. O'Reilly & Associates, Inc.

[BHvW00] Bruls, Mark, Kees Huizing et Jarke J. van Wijk: *Squarified Treemaps.* Dans *Proceedings of the joint Eurographics and IEEE TCVG Symposium on Visualization*, pages 33–42, Vienna, 2000. Springer.

[Bit03] Bitcheva, Julia: *PointSynchro : un service pour la coordination de procédés interentreprises coopératifs - Thèse en Informatique - Université Henri Poincaré Nancy 1*, 2003.

[BL99] Beaudouin-Lafon, Michel (rédacteur): *Computer Supported Co-operative Work*, tome 7 de *Trends in Software.* John Wiley & Sons, 1999.

[BLK92] Beaudouin-Lafon, Michel et Alain Karsenty: *Transparency and Awareness in a Real-Time Groupware System.* Dans *Proceedings of the 5th ACM Annual Symposium on User Interface Software and Technology (UIST-92)*, pages 171–180, Monterey, 1992. ACM Press.

[BLL00] Beaudouin-Lafon, Michel et Henry Michael Lassen: *The architecture and implementation of CPN2000, a post-WIMP graphical application.* Dans *Proceedings of the 13th ACM Annual Symposium on User Interface Software and Technology (UIST-00)*, pages 181–190, San Diego, 2000. ACM Press.

[BLM00] Beaudouin-Lafon, Michel et Wendy E. Mackay: *Reification, polymorphism and reuse : Three principles for designing visual interfaces.* Dans *Proceedings of the ACM working conference on Advanced Visual Interfaces (AVI-00)*, pages 102–109, Palermo, Italy, 2000. ACM Press.

[Bød89] Bødker, Susanne: *A Human Activity Approach to User Interfaces.* Human Computer Interaction, 4(3) :151–196, 1989.

[Bol32] Bolyai, Johann: *Appendix, scientiam spatii absolute veram exhibens.* Dans *Tentamen juventutem studiosam in elementa matheseos purae (...) introducenti*, 1832.

[Bou01] Bouthier, Christophe: *A Free Java Library for Treemap Visualization.* Dans *Treemap Implementations and Applications Workshop, HCIL, University of Maryland.* Ben Shneiderman, mai 2001.

[Bou02] Bouthier, Christophe: *Using TreeMaps and Hyperbolic Trees for Statistical Medical Data Visualization.* Dans *4th International Workshop on Enterprise Networking and Computing in Healthcare Industry - HEALTHCOM'2002, Nancy, France.* Jean-Pierre Thomesse, juin 2002.

[BP99] Billsus, Daniel et Michael J. Pazzani: *A personal news agent that talks, learns and explains.* Dans *Proceedings of the 4th ACM International Conference on Autonomous Agents (AGENTS-99)*, pages 268–275, Seattle, 1999. ACM Press.

[BS93] Bellotti, Victoria et Abigail Sellen: *Design for Privacy in Ubiquitous Computing Environments.* Dans *Proceedings of the Third European Conference*

on Computer-Supported Cooperative Work (ECSCW-93). Kluwer Academic Press., 1993.

[BS98] Bogdan, Christian et Olle Sundblad: *A Cue-based, Integrated System for Supporting Social Awareness.* Technical Report TRITA-NA- D9904, CID-45, Centre for User Oriented IT Design, Dept. Computing Science, Royal Institute of Technology, 1998.

[BS03] Bederson, Benjamin B. et Ben Shneiderman: *The Craft of Information Visualization - Readings and Reflections.* Morgan Kaufmann, 2003.

[BSW02] Bederson, Benjamin B., Ben Shneiderman et Martin Wattenberg: *Ordered and Quantum Treemaps : Making Effective Use of 2D Space to Display Hierarchies.* ACM Transactions on Graphics, 21(4) :833–854, 2002.

[BT94] Blanchet, Alain et Alain Trogon: *La Psychologie des Groupes.* Nathan Université, 1994.

[Bul96] Bulatov, Vladimir: *HyperProf applet.* `http://www.physics.orst.edu/~bulatov/HyperProf/`, 1996.

[Car87] Carroll, John M.: *Interfacing Thought : Cognitive Aspects of Human-Computer Interaction.* Morgan Kaufmann, 1987.

[Car91] Carroll, John M.: *Designing Interaction : Psychology at the Human-Computer Interface.* Cambridge University Press, 1991.

[Car03] Carroll, John M.: *HCI Models, Theories, and Frameworks : Toward a Multidisciplinary Science.* Morgan Kaufmann, 2003.

[CB91] Clark, Herbert H. et Susan E. Brennan: *Grounding in Communication.* Dans *Perspectives on Socially Shared Cognition*, Washington, 1991. American Psychological Association.

[CBGM98] Canals, Gérôme, Christophe Bouthier, Claude Godart et Pascal Molli: *Tuamotu : Une infrastructure distribuée pour le support des entreprise-projets.* Dans R. Dssouli, P. Dini, M. Kadoch (rédacteur) : *Actes du Colloque International sur les Nouvelles Technologies de la Répartition (NOTERE-98)*, pages 103–118, Montréal, 1998. CRIM.

[CBL96] Conversy, Stephane et Michel Beaudouin-Lafon: *Le son dans les applications interactives.* Dans *Nouvelles Interfaces Homme-Machine, Rapport du groupe de travail de l'OFTA*, numéro 18 dans *Série ARAGO*, pages 65–81. OFTA, 1996.

[CdF+99] Casati, Roberto, J.-P. d'Alès, J. Froment, J.-M. Morel, M.-A. Peterson, S.-E. Palmer, B. Smith et H.-A. Simon: *Présence de la Gestalt. Intellectica, 1999/1 (28).* ARCo, 1999.

[CG99] Chen, Datong et Hans-Werner Gellersen: *Recognition and Reasoning in an Awareness Support System for Generation of Storyboard-like Views of Recent Activity.* Dans *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work (GROUP-99)*, pages 356–364, Phoenix, 1999. ACM Press.

[CG01] Cheng, Jie et Russell Greiner: *Learning Bayesian Belief Network Classifiers : Algorithms and System.* Dans *Advances in Artificial Intelligence (AI-2001)*, tome 2056 de *Lecture Notes in Computer Science*, pages 141–151. Springer, 2001.

[CH92] Cooper, Gregory F. et Edward Herskovits: *A Bayesian Method for the Induction of Probabilistic Networks from Data.* Machine Learning, 9(4) :309–347, 1992.

[Cha91] Charniak, Eugene: *Bayesian networks without tears.* AI, 12(4) :50–63, 1991.

[CJMS00] Cohen, Doron, Michael Jacovi, Yoelle Maarek et Vladimir Soroka: *Collection Awareness on the Web via Livemaps.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-00)*, Philadelphia, 2000. ACM Press.

[CMG98] Canals, Gérôme, Pascal Molli et Claude Godart: *TuaMotu : Supporting Telecooperative Engineering Applications Using Replicated Versions.* Dans *Internet-based GROupware for User Participation in product development (IGROUP-98)*, Seattle, 1998.

[CMN83] Card, Stuart K., Thomas P. Moran et Allen Newell: *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates Publishers, 1983.

[CMS99] Card, Stuart K., Jock D. Mackinlay et Ben Shneiderman: *Readings in Information Visualization - Using Vision To Think.* Morgan Kaufmann, 1999.

[CSMD00] Cheverst, Keith, Gareth Smith, Keith Mitchell et Nigel Davies: *Exploiting Context to Support Social Awareness and Social Navigation.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-00)*, Philadelphia, 2000. ACM Press.

[CU93] Chang, Bay-Wei et David Ungar: *Animation : From Cartoons to the User Interface.* Dans *Proceedings of the ACM Annual Symposium on User Interface Software and Technology (UIST-93)*, pages 45–55. ACM Press, 1993.

[DA99] Dey, Anind K. et Gregory D. Abowd: *Towards a Better Understanding of context and context-awareness.* Technical Report GIT-GVU-99-22, Graphics, Visualization, and Usability Center, College of Computing, Georgia Institute of Technology, 1999.

[DA01] Dey, Anind K. et Gregory D. Abowd: *A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications.* Human Computer Interaction (HCI) Journal, 16(2-4) :97–166, 2001.

[Dan00] Danis, Catalina M.: *Extending the Concept of Awareness to Include Static and Dynamic Person Information.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-00)*, Philadelphia, 2000. ACM Press.

[DB92a] Dourish, Paul et Victoria Bellotti: *Awareness and Coordination in Shared Workspaces.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-92)*, pages 107–114, Toronto, Ontario, 1992. ACM Press.

[DB92b] Dourish, Paul et Sara Bly: *Portholes : Supporting Awareness in a Distributed Work Group.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-92)*, pages 541–547, Monterey, 1992. ACM Press.

[DFAB98] Dix, Alan, Janet Finlay, Gregory D. Abowd et Russell Beale: *Human-Computer Interaction, second edition.* Prentice Hall, 1998.

[Dio97] Dion, Emmanuel: *Invitation à la théorie de l'information.* Points Sciences, 1997.

[DK01a] Dodge, Martin et Rob Kitchin: *Atlas of Cyberspace.* Addison Wesley, 2001.

[DK01b] Dodge, Martin et Rob Kitchin: *Mapping Cyberspace.* Routledge, 2001.

[Dor03] Dortier, Jean-François: *Le Cerveau et la Pensée, 2e édition.* Sciences Humaines Éditions, 2003.

[DRW95] Dix, Alan, Devina Ramduny et Julie Wilkinson: *Interruptions, Deadlines and Reminders : Investigations into the Flow of Cooperative Work.* Technical Report RR9509, School of Computing and Mathematics, Huddersfield University, 1995.

[Edw97] Edwards, Keith W.: *Representing Activity in Collaborative Systems.* Dans *Proceedings of the Sixth IFIP Conference on Human Computer Interaction (Interact-97)*, Sydney, 1997.

[EMP+97] Edwards, Keith W., Elizabeth D. Mynatt, Karin Petersen, Mike J. Spreitzer, Douglas B. Terry et Marvin M. Theimer: *Designing and Implementing Asynchronous Collaborative Applications with Bayou.* Dans *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST-97)*, pages 119–128, Banff, 1997. ACM Press.

[End95] Endsley, Mica R.: *Toward a Theory of Situation Awareness in Dynamic Systems.* Human Factors, 37(1) :32–64, 1995.

[FF00] Fontana, Francesca Arcelli et Ferrante Formato: *User Adaptive Models Based on Similarity.* Dans *Proceedings of the ACM/SIGAPP Symposium on Applied Computing (SAC-00)*, pages 501–504, Como, 2000. ACM Press.

[FLMM91] Fischer, Gerhard, Andreas C. Lemke, Thomas Mastaglio et Anders I. Morch: *The Role of Critiquing in Cooperative Problem Solving.* ACM Transactions on Information Systems, 9(2) :123–151, 1991.

[FMK96] Fitzpatrick, Geraldine, Tim Mansfield et Simon M. Kaplan: *Locales Framework : Exploring foundations for collaboration support.* Dans *Proceedings of the Sixth Australian Conference on Computer-Human Interaction (OzCHI-96)*, Hamilton, 1996. IEEE Computer Society Press.

[FMK+99] Fitzpatrick, Geraldine, Tim Mansfield, Simon Kaplan, David Arnold, Ted Phelps et Bill Segall: *Augmenting the workaday world with Elvin.* Dans *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work (ECSCW-99)*, pages 431–450, Copenhagen, 1999. Kluwer Academic Press.

[FPSK98] Fitzpatrick, Geraldine, Sara Parsowith, Bill Segall et Simon Kaplan: *Tickertape : Awareness in a Single Line.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (Summary) (CHI-98)*, pages 281–282, Seattle, 1998. ACM Press.

[FR89] Fortin, Claudette et Robert Rousseau: *Psychologie Cognitive, une approche de traitement de l'information.* Télé-université, Université du Québec, 1989.

[FRC+93] Farrand, Brady A., Marc Rochkind, Jean-Marie Chauvet, Bruce "Tog" Tognazzini et David C. Smith: *Common Elements in Today's Graphical User Interfaces : The Good, the Bad, and the Ugly.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (INTERCHI-93)*, pages 470–473. ACM Press, 1993.

[Fry04] Fry, Ben: *Organic Information Design.* `http://acg.media.mit.edu/people/fry/`, 2004.

[FS94] Furuta, Richard et David P. Stotts: *Interpreted Collaboration Protocols and their use in Groupware Prototyping.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-94)*, pages 121–131, Chapel Hill, 1994. ACM Press.

[FS01] Fiore, Andrew et Marc A. Smith: *Treemap Visualizations of Newsgroups.* Technical Report MSR-TR-2001-94, Microsoft Research, 2001.

[GB95] Greengalgh, Chris et Steve Benford: *MASSIVE : A Collaborative Virtual Environment for Teleconferencing.* ACM Transactions on Computer-Human Interactions, 2(3) :239–261, 1995.

[GBC+01] Godart, Claude, Christophe Bouthier, Philippe Canalda, François Charoy, Pascal Molli, Olivier Perrin, Helene Saliou, Jean-Claude Bignon, Gilles Halin et Olivier Malcurat: *Asynchronous Coordination of Virtual Teams in Creative Applications (co-design orco-engineering) : requirements and Design Criteria.* Dans *Information Technologies for Virtual Enterprises*, janvier 2001.

[GCP+03] Godart, Claude, François Charoy, Marc Patten, Nicolas Grégori, Jean-Charles Hautecouverture et Isabelle Faugeras: *Coopéra : apprendre à coopérer et apprendre en coopérant.* Dans *Conférence Internationale sur l'Enseignement Ouvert et en Ligne (ICOOL-03)*, Ile Maurice, 2003.

[GFLM99] Garcia, Octavio, Jesus Favela, Guillermo Licea et Roberto Machorro: *Extending a collaborative architecture to support emotional awareness.* Dans *Proceedings of the Workshop on Emotion-Based Agent Architectures, Autonomous Agents (EBAA-99)*, pages 46–52, 1999.

[GFM99] Garcia, Octavio, Jesus Favela et Roberto Machorro: *Emotional Awareness in Collaborative Systems.* Dans *Proceedings of the String Processing and Information Retrieval Symposium & International Workshop on Groupware (SPIRE-99)*, Cancun, 1999. IEEE Computer Society Press.

[GG96] Gutwin, Carl et Saul Greenberg: *Workspace Awareness for Groupware.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI- 96)*, pages 208–209, Vancouver, 1996. ACM Press.

[GG98] Gutwin, Carl et Saul Greenberg: *Effects of Awareness Support on Groupware Usability.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-98)*, pages 511–518, Los Angeles, 1998. ACM Press.

[GG99] Gutwin, Carl et Saul Greenberg: *The effects of workspace awareness support on the usability of real-time distributed groupware.* ACM Transactions on Computer-Human Interaction, 6(3) :243–281, 1999.

[GG01] Gutwin, Carl et Saul Greenberg: *A Descriptive Framework of Workspace Awareness for Real-Time Groupware.* Dans *Computer Supported Cooperative Work.* Kluwer Academic Press., 2001.

[GGC96] Greenberg, Saul, Carl Gutwin et Andy Cockburn: *Awareness Through Fisheye Views in Relaxed-WYSIWIS Groupware.* Dans *Graphics Interface (GI-96)*, pages 28–38, Toronto, 1996. Morgan Kaufmann.

[GGR96] Greenberg, Saul, Carl Gutwin et Mark Roseman: *Semantic Telepointers for Groupware.* Dans *The Sixth Australian Conference on Computer-Human Interaction (OzCHI-96)*, pages 54–61, Hamilton, 1996. IEEE Computer Society Press.

[GHB+01] Godart, Claude, Gilles Halin, Jean-Claude Bignon, Christophe Bouthier, Pascal Malcurat et Pascal Molli: *Implicit or Explicit Coordination of Virtual Teams in Building Design.* Dans *CAADRIA (Computer-Aided Architectural Design Research In Asia), Sydney*, avril 2001.

[Gib86] Gibson, James J.: *The Ecological Approach to Visual Perception.* LEA Publisher, 1986.

[GMM+92] Gaver, William, Thomas Moran, Allan MacLean, Lennart Lovstrand, Paul Dourish, Kathleen Carter et Bill Buxton: *Realizing a Video Environment : EuroPARC's RAVE System.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-92)*, pages 27–35, Monterey, 1992. ACM Press.

[GMO+04] Godart, Claude, Pascal Molli, Gérald Oster, Olivier Perrin, Hala Skaf-Molli, Pradeep Ray et Fethi Rabhi: *The ToxicFarm integrated cooperation framework for virtual teams.* Distributed and Parallel Databases, 15(1) :67–88, 2004.

[Gof73a] Goffman, Erving: *La mise en scène de la vie quotidienne : 1. La présentation de soi.* Les éditions de minuit, 1973.

[Gof73b] Goffman, Erving: *La mise en scène de la vie quotidienne : 2. Les relations en public.* Les éditions de minuit, 1973.

[Gof74] Goffman, Erving: *Les rites d'interaction.* Les éditions de minuit, 1974.

[Gor97] Gordon, Ian E.: *Theories of Visual Perception, second edition.* John Wiley and Sons, 1997.

[GP00] Gross, Tom et Wolfgang Prinz: *Web-Browsing on Stage : Using the Theatre of Work for Awareness on the WWW.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-00)*, Philadelphia, 2000. ACM Press.

[GP03] Gross, Tom et Wolfgang Prinz: *Awareness in Context : A Light-Weight Approach.* Dans *Proceedings of the Eighth European Conference on Computer-Supported Cooperative Work (ECSCW-03)*, pages 295–314, Helsinki, 2003. Kluwer Academic Press.

[GR98] Greenberg, Saul et Mark Roseman: *Using a Room Metaphor to Ease Transitions in Groupware.* Research report 98/611/02, Department of Computer Science, University of Calgary, 1998.

[GRG96] Gutwin, Carl, Mark Roseman et Saul Greenberg: *A Usability Study of Awareness Widgets in a Shared Workspace Groupware System.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-96)*, pages 258–267, Cambridge, 1996. ACM Press.

[Gru88] Grudin, Jonathan: *Why CSCW Applications Fail : Problems in the Design and Evaluation of Organizational Interfaces.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-88)*, pages 85–93. ACM Press, 1988.

[GS00] Gellersen, Hans-W. et Albrecht Schmidt: *Look, Who's Visiting ? Supporting Visitor Awareness in the Web.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-00)*, Philadelphia, 2000. ACM Press.

[Gui79] Guillaume, Paul: *La Psychologie de la forme.* Flammarion, 1979.

[Gut97] Gutwin, Carl: *Workspace Awareness in Real-Time Distributed Groupware (PhD Thesis) - Departement of Computer Science, University of Calgary*, 1997.

[Hal71] Hall, Edward T.: *La dimension cachée.* Édition du Seuil, 1971.

[Hal79] Hall, Edward T.: *Au delà de la culture.* Édition du Seuil, 1979.

[Hal84a] Hall, Edward T.: *La danse de la vie.* Édition du Seuil, 1984.

[Hal84b] Hall, Edward T.: *Le language silencieux.* Édition du Seuil, 1984.

[HBH+98] Horvitz, Eric, Jack Breese, David Heckerman, David Hovel et Koos Rommelse: *The Lumière Project : Bayesian User Modeling for Inferring the Goals and Needs of Software Users.* Dans *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 256–265. Morgan Kaufmann, 1998.

[Hen01] Henle, Michael: *Modern Geometries.* Prentice Hall, 2001.

[HHK00] Hollan, James, Edwin Hutchins et David Kirsh: *Distributed Cognition : Toward a New Foundation for Human-Computer Interaction Research.* ACM Transactions on Computer-Human Interaction, 7(2) :174–196, 2000.

[HMM00] Herman, Ivan, Guy Melançon et Scott M. Marshall: *Graph Visualization and Navigation in Information Visualization : a Survey.* IEEE Transactions on Visualization and Computer Graphics, 6(1) :24–43, 2000.

[Hor99] Horvitz, Eric: *Principles of Mixed-Initiative User Interfaces.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-99)*, pages 159–166, Pittsburgh, 1999. ACM Press.

[Hor03] Horvitz, Eric: *Models of ATTENTION in Computing and Communication : From Principles to Applications.* Communications of the ACM, 46(3) :52–59, 2003.

[HS92] Hollan, Jim et Scott Stornetta: *Beyond Being There.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-92)*, pages 119–125. ACM Press, 1992.

[Hut95] Hutchins, Edwin: *Cognition in the Wild.* MIT Press, 1995.

[HW00] Handel, Mark et Graham Wills: *TeamPortal : Providing Team Awareness On the Web.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-00)*, Philadelphia, 2000. ACM Press.

[Hyu00] Hyun, Young: *Walrus.* `http://www.caida.org/~youngh/walrus/walrus.html`, 2000.

[Jac00] Jacobson, Robert: *Information Design.* The MIT Press, 2000.

[Jam96] Jambon, Francis: *Formal Modelling of Task Interruptions.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-96)*, pages 45–46, Vancouver, 1996. ACM Press.

[JATP95] Jager Adams, Marilyn, Yvette J. Tenney et Richard W. Pew: *Situation Awareness and the Cognitive Management of Complex Systems.* Human Factors, 37(1) :85–104, 1995.

[Jim97] Jimenez, Manuel: *La Psychologie de la Perception.* Flammarion, 1997.

[Joh88] Johansen, Robert: *Current User Approaches to Groupware.* Dans *Groupware : Computer Support for Business Teams*, pages 12–44. The Free Press., 1988.

[Joh00] Johnson, Jeff: *GUI Bloopers.* Morgan Kaufmann, 2000.

[Jon02] Jonsson, Erik: *Inner Navigation.* Scribner, 2002.

[JS91] Johnson, Brian et Ben Shneiderman: *Tree-Maps : A Space-Filling Approach to the Visualization of Hierarchical Information Structures.* Dans *Proceedings of the International IEEE Visualization Conference*, pages 284–291, San Diego, 1991.

[JT92] Jungmeister, Walter-Alexander et David Turo: *Adapting Treemaps to Stock Portfolio Visualization.* Technical Report UMCP-CSD CS-TR-2996, University of Maryland, 1992.

[Kir00] Kirsh, David: *A Few Thoughts on Cognitive Overload.* Intellectica, 30(1) :19–51, 2000.

[Koh47] Kohler, Wolfgang: *Gestalt Psychology.* Liveright, 1947.

[KPTS94] Kumar, Marsha, Catherine Plaisant, Marko Teittinen et Ben Shneiderman: *Visual Information Management for Network Configuration.* Technical Report CAR-TR-716 CS-TR-3288 ISR-TR-94-45, Dept. of Comp. Sci., U. of Maryland, 1994.

[KTBL93] Karsenty, Alain, Christophe Tronche et Michel Beaudouin-Lafon: *GroupDesign : Shared editing in a heterogeneous environment.* Usenix Journal of Computing Systems, 6(2) :167–195, 1993.

[Kuu92] Kuutti, Kari: *Identifying Potential CSCW Applications by Means of Activity Theory Concepts : A Case Example.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-92)*, pages 233–240. ACM Press, 1992.

[KW99] Kramer, Bernd J. et Lutz M. Wegner: *Beyond the Whiteboard : Synchronous Collaboration in Shared Object Spaces.* Dans *Proceedings of the Seventh IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS-99)*, Tunisia, 1999. IEEE Computer Society Press.

[Lah00a] Lahlou, Saadi: *Le Design Cognitif : Construire des Outils au Service des Travailleurs Intellectuels.* Septième École d'été, Association pour la Recherche Cognitive, Bonas, France, Juillet 2000, 10-21 Juillet 2000.

[Lah00b] Lahlou, Saadi: *Attracteurs cognitifs et travail de bureau.* Intellectica, 30(1) :75–113, 2000.

[Lal87] Lalesco, Trajan: *La géométrie du triangle, 2^e édition.* Jacques Gabay Éd., 1987.

[Las87] Lasseter, John: *Principles of traditional animation applied to 3D computer animation.* Computer Graphics, 21(4) :35–44, 1987.

[Lat99] Latorella, Kara A.: *Investigating Interruptions : Implications for Flightdeck Performance, Tech Report TM-1999-209707, NASA, Langley Research Center*, 1999.

[Lau90] Laurel, Brenda: *The Art of Human-Computer Interface Design.* Addison-Wesley, 1990.

[LED+99] LaMarca, Anthony, Keith W. Edwards, Paul Dourish, John Lamping, Ian Smith et Jim Thornton: *Taking the work out of workflow : Mechanisms for document-centered collaboration.* Dans *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work (ECSCW-99)*, pages 1–20, Copenhagen, 1999. Kluwer Academic Press.

[Leo78] Leontiev, Aleksie Nikolaevich: *Activity, Consciousness, and Personality.* Prentice Hall, 1978.

[LGS97] Lee, Alison, Andreas Girgensohn et Kevin Schlueter: *NYNEX Portholes : Initial User Reactions and Redesign Implications.* Dans *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work : The Integration Challenge (GROUP-97)*, pages 385–394, Phoenix, 1997. ACM Press.

[LLGZ97] Lahlou, Saadi, Charles Lenay, Yves Gueniffey et Manuel Zacklad: *Le COS (Cognitive Overflow Syndrome).* Bulletin de l'Association pour la Recherche Cognitive, 42(1) :39, 1997.

[Lob37] Lobatchewsky, Nikolaï: *Géométrie imaginaire.* J. Reine Angew. Math., 17 :295–320, 1837. originally published at Kazan in 1836.

[Lon03] Lonchamp, Jacques: *Le Travail Coopératif et ses Technologies.* Hermes Science Publications, 2003.

[LR94] Lamping, John et Ramana Rao: *Laying out and Visualizing Large Using a Hyperbolic Space.* Dans *Proceedings of the ACM symposium on User interface software and technology (UIST-94)*, pages 13–14, California, 1994. ACM Press.

[LR96a] Lamping, John et Ramana Rao: *The Hyperbolic Browser : A Focus+Context Technique for Visualizing Large Hierarchies.* Journal of Visual Languages and Computing, 7(1) :33–55, 1996.

[LR96b] Lamping, John et Ramana Rao: *Visualizing Large Trees Using the Hyperbolic Browser.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-96)*, pages 388–389, Vancouver, 1996. ACM Press.

[LRP95] Lamping, John, Ramana Rao et Peter Pirolli: *A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-95)*, pages 401–408, Denver, 1995. ACM Press.

[LZ03] Lewandowsky, Pina et Francis Zeischegg: *Les Sens du Visuel > graphisme : l'apprentissage du regard.* bloc notes publishing, 2003.

[MB95] Munzner, Tamara et Paul Burchard: *Visualizing the Structure of the World Wide Web in 3D Hyperbolic Space.* Dans *Proceeding of the ACM VRML Symposium (VRML 95) - ACM SIGGRAPH*, pages 33–38, San Diego, 1995. ACM Press.

[MB97] McDaniel, Susan E. et Tom Brinck: *Awareness in Collaborative Systems : A CHI 97 Workshop.* ACM SIGCHI Bulletin, 29(4) :68–71, 1997.

[MBW+98] Mynatt, Elizabeth D., Maribeth Back, Roy Want, Michael Baer et Jason B. Ellis: *Designing Audio Aura.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-98)*, pages 566–573, Los Angeles, 1998. ACM Press.

[Miq00] Miquel, Alexandre: *Affichage Générique d'Arbres à l'aide de la Géométrie Hyperbolique.* Dans *Journées Francophones des Langages Applicatifs (JFLA-00)*, pages 1–14, 2000.

[MM93] Minor, Sten et Boris Magnusson: *A Model for Semi-(a)Synchronous Collaborative Editing.* Dans *Proceedings of the Third European Conference on Computer-Supported Cooperative Work (ECSCW-93)*, pages 219–231, Milano, 1993. Kluwer Academic Publishers.

[MS95] Mullet, Kevin et Darrell Sano: *Designing Visual Interfaces : communication oriented techniques.* Prentice Hall, 1995.

[MSMB01] Molli, Pascal, Hala Skaf-Molli et Christophe Bouthier: *State Treemap : an Awareness Widget for Multi-Synchronous Groupware.* Dans *7th International Workshop on Groupware (CRIWG-01)*, Darmstadt, Germany, septembre 2001.

[Mun97] Munzner, Tamara: *H3 : Laying Out Large Directed Graphs in 3D Hyperbolic Space.* Dans *Proceeding of the IEEE Symposium on Information Visualization (InfoViz 97)*, pages 2–10, 1997.

[Mun98] Munzner, Tamara: *Exploring Large Graphs in 3D Hyperbolic Space.* IEEE Computer Graphics and Applications, 18(4) :18–23, 1998.

[Nar96] Nardi, Bonnie A.: *Context and Consciousness : Activity Theory and Human-Computer Interaction.* MIT Press, 1996.

[Nar98] Nardi, Bonnie A.: *Concepts of Cognition and Consciousness : Four Voices.* Journal of Computer Documentation, 22(1) :31–48, 1998.

[Nei76] Neisser, Ulric: *Cognition and Reality : Principles and Implications of Cognitive Psychology.* Freeman, 1976.

[NHHG98] Nomura, Takahiko, Koichi Hayashi, Tan Hazama et Stephan Gudmundson: *Interlocus : Workspace Configuration Mechanisms for Activity Awareness.* Dans *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW-98)*, pages 19–28, Seattle, 1998. ACM Press.

[Nil98] Nilsson, Nils J.: *Artificial Intelligence : A New Synthesis.* Morgan Kaufmann, 1998.

[NMHR+98] Neuwirth, Christine M., James H. Morris, Susan Harkness Regli, Raviner Chandhok et Geoffrey C. Wenger: *Envisioning Communication : Task-Tailorable Representations of Communication in Asynchronous Work.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-98)*, pages 265–274, Seattle, 1998. ACM Press.

[Nor93] Norman, Donald: *Things That Make Us Smart.* Perseus Books, 1993.

[Nor98a] Norman, Donald: *The Design Of Everyday Things.* The MIT Press, 1998.

[Nor98b] Norman, Donald: *The Invisible Computer.* The MIT Press, 1998.

[Nor04] Norman, Donald: *Emotional Design.* Basic Books, 2004.

[OF95] O'Conaill, Brid et David Frohlich: *Timespace in the Workplace : Dealing with Interruptions.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-95)*, pages 262–263, Denver, 1995. ACM Press.

[OJH03] Orso, Alessandro, James Jones et Mary Jean Harrold: *Visualization of Program-Execution Data for Deployed Software.* Dans *Proceedings of the ACM Symposium on Software Visualization*, pages 67–76, San Diego, 2003. ACM Press.

[Pal99] Palfreyman, Kevin: *Supporting Infrastructure for Presence Awareness and Shared Interaction (PhD thesis), Departement of Computing, Lancaster University*, 1999.

[Pas98] Pascoe, Jason: *Adding Generic Contextual Capabilities to Wearable Computers.* Dans *Proceedings of the Second International Symposium on Wearable Computers (ISWC-98)*, Pittsburgh, 1998. IEEE Computer Society Press.

[Pat99] Paton, Norman W.: *Active Rules in Database Systems.* Springer, 1999.

[PCVDW00] Pirolli, Peter, Stuart K. Card et Mija M. Van Der Wege: *The Effect of Information Scent on Searching Information Visualizations of Large Tree Structures.* Dans *Proceeding of the ACM working conference on Advanced Visual Interfaces (AVI-00)*, pages 161–172, Palermo, 2000. ACM Press.

[PCVDW01] Pirolli, Peter, Stuart K. Card et Mija M. Van Der Wege: *Visual Information Foraging in a Focus+Context Visualization.* Dans *Proceeding of the ACM Conference on Human Factor in Computing Systems (CHI-01)*, pages 506–513, Seattle, 2001. ACM Press.

[Pea95] Pearl, Judea: *Causal Diagrams for Empirical Research.* Biometrika, 82(4) :669–709, 1995.

[Pin97] Pinker, Steven: *How The Mind Works.* Penguin Books, 1997.

[PQS96] Perez-Quinones, Manuel A. et John L. Sibert: *Negotiating User-Initiated Cancellation and Interruption Requests.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-96)*, pages 267–268, Vancouver, 1996. ACM Press.

[Pri99] Prinz, Wolfgang: *NESSIE : An awareness environment for cooperative settings.* Dans *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work (ECSCW-99)*, pages 391–410, Copenhagen, 1999. Kluwer Academic Publishers.

[PRS+94] Preece, Jenny, Yvonne Rodgers, Helen Sharp, David Benyon, Simon Holland et Tom Carey: *Human-Computer Interaction.* Addison-Wesley, 1994.

[Ras00] Raskin, Jef: *The Humane Interface.* Addison Wesley, 2000.

[Rat94] Ratcliffe, John G.: *Foundations of Hyperbolic Manifolds*, tome 149 de *Graduate Texts in Mathematics.* Springer, 1994.

[Rau96] Rauschenbach, Uwe: *Supporting Awareness in Shared Workspaces Using Relevance-dependent Event Notifications.* Dans *Workshop on Collaborative Virtual Environments (CVE-96)*, Nottingham, 1996.

[RHRV94] Rouncefiels, Mark, John A. Hughes, Tom Rodden et Stephen Viller: *Working with "Constant Interruption" : CSCW and the Small Office.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-94)*, pages 275–286, Chapel Hill, 1994. ACM Press.

[Rie67] Riemann, Bernhard: *Über die Hypothesen, welche der Geometrie zu Grunde liegen.* Abh. Ges. Wiss. Göttingen, 13 :133–152, 1867.

[Rod96] Rodden, Tom: *Populating the Application : A Model of Awareness for Cooperative Applications.* Dans *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-96)*, pages 87–96, Cambridge, 1996. ACM Press.

[Rou98] Rouquette, Michel-Louis: *La Communication Sociale.* Dunod, 1998.

[RSB02] Rilling, Juergen, Ahmed Seffah et Christophe Bouthier: *The CONCEPT Project : Applying Source Code Analysis to Reduce Information Complexity*

of Static and Dynamic Visualization Techniques. Dans *Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2002), Paris, France.* IEEE, juin 2002.

[SB97] Simone, Carla et Stefania Bandini: *Compositional Features For Promoting Awareness Within And Across Cooperative Applications.* Dans *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work : The Integration Challenge (GROUP-97)*, pages 358–367, Phoenix, 1997. ACM Press.

[SCGM00] Stasko, John, Richard Catrambone, Mark Guzdial et Kevin McDonald: *An evaluation of space-filling information visualizations for depicting hierarchical structures.* Technical Report GIT-GVU00-03, Graphics, Visualization, and Usability Center, Georgia Institute of Technology, 2000.

[Sch96] Schwabe, Gerhard: *Supporting large meetings with query awareness.* Dans *Proceedings of the Seventh International Workshop on Database and Expert Systems Applications (DEXA-96)*, Zurich, 1996. IEEE Computer Society Press.

[Sch02] Schmidt, Kjeld: *The Problem with 'Awareness'.* Computer Supported Cooperative Work, 11(1) :285–298, 2002.

[SDA99] Salber, Daniel, Anind K. Dey et Gregory D. Abowd: *The Context Toolkit : Aiding the Development of Context-Enabled Applications.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-99)*, pages 434–441, Pittsburgh, 1999. ACM Press.

[SG81] Sanford, Anthony J. et Simon C. Garrod: *Understanding Written Language.* Wiley, 1981.

[Sha48] Shannon, Claude E.: *A Mathematical Theory of Communication.* Bell System Technical Journal, 27 :379–423, 623–656, 1948.

[Shn92] Shneiderman, Ben: *Tree Visualization with Tree-Maps : A 2-D Space-Filling Approach.* ACM Transactions on Computer-Human Interaction, 11(1) :92–99, 1992.

[Shn97] Shneiderman, Ben: *Designing the User Interface, 3^{rd} edition.* Addison-Wesley, 1997.

[Shn00] Shneiderman, Ben: *Workshop on Treemap Implementations and Applications.* `http://www.cs.umd.edu/hcil/soh/2001/abstracts.html`. 18th HCIL Symposium and Open House, University of Maryland, College Park, Maryland, USA, May 31, 2000.

[Sim72] Simon, Herbert A.: *Theories of Bounded Rationality.* In Decision and Organization – McGuire and Radner, Eds., pages 161–176, 1972.

[Spe01] Spence, Robert: *Information Visualization.* ACM Press - Addison-Wesley, 2001.

[SS99] Sawhney, Nitin et Chris Schmandt: *Nomadic Radio : Scaleable and Contextual Notification for Wearable Audio Messaging.* Dans *Proceedings of the*

ACM Conference on Human Factors in Computing Systems (CHI-99), pages 96–103, Pittsburgh, 1999. ACM Press.

[SS00] Sawhney, Nitin et Chris Schmandt: *Nomadic Radio : Speech and Audio Interaction for Contextual Messaging in Nomadic Environments.* ACM Transactions on Computer-Human Interaction, 7(3) :353–383, 2000.

[SSV03] Shell, Jeffrey S., Ted Selker et Roel Vertegaal: *Interacting with Groups of Computers.* Communications of the ACM, 46(3) :40–46, 2003.

[Ste03] Steenberg, Eskil: *Loq Airou.* `http://www.quelsolaar.com`, 2003.

[Suc87] Suchman, Lucy A.: *Plans and Situated Actions : The Problem of Human-Machine Communication.* Cambridge University Press, 1987.

[SVV97] Speier, Cheri, Joseph S. Valacich et Iris Vessey: *The Effects of Task Interruption and Information Presentation on Individual Decision Making.* Dans *Proceedings of the ACM International Conference on Information Systems (CIS-01)*, pages 21–36, Atlanta, 1997. ACM Press.

[SZ00] Stasko, John et Eugene Zhang: *Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations.* Dans *Proceedings of the IEEE Symposium on Information Vizualization 2000*, 2000.

[Tog92] Tognazzini, Bruce: *TOG on Interface.* Addison Wesley, 1992.

[Tog93] Tognazzini, Bruce: *Principles, Techniques, and Ethics of Stage Magic and Their Potential Application to Human Interface Design.* Dans *Proceedings of the ACM Conference on Human Factors in Computing Systems (INTERCHI-93)*, pages 355–362. ACM Press, 1993.

[Tuf90] Tufte, Edward R.: *Envisioning Information.* Graphic Press, 1990.

[Tuf97] Tufte, Edward R.: *Visual Explanations : Images and Quantities, Evidence and Narrative.* Graphic Press, 1997.

[Tuf01] Tufte, Edward R.: *The Visual Display of Quantitative Information, 2^{nd} edition.* Graphic Press, 2001.

[Ver03] Vertegaal, Roel: *Attentive User Interfaces.* Communications of the ACM, 46(3) :30–33, 2003.

[VN00] Vernier, Frederic et Laurence Nigay: *Modifiable Treemaps Containing Variable-Shaped Units.* Dans *Proceedings of the Visualization 2000 Conference*, Salt Lake City, 2000.

[vWvdW99] Wijk, Jarke J. van et Huub van de Wetering: *Cushion treemaps : visualization of hierarchical information.* Dans *Proceedings of the IEEE Symposium on Information Visualization (InfoVis-99)*, pages 73–78, 1999.

[Vyg78] Vygotsky, Lev S.: *Mind and Society.* Harvard University Press, 1978.

[Wal98] Walker, William F.: *Rapid Prototyping of Awareness Services Using a Shared Information Server.* ACM SIGCHI Bulletin, 30(2) :95–101, 1998.

[War00] Ware, Colin: *Information Visualization.* Morgan Kaufmann, 2000.

[WB01] Wildbur, Peter et Michael Burke: *Le Graphisme d'Information.* Thames and Hudson, 2001.

[WC95] Widom, Jennifer et Stephano Ceri: *Active Database Systems.* Morgan Kaufmann, 1995.

[Wil04] Williams, Robin: *The Non-Designer's Design Book, 2nd edition.* Peachpit Press, 2004.

[Win00] Winkin, Yves: *La Nouvelle Communication.* Édition du Seuil, 2000.

[Woo03] Woolman, Matt: *Données à voir : le graphisme d'information sur support numérique.* Thames and Hudson, 2003.

[Zac00] Zacklad, Manuel: *La théorie des transactions intellectuelles : une approche gestionnaire et cognitive pour le traitement du COS.* Intellectica, 30(1) :195–222, 2000.

[Zac04] Zacklad, Manuel: *Transactions communicationnelles symboliques et communauté d'action : une approche de la création de valeur dans les processus coopératifs.* Dans *Les actes du colloque de Cerisy : Connaissance, Organisation, Activité.* Édition la Découverte, 2004.

INSTITUT NATIONAL
POLYTECHNIQUE
DE LORRAINE

AUTORISATION DE SOUTENANCE DE THESE DU DOCTORAT DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE

o0o

VU LES RAPPORTS ETABLIS PAR :
Monsieur Michel BEAUDOUIN-LAFON, Professeur, Université de Paris Sud, Orsay
Monsieur Manuel ZACKLAD, Professeur, Université de Technologie de Troyes

Le Président de l'Institut National Polytechnique de Lorraine, autorise :

Monsieur BOUTHIER Christophe

à soutenir devant un jury de l'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE, une thèse intitulée :

"Mise en contexte de la conscience de groupe : adaptation et visualisation"

en vue de l'obtention du titre de :

DOCTEUR DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE

Spécialité : « **Informatique** »

Fait à Vandoeuvre, le 25 août 2004
Le Président de l'I.N.P.L.
L. SCHUFFENECKER

NANCY BRABOIS
2, AVENUE DE LA
FORET-DE-HAYE
BOITE POSTALE 3
F - 54501
VANDŒUVRE CEDEX

TEL. 33/03.83.59.59.59
FAX. 33/03.83.59.59.55

Résumé:

Cette thèse propose deux contributions à la problématique de la conscience de groupe. La première contribution porte sur la mise en contexte de la conscience de groupe pour sa construction et son adaptation. Le contexte de travail de l'utilisateur est utilisé comme source des informations constituant la conscience de groupe, ainsi que pour adapter la propagation et la représentation de ces informations. L'utilisation du contexte se fait suivant un modèle cognitif proposé après analyse des mécanismes cognitifs de la conscience de groupe. Il en résulte une conscience de groupe plus pertinente et moins intrusive. La deuxième contribution porte sur la représentation des informations de conscience de groupe, et plus particulièrement sur l'utilisation combinée de deux techniques de visualisations, la visualisation *treemap* et la visualisation *hypertree*. L'emploi combiné de ces deux techniques de visualisation permet d'utiliser la représentation la plus adéquate suivant la tâche à accomplir.

Mots-clés: Conscience de groupe, modèle cognitif, contexte, visualisation, *treemap*, *hypertree*.

Group Awareness in Context : Adaptation and Visualization

Abstract:

This thesis deals with two contributions for the group awareness problematic. The first contribution is to put group awareness in context in order to build and adapt it. User's work context is used not only as the source for group awareness information, but also to adapt the propagation and representation of this information. Context use is based on a cognitive model derived from the analysis of group awareness cognitive mechanisms. The result is a more pertinent and less intrusive group awareness. The second contribution deals with the representation of group awareness, and more especially with the combinated uses of two visualization techniques, the *treemap* visualization and the *hypertree* visualization. The combination of those two visualization techniques allows to use the more adapted representation for the task at hand.

Keywords: Group awareness, cognitive model, context, visualization, *treemap*, *hypertree*.

Discipline: informatique

Laboratoire:

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503
Campus Scientifique - B.P. 239 - 54506 Vandœuvre-lès-Nancy CEDEX