Le contexte comme base de la conscience de groupe

Christophe Bouthier, Gérome Canals
LORIA, INRIA Lorraine
{bouthier,canals}@loria.fr

9 novembre 2001

Résumé

Dans cet article, nous proposons une nouvelle façon d'aborder le problème de la conscience de groupe, par l'utilisation du contexte de travail d'un utilisateur. Nous détaillons la modélisation d'un utilisateur par son contexte, ainsi que les méthodes permettant une telle modélisation. Nous montrons alors que l'utilisation du contexte permet d'avoir une conscience de groupe plus complète, plus adéquate et plus adaptable.

Mot-clefs : conscience de groupe, contexte, CSCW

1 Introduction

La conscience de groupe — aussi appelée "group awareness" — est un des problèmes toujours d'actualité dans le domaine de recherche du travail coopératif assisté par ordinateur (Computer Supported Cooperative Work, ou CSCW). Cette notion est définie ainsi par Dourish et Belloti [6] :

> «la compréhension des activités des autres, qui permet de donner un contexte à sa propre activité.»

Cette conscience de groupe est créée par l'ensemble des informations qu'un utilisateur peut recevoir ou déduire à propos de l'activité des autres membres du groupe. Cela passe par l'établissement de communications formelles et informelles entre les participants, ainsi que par la déduction d'informations à partir d'indices de perception. Habituellement, la plupart de ces informations peuvent être recueillies parce que les membres du groupe sont en même temps dans un même lieu, et donc peuvent se «percevoir mutuellement» [25] : voir la lumière sous la porte d'un bureau, ou entendre des bruit de pas derrière.

Dans ce travail, nous abordons le problème de la conscience de groupe dans les groupes virtuels. Un groupe virtuel est constitué de personnes réelles réparties dans l'espace, dans le temps, et éventuellement dans des organisations différentes. Elles doivent cependant coopérer pour atteindre un objectif commun. Elles sont pour cela reliées par un réseau informatique.

Lorsque les membres d'une équipe devant travailler ensemble sont répartis dans le temps ou dans l'espace, une grande partie des informations disparaît, appauvrissant, voire détruisant, cette conscience de groupe. Notre problématique est alors de proposer des outils informatiques qui peuvent aider l'utilisateur à recréer cette conscience de groupe, en envoyant à chaque participant des informations pertinentes sur ce que font les autres membres du groupe.

Pour la conscience de groupe, la notion de pertinence s'exprime par deux conditions :

- les informations doivent être utiles. Elles doivent être compréhensibles par le receveur, mais ne doivent être ni redondantes ni superflues.
- Les informations doivent être assez riches pour permettre la «compréhension des activités des autres», donc la construction de la conscience de groupe.

De plus, lorsque tout le groupe est présent en même temps dans le même lieu, un membre construit sa

conscience de groupe de deux façons :

- de manière périphérique, lorsqu'il est absorbé par une autre tâche ;
- de manière active, lorsqu'il souhaite avoir plus de détails sur ce que font les autres.

Pour être vraiment bénéfique, un système de conscience de groupe doit permettre à l'utilisateur de construire sa conscience de groupe des deux façons, suivant son activité. Le système doit présenter des informations pertinentes, de manière plus périphérique lorsque l'utilisateur est occupé, et de manière plus détaillée lorsque l'utilisateur a besoin de plus d'informations.

Un risque majeur est de submerger les utilisateurs d'informations inutiles. En effet, dans certaines situations — lorsque le groupe est très grand, lorsque l'utilisateur fait partie de plusieurs groupes, ou bien lorsque les objets manipulés sont nombreux — l'utilisateur d'un tel système se retrouve vite noyé sous un flot d'informations, pas toujours intéressantes, ce qui réduit l'utilisabilité du système. Pour palier à ce problème, le système doit pouvoir filtrer les informations, de façon à n'envoyer à une personne que les informations qui l'intéressent.

L'utilisateur doit aussi être tenu au courant des changements importants dans les activités des autres membres du groupe. Un autre risque est alors que l'utilisateur rate une de ces notifications, et ait une fausse conscience de ce qui se passe dans le groupe. Les notifications doivent donc être clairement vues par l'utilisateur, quel que soit sa tâche. Cependant, l'utilisateur ne doit pas non plus être continuellement interrompu par les notifications excessives du système de conscience de groupe à chaque fois qu'une information change. Le système doit pouvoir adapter ses notifications à l'activité de l'utilisateur.

Le système de conscience de groupe doit donc :

- fournir à chaque membre du groupe les informations nécessaires pour comprendre les activités des autres utilisateurs ;
- adapter la diffusion et la représentation de ces informations suivant le contexte du travail de chacun.

Or l'activité d'un utilisateur fait partie de son contexte de travail. Notre proposition est donc d'utiliser le contexte de travail de chacun comme base de la conscience de groupe. De cette manière, le contexte sert de source des informations formant la conscience de groupe, et permet l'adaptation de sa diffusion et de sa représentation.

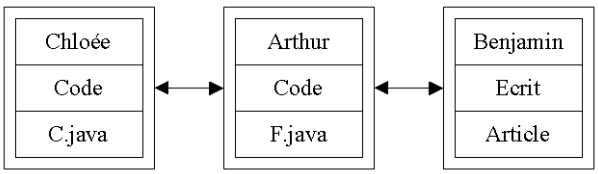

FIG. 1 – *Les contextes d'Arthur, Benjamin et Chloée*

Dans la prochaine section, nous décrivons un exemple servant de base à l'illustration de nos propos. La 3 section est consacrée à la modélisation du contexte. Nous abordons dans la section d'après son utilisation, d'abord dans la diffusion, puis dans la représentation des informations. Ensuite, nous nous intéressons aux méthodes permettant la construction du contexte, puis nous parlons de la mise en œuvre. Finalement nous situons notre travail par rapport à l'existant du domaine.

2 Exemple

L'exemple détaillé dans cette section est un exemple fictif qui a pour but d'illustrer les fonctionnalités attendues d'un mécanisme de conscience de groupe.

Prenons Arthur, chercheur. Il a de nombreux projets en cours, dont deux particulièrement en ce moment : un article, qu'il écrit avec Benjamin, et un logiciel, qu'il développe avec Chloée. Pour le moment, il est en train de coder sur le logiciel, et il attend la partie du code que doit faire Chloée pour pouvoir commencer à faire les tests.

Sur son écran, en bas à droite, se trouve une petite fenêtre, contenant différents rectangles colorés. Chaque rectangle représente une personne avec qui Arthur travaille. En ce moment, les rectangles de Benjamin et de Chloée sont rouges, ce qui signifie qu'ils sont en plein travail (figure 1). En passant la souris sur un rectangle, une aide indique à Arthur

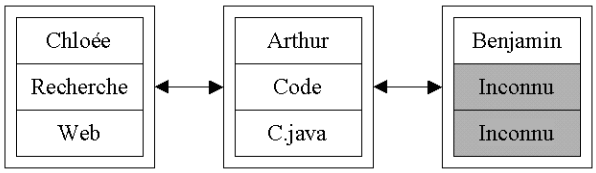

FIG. 2 – *Les nouveaux contextes d'Arthur, Benjamin et Chloée*

que Chloée est bien en train de travailler sur le code, cela depuis 2 heures.

Un quart d'heure plus tard, après avoir lancé une recompilation, Arthur s'aperçoit que le rectangle de Benjamin a changé de couleur (bleu). Cela signifie que Benjamin a fini de travailler sur l'article. En passant la souris sur le rectangle, Arthur obtient l'information que Benjamin est en train de faire une autre tâche, mais qui ne regarde pas Arthur. En cliquant sur le rectangle, une fenêtre indique à Arthur que Benjamin a arrêté son travail sur l'article il y a 5 minutes, et que la durée de la tâche a été de 1 heure et demi. Arthur préfère se concentrer sur le code qui vient de finir de compiler, et ira lire les modifications faites sur l'article plus tard.

Alors qu'il est en train de corriger un problème mineur dans le code qu'il vient d'écrire, Arthur remarque du coin de l'œil qu'un rectangle clignote dans son gestionnaire de conscience de groupe. Une fois sa ligne tapée, Arthur porte son regard sur le gestionnaire de contexte. Le rectangle qui a clignoté est celui de Chloée, qui lui aussi a changé de couleur maintenant. Arthur vérifie sur le serveur de fichier : oui, il y a bien une nouvelle version qui vient de Chloée. Arthur la récupère et commence à l'intégrer à son code. Cette information provoque un changement de couleur dans le gestionnaire de contexte de Chloée, qui sait ainsi que maintenant non seulement Arthur travaille sur le code, mais en plus sur l'objet "fichier" qu'elle vient de lui envoyer.

L'intégration ne se passe pas bien, car Chloée a oublié de transmettre un fichier. Arthur lui envoie aussitôt un mail. Lorsque Chloée le reçoit, si rapidement après qu'Arthur ait commencé l'intégration, elle sent bien que quelque chose ne va pas. Au moment de lire le mail, elle essaye mentalement de s'imaginer ce qui a bien pu se passer. Elle se souvient alors du nouveau fichier qu'elle a créé et qu'elle a oublié d'envoyer. La lecture du mail confirme cela. Une fois l'oubli réparé, Chloée commence une autre activité. Elle choisit une recherche sur le web, car elle sait qu'il sera facile de s'interrompre si Arthur lui demande de l'aide (figure 2). Elle sait aussi que le gestionnaire de contexte l'interrompra facilement, non seulement parce que la tâche est considérée comme facilement interruptible, mais aussi parce que Arthur sera prioritaire, travaillant sur un objet qu'elle a envoyé. Au lieu d'un changement de couleur ou d'un clignotement, ce sera directement une fenêtre qui jaillira à coté du gestionnaire de contexte, lorsque Arthur aura fini sa tâche d'intégration.

3 Modélisation du contexte

> Dey et Abowd [4] définissent le contexte ainsi :
> «Le contexte est l'ensemble des informations qui peuvent être utilisées pour caractériser la situation d'une entité.»

Cette définition correspond bien à notre acception du contexte. Mais le contexte est bien trop riche pour être modelé dans le cas général. C'est pourquoi nous nous placerons dans le cas particulier où l'«entité» est un membre d'un groupe virtuel collaborant sur un ordinateur à un projet du groupe.

Beaucoup de recherches ont déjà été faites en ce qui concerne la modélisation du contexte d'une personne, principalement en intelligence artificielle [1, 9], ainsi que dans l'"ubiquitous computing" [21, 17], domaine de recherche s'intéressant à l'informatique embarquée et présente dans la vie de tous les jours.

Notre modèle du contexte est basé d'une part sur ces recherches, et d'autre part sur les informations classiques citées dans la littérature comme étant les constituants de base de la conscience de groupe [16]. On peut regrouper les informations nécessaires suivant 6 catégories, résumées en 6 questions :

quoi : le type de la tâche effectuée, ainsi que les objets sur lesquels elle s'applique ;

qui : les personnes avec lesquelles l'utilisateur travaille sur cette activité, et leurs relations avec

l'utilisateur ;

quand : les dates de début et de fin de chaque activité, et leur historique dans le temps ;

où : l'endroit et la machine sur laquelle l'utilisateur travaille ;

pourquoi : l'enchaînement causal entre les tâches ;

comment : les objets utilisés pour l'accomplissement de la tâche.

Dans notre exemple, lorsqu'Arthur est en train d'intégrer le code de Chloée, son contexte comprend les informations suivantes :

quoi ?	Coder sur les fichiers du logiciel, y compris celui envoyé par Chloée.
qui ?	Entre autres, Benjamin et Chloée. Benjamin est co-auteur de l'article. Chloée est co-développeur du logiciel.
quand ?	Le moment où Arthur a commencé à coder, et l'historique de ses activités précédentes.
où ?	Sur son ordinateur de bureau, au laboratoire.
pourquoi ?	Suite à l'arrivé du fichier de code fait par Chloée.
comment ?	Avec son éditeur de texte "vi" et son compilateur.

4 Utilisation du contexte

Le contexte de travail de l'utilisateur contient les informations nécessaires aux autres membres du groupe pour comprendre ses activités et donc pour construire leur conscience de groupe.

Mais si le système envoie directement tout le contenu du contexte aux autres utilisateurs, il va les saturer d'informations pas toujours intéressantes. L'utilisateur, lui, n'a pas forcément envie que tout le monde sache avec autant de détails ce qu'il est en train de faire. La diffusion des informations constituant le contexte de cet utilisateur aux autres membres du groupe doit être adaptée en fonction de l'information et de son receveur : quelle information et avec quel niveau de détail pour quel membre.

Par ailleurs, du coté du receveur, le système de conscience de groupe ne doit pas être intrusif. La représentation des informations doit être la plupart du temps périphérique, pour ne pas gêner le travail effectif, mais doit donner accès à tous les détails nécessaires en cas de besoin. Les notifications du système lors de changement du contexte doivent être vues de l'utilisateur quel que soit sa tâche tout en restant assez peu intrusives pour ne pas perturber le travail de ce dernier. La représentation doit être adaptée en fonction des informations elles-mêmes et surtout en fonction du contexte de travail du receveur.

Le contexte de travail sert donc non seulement comme base de la conscience de groupe, mais aussi à l'adaptation de la diffusion et de la représentation des informations.

4.1 La diffusion

Chaque changement du contexte d'un utilisateur est automatiquement diffusé aux autres membres du groupe. La propagation se fait par catégories d'informations, en deux étapes : d'abord faire la liste des personnes potentiellement intéressées par cette catégorie (les récepteurs), puis, pour chaque récepteur, déterminer le niveau de détail auquel il a droit.

La liste des récepteurs est construite à partir des informations du contexte de l'émetteur, suivant des règles simples qui peuvent être adaptées par l'utilisateur. Sont considérées intéressées par toutes les catégories d'informations :

- les personnes qui travaillent sur les mêmes objets que l'utilisateur pour la tâche en cours (informations 'qui') ;
- les personnes apparaissant dans la cause de la tâche en cours (informations 'pourquoi') ;
- les personnes notées comme ayant un lien particulier avec l'utilisateur (informations 'qui').

Toutes les autres personnes du groupe sont considérées comme intéressées uniquement par les catégories 'quoi', 'quand' et 'pourquoi'.

Par exemple, si Arthur code, lorsque Chloée code aussi sur le même projet, alors elle est notée comme intéressée par toutes les catégories du contexte d'Arthur.

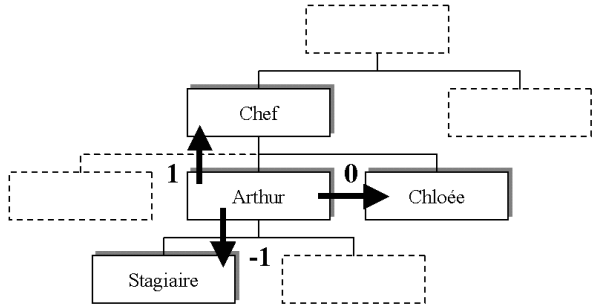

FIG. 3 – *Calcul de la distance*

Une fois qu'une personne est notée comme réceptrice de la catégorie en cours, le système détermine le niveau de détail auquel elle a droit pour cette catégorie. Pour cela, le système utilise la hiérarchie du groupe (contenu dans la catégorie 'qui'), et calcule la *distance* séparant l'émetteur du récepteur. Le signe de la distance dépend du sens de parcours de la hiérarchie.

Par exemple, Chloée étant une collègue au même niveau qu'Arthur, elle est à une distance de zéro. Par contre, le chef de l'équipe d'Arthur est à une distance de 1, tandis que le stagiaire d'Arthur est à une distance de -1.

La distance est ensuite ajustée suivant des règles simples, peu nombreuses et adaptables, pour prendre en compte certaines conditions particulières, comme par exemple :

- le récepteur travaille sur les mêmes objets que l'émetteur ;
- le récepteur apparaît dans la cause de la tâche en cours ;
- le récepteur est noté comme ayant un lien particulier avec l'émetteur.

La donnée d'une catégorie et d'une distance donne, grâce à une table modifiable par l'utilisateur, le niveau de détail auquel le récepteur a droit pour cette catégorie. L'échelle des niveaux possibles est propre à chaque catégorie, bien que toutes possèdent les mêmes niveaux minimum (aucune information envoyée) et maximum (toutes les informations de la catégorie sont envoyées).

Dans notre exemple, lorsqu'Arthur intègre le code de Chloée, cette dernière a droit au niveau maximum de détails sur la catégorie 'quoi' : elle a une distance de 0 et est la cause de la tâche en cours (figure 3).

Les informations correspondantes sont alors transmises directement au système de conscience de groupe du récepteur pour l'affichage.

4.2 La représentation

Le problème de la représentation des données se place suivant deux plans :

- le plan de la notification des changements ;
- le plan de l'accès aux données.

Le système notifie l'utilisateur à chaque fois qu'une information de la catégorie 'quoi' change chez un autre membre. Cela peut représenter un changement de tâche ou d'objet manipulé. Il utilise alors le contexte de travail de l'utilisateur pour choisir la notification la moins intrusive possible. Le système calcule d'abord la distance "d" entre l'émetteur et l'utilisateur, de la même manière que pour la diffusion. Une table, propre à la tâche en cours, donne alors, suivant la distance et l'information à reporter, le niveau d'intrusion supporté. Grâce à ce niveau, le système décide du type de notification à utiliser :

- inexistante (niveau 0) : pas de notification ; l'information est inutile d'après la tâche en cours de l'utilisateur ;
- peu intrusive (niveau 1) : un changement de couleur en vision périphérique ;
- assez intrusive (niveau 2) : un mouvement ou un clignotement en vision périphérique (la plus sensible au mouvement [5]) ;
- très intrusive (niveau 3) : l'ouverture d'une fenêtre de dialogue à coté du système.

Dans notre exemple, lorsque Arthur a fini d'intégrer le code de Chloée, cette dernière en est notifiée. Son système évalue alors sa distance avec Arthur (zéro). La table de notification correspondant à sa tâche («chercher sur le web») indique — pour une distance de zéro et la fin d'une tâche portant sur un objet partagé — un niveau 3, soit une notification très intrusive (figure 4). Une fenêtre apparaît donc à coté du système de conscience de groupe de Chloée avec le message notifiant la fin de l'intégration d'Arthur.

Info \ "d"	-1	0	1
Changement de tâche	niveau 1	niveau 1	niveau 1
Fin d'une tâche portant sur un objet partagé	niveau 2	niveau 3	niveau 2
Lecture ou modification d'un objet partagé	niveau 1	niveau 2	niveau 1

FIG. 4 – *Table de notification de la tâche «chercher sur le web»*

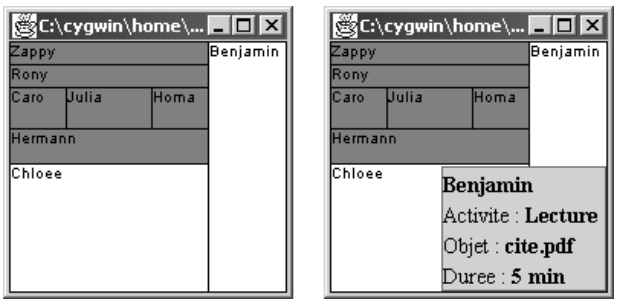

FIG. 5 – *Un exemple de TreeMap*

Pour la représentation des informations, le système utilise la catégorie 'où' du contexte du récepteur pour choisir une représentation adaptée à son environnement matériel et réseau.

Sur un ordinateur de bureau par exemple, le système utilise pour la représentation des données une visualisation *treemap* [22] (figure 5). Cette visualisation permet d'obtenir de manière périphérique une vision gestaltique de l'activité du groupe. 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 donne une information un peu plus détaillée sur l'activité en cours de la personne. Un clic sur le rectangle affiche une information complète sur la personne.

5 Construction du contexte

La collecte de toutes les informations constituant le contexte de travail d'un utilisateur peut presque toujours se faire de deux manières différentes : automatiquement, ou en demandant explicitement à l'utilisateur.

Certaines informations, notamment celles ayant un caractère social, ne peuvent pratiquement être collectées qu'en les demandant à l'utilisateur. La collecte explicite est plus facile pour le système. Elle donne une réponse plus fiable qu'une réponse calculée par une heuristique. Mais plus le nombre et la fréquence des demandes explicites du système croissent, moins le système est utilisable.

Prenons l'exemple d'ICQ, où l'état de l'utilisateur ("occupé", "ne pas déranger", ...), doit être indiqué explicitement par ce dernier. Comme l'état de l'utilisateur change souvent, au bout d'un certain temps, l'utilisateur ne prend plus la peine d'indiquer à ICQ son état réel. L'indication d'état perd alors tout son sens.

Il est important de faire en sorte que le maximum d'informations puisse être collecté de manière automatique, de façon à ce que le système ne soit ni inutile, ni une surcharge de travail pour l'utilisateur.

Cependant, le risque d'un système trop automatique est que l'utilisateur se sente pris au piège d'un système qu'il ne comprend pas, qu'il ne maîtrise plus [23, 11]. Le système doit donc toujours fournir à l'utilisateur assez de retour sur son déroulement, afin que ce dernier puisse apporter les corrections nécessaires si besoin est, et ainsi garder son sentiment de contrôle sur le système.

Le système d'exploitation ne fournit que des informations de bas niveau sur l'utilisateur :

- sa frappe clavier et ses clics souris ;
- les applications lancées, les documents ouverts, l'application et le document ayant le focus ;
- la date de début et de fin de chacun de ces événements ;
- le type du matériel utilisé, le type de la connexion réseau, si elle existe.

Si le système de conscience de groupe fait partie d'une infrastructure plus complète, celle-ci peut aussi lui fournir un certain nombre d'informations, comme par

exemple l'identité des membres du groupe, ou bien les tâches attribuées à chacun. C'est à partir de ces informations que le logiciel de conscience de groupe construit les informations constituant le contexte. Vu que la nature des informations fournies par l'infrastructure est entièrement dépendante de l'infrastructure elle-même, nous décrirons ici un système sans infrastructure. La présence d'une infrastructure ne fera que faciliter la construction,

Les informations de la catégorie 'quand' sont construites directement à partir des informations systèmes temporelles, et celles de la catégorie 'où' à partir des informations systèmes sur le matériel et le réseau. Dans notre exemple, le système d'exploitation donne comme information le type du processeur, le nom de l'ordinateur et son adresse IP. Le système reconnaît l'adresse comme appartenant au réseau du laboratoire, et à partir du nom, reconnaît l'ordinateur de bureau d'Arthur.

Mais les autres informations sont plus difficiles à calculer, car de bien plus haut niveau que les informations système. Pour les obtenir, le système utilise des réseaux bayesiens, technique qui a déjà prouvé son efficacité pour ce genre de déductions [12]. Un réseau bayesien permet de retrouver la cause la plus probable d'un ensemble d'effets observés, cela en utilisant un réseau de probabilité entre les causes et les effets.

Le réseau bayesien modélise la dépendance entre une cause et un effet par une relation affectée d'une probabilité conditionnelle, modélisant la probabilité de l'apparition de l'effet sachant la cause. Par exemple, dans la figure 6, la probabilité p_1 correspond à la probabilité de lancement du compilateur sachant que l'utilisateur est en train de coder. Ensuite, le moteur d'inférence est capable, à partir d'une liste d'effets observés, de remonter l'arbre des probabilités et de trouver la cause la plus probable. Pour plus d'information sur les réseaux bayesiens, le lecteur intéressé est invité à lire l'excellent tutoriel d'Eugène Charniak [2].

C'est la technique utilisée pour trouver les informations de la catégorie 'quoi' (figure 6). Les effets observés sont alors les informations systèmes sur les applications et les documents, l'activité clavier/souris de l'utilisateur, et les informations de la catégorie

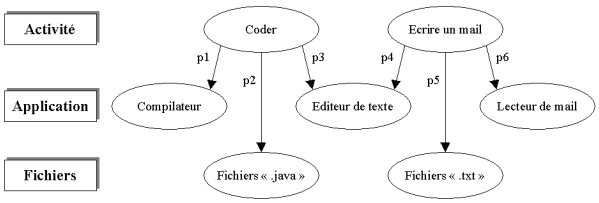

FIG. 6 – *Un exemple de réseau bayesien*

'quand'. Les informations que l'on cherche, à savoir le type de la tâche effectuée et les objets sur lesquels elle s'applique, sont modélisés dans le réseau de probabilité comme les causes de ces effets. Dans notre exemple, lorsque Arthur intègre le code de Chloée, les applications lancées sont entre autres un éditeur de texte et un compilateur ; ce sont celles qui prennent le focus le plus fréquemment à cet instant ; elles reçoivent la majorité des frappes claviers et des clics souris. Les documents ouverts sont des fichiers textes ayant l'extension ".java". Le réseau bayesien en déduit alors que la cause de tout cela est la tâche "coder sur les fichiers du logiciel".

Les informations de la catégorie 'comment' sont construites en mettant en correspondance les informations système sur les applications et les informations de la catégorie 'quoi'. Dans l'exemple, le système met en correspondance le fait que "vi" et le compilateur soient les applications majoritairement utilisées et le fait que la tâche soit "coder".

Les informations de la catégorie 'pourquoi' sont elles aussi déduites à l'aide d'un réseau bayesien. Cette fois-ci, les effets observés sont les informations des catégories 'quoi' et 'qui', et l'historique du contexte. La cause déduite est l'enchaînement causal des tâches. Par exemple, lors de l'intégration, Arthur code ; il reçoit un fichier de Chloée ; il se met à coder dans d'autres fichiers, notamment celui de Chloée. Le système en déduit qu'Arthur est dans une tâche "coder" différente, causée par le fichier de Chloée.

Enfin, les informations de la catégorie 'qui' sont divisées en deux parties. La partie concernant les personnes travaillant avec l'utilisateur sur son activité est elle aussi déduite par un réseau bayesien, les effets étant ici les informations 'quoi' et l'historique du

contexte. Dans l'exemple, la tâche "coder le logiciel" se fait sur les objets envoyés par Chloée, alors que la tâche "écrire l'article" se fait sur les objets envoyés par Benjamin. Le système en déduit que Chloée est co-développeur du logiciel et Benjamin co-auteur de l'article.

L'autre partie des informations de la catégorie 'qui' sont celles concernant les membres du groupe et leurs relations socio-professionnelles, c'est-à-dire :

- l'identité des différents membres du groupe ;
- la structure hiérarchique du groupe ;
- les personnes avec lesquelles cet utilisateur a des liens particuliers.

Comme ces informations ne changent pas souvent, et comme toutes les autres informations sont construites automatiquement, ce n'est pas une grande charge de travail que de les demander à l'utilisateur. Et c'est beaucoup plus facile et plus fiable que d'essayer de les déduire. Ce sont donc les seules informations que l'utilisateur doit donner explicitement au système. En pratique, l'utilisateur ne rentre ces informations qu'une seule fois, à l'installation du logiciel. Évidemment, il reste possible de modifier ces informations à tout instant.

6 Mise en œuvre

Le système de conscience de groupe est constitué de 4 modules (figure 7) :

- un module contenant le modèle du contexte ;
- un module chargé de la construction du contexte ;
- un module chargé de la diffusion ;
- un module chargé de la représentation.

Le premier module est le module principal. Il gère le modèle du contexte de l'utilisateur — le contexte *local* — sous la forme d'un réseau d'informations. C'est à lui que les trois autres modules s'adressent, lors de la prise de décisions, pour connaître le contexte de l'utilisateur. Il gère aussi, à partir des informations reçues des autres systèmes, un modèle du contexte de chaque autre membre du groupe — les contextes *distants*. Lorsque le contexte local change, le module principal notifie le module de diffusion, et

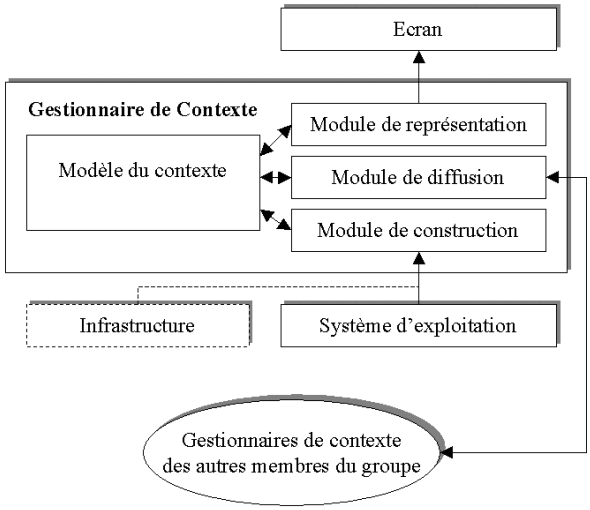

FIG. 7 – *Architecture du logiciel*

lorsqu'un contexte distant change, il notifie le module de représentation. C'est un module passif, qui ne prend aucune décision. Il ne fait que gérer les modèles de contexte.

Le second module s'occupe de la construction du modèle du contexte. Il communique avec le système d'exploitation — et avec l'infrastructure si elle est présente — pour recevoir les informations de bas niveaux. C'est avec ces informations et à l'aide de réseaux bayesiens, comme décrit dans la section 5, que ce module construit les informations de plus haut niveau qui constituent le contexte. Il les transmet alors au module principal, qui les stocke comme composant du modèle. À chaque notification d'un changement de la part du système ou de l'infrastructure, le module reconstruit l'information de haut niveau appropriée et la transmet au module principal.

Le module suivant est chargé de la communication avec les systèmes de conscience de groupe des autres membres du groupe. Il gère la diffusion des informations vers ces autres systèmes, ainsi que la réception de leurs informations. L'architecture de communication est en "peer-to-peer", sans passer par un serveur (figure 8). Chaque système connaît l'adresse IP des systèmes des autres membres du groupe grâce aux in-

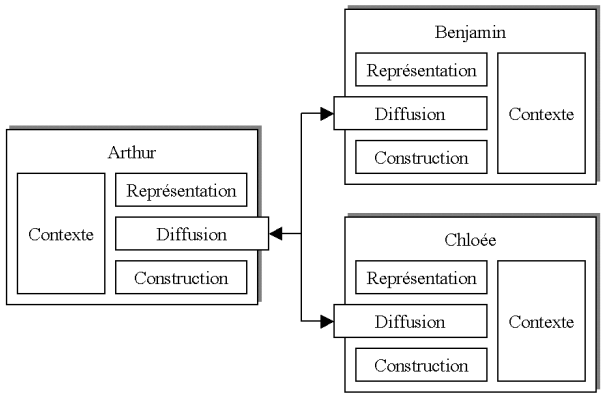

FIG. 8 – *Communications entre les systèmes*

formations rentrées par l'utilisateur dans la catégorie 'qui' du contexte (section 5). Lorsque le module principal lui indique que le contexte local a changé, il est chargé de trouver quelles informations envoyer à quel membre, et de faire la diffusion proprement dite. Il prends ses décisions en s'appuyant sur les renseignements fournis par le module principal, comme indiqué à la section 4.1. Lorsqu'il reçoit des informations venant d'autres systèmes, il les transmet directement au module principal, qui change le contexte distant correspondant.

Le dernier module gère la représentation des informations à l'attention de l'utilisateur. Cela comprend l'affichage des informations du module principal, les notifications lors de changement, et les réponses aux actions de l'utilisateur sur l'interface du système. Il est en relation étroite avec le module principal, autant pour les informations à représenter que pour déterminer la manière de les représenter. Ce module est le plus configurable des quatre, permettant à l'utilisateur une grande liberté dans le choix de la représentation.

7 Conclusion et perspectives

Il existe différents modèles de la conscience de groupe, basés sur une métaphore spatiale [19] ou physique [24]. Ces modèles permettent de structurer l'espace des informations constituant la conscience de groupe, grâce aux notions de relations, de dépendances et de distances entre les objets.

Le travail de Chen [3] est celui qui se rapproche le plus du nôtre. Chen regroupe des informations venant de sources multiples (vidéo, audio, ...) sous le terme de contexte, mais sans le modéliser ou le définir. Il utilise ensuite ces informations comme constituant de la conscience de groupe. Le contexte lui sert pour filtrer les informations à afficher sous la forme d'un «story-board».

Le système Porthole [7] a été l'un des premiers systèmes de conscience de groupe. Ses canaux vidéo permanents sur les bureaux des membres du groupe permettaient d'obtenir des informations sur la disponibilité des personnes et sur leur implication dans la tâche en cours. Par contre, le système n'apportait pas d'information sur la tâche elle-même, ni sur les objets employés.

Au contraire, les interfaces graphiques de Gutwin et Greenberg [10] représentent l'activité des membres du groupe, y compris au niveau des objets partagés. Mais cela uniquement pour des tâches collaboratives synchrones et en utilisant des outils collaboratifs adéquats.

Les systèmes de notifications d'événements comme Elvin [8] ou NESSIE [18] sont génériques et indépendants de tout outil collaboratif particulier. La définition des événements restant à la charge de l'implantation, de tels systèmes peuvent en théorie être utilisés pour la construction de n'importe quel type de conscience de groupe. Mais c'est à l'utilisateur de définir quelles informations l'intéressent, à l'aide d'une liste de souscriptions sur le serveur relayant les notifications.

Notre système apporte, par rapport à ces travaux, une conscience de groupe plus complète et plus adaptable. 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, com-

ment notifier l'utilisateur d'un changement de la manière la moins intrusive possible.

Nous sommes pour l'instant dans la phase d'implantation du prototype. Une fois ce dernier terminé, nous pourrons mettre en place une expérimentation réelle qui permettra de 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 ?

Par ailleurs, nous continuons à travailler sur la modélisation du contexte. Il faut pouvoir prendre en compte de nouvelles formes et de nouvelles sources d'informations, qui permettrons d'améliorer les inférences du moteur de réseau bayesien, ainsi que d'enrichir le contexte proprement dit. Nous étudions aussi l'apport qu'aurait l'ajout de rôles (en plus ou à la place de la hiérarchie) dans la diffusion du contexte. Enfin, nous étudions les différentes façons de représenter les informations constituant la conscience de groupe. C'est un problème d'Interaction Homme-Machine, et plus particulièrement de Visualisation d'Information. C'est pourquoi nous regardons comment utiliser les outils existants dans le domaine, comme par exemple les treemaps [22] ou bien les arbres hyperboliques [15].

Une autre direction de recherche concerne l'utilité pour l'utilisateur lui-même d'un système qui garde une trace de son contexte de travail, par exemple pour reprendre le travail plus facilement en cas d'interruption [20, 14, 13].

8 Remerciements

Ce travail a été financé par une bourse B.D.I. du C.N.R.S. et de la Région Lorraine.

Références

[1] Daniel Billsus and Michael J. Pazzani. A personal news agent that talks, learns and explains. In *Proceedings of the 4th ACM International Conference on Autonomous Agents (AGENTS-99)*, pages 268–275, Seattle, 1999. ACM Press.

[2] Eugene Charniak. Bayesian networks without tears. *AI*, 12(4):50–63, 1991.

[3] Datong Chen and Hans-Werner Gellersen. Recognition and reasoning in an awareness support system for generation of storyboard-like views of recent activity. In *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work (GROUP-99)*, pages 356–364, Phoenix, 1999. ACM Press.

[4] Anind K. Dey and 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.

[5] Alan Dix, Janet Finlay, Gregory D. Abowd, and Russell Beale. *Human-Computer Interaction, second edition*. Prentice Hall, 1998.

[6] Paul Dourish and Victoria Bellotti. Awareness and coordination in shared workspaces. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-92)*, pages 107–114, Toronto, Ontario, 1992. ACM Press.

[7] Paul Dourish and Sara Bly. Portholes: Supporting awareness in a distributed work group. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-92)*, pages 541–547, Monterey, 1992. ACM Press.

[8] Geraldine Fitzpatrick, Tim Mansfield, Simon Kaplan, David Arnold, Ted Phelps, and Bill Segall. Augmenting the workaday world with Elvin. In *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work (ECSCW-99)*, pages 431–450, Copenhagen, 1999. Kluwer Academic Press.

[9] Francesca Arcelli Fontana and Ferrante Formato. User adaptive models based on similarity. In *Proceedings of the ACM/SIGAPP Sym-*

posium on Applied Computing (SAC-00), pages 501–504, Como, 2000. ACM Press.

[10] Carl Gutwin and Saul Greenberg. A descriptive framework of workspace awareness for real-time groupware. In *Computer Supported Cooperative Work*. Kluwer Academic Press., 2001.

[11] Eric Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-99)*, pages 159–166, Pittsburgh, 1999. ACM Press.

[12] Eric Horvitz, Jack Breese, David Heckerman, David Hovel, and Koos Rommelse. The Lumière project: Bayesian user modeling for inferring the goals and needs of software users. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 256–265. Morgan Kaufmann, 1998.

[13] David Kirsh. A few thoughts on cognitive overload. *Intellectica*, 30(1):19–51, 2000.

[14] Saadi Lahlou. Attracteurs cognitifs et travail de bureau. *Intellectica*, 30(1):75–113, 2000.

[15] John Lamping, Ramana Rao, and Peter Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-95)*, pages 401–408, Denver, 1995. ACM Press.

[16] Susan E. McDaniel and Tom Brinck. Awareness in collaborative systems: A CHI 97 workshop. *SIGCHI*, 29(4):68–71, 1997.

[17] Jason Pascoe. Adding generic contextual capabilities to wearable computers. In *Proceedings of the Second International Symposium on Wearable Computers (ISWC-98)*, Pittsburgh, 1998. IEEE Computer Society Press.

[18] Wolfgang Prinz. NESSIE: An awareness environment for cooperative settings. In *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work (ECSCW-99)*, pages 391–410, Copenhagen, 1999. Kluwer Academic Publishers.

[19] Tom Rodden. Populating the application: A model of awareness for cooperative applications. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-96)*, pages 87–96, Cambridge, 1996. ACM Press.

[20] Mark Rouncefiels, John A. Hughes, Tom Rodden, and Stephen Viller. Working with "constant interruption": CSCW and the small office. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW-94)*, pages 275–286, Chapel Hill, 1994. ACM Press.

[21] Daniel Salber, Anind K. Dey, and Gregory D. Abowd. The context toolkit: Aiding the development of context-enabled applications. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI-99)*, pages 434–441, Pittsburgh, 1999. ACM Press.

[22] Ben Shneiderman. Tree visualization with treemaps: A 2-D space-filling approach. *ACM Transactions on Computer-Human Interaction*, 11(1):92–99, 1992.

[23] Ben Shneiderman. *Designing the User Interface, third edition*. Addison-Wesley, 1997.

[24] Carla Simone and Stefania Bandini. Compositional features for promoting awareness within and across cooperative applications. In *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work: The Integration Challenge (GROUP-97)*, pages 358–367, Phoenix, 1997. ACM Press.

[25] Manuel Zacklad. La théorie des transactions intellectuelles : une approche gestionnaire et cognitive pour le traitement du COS. *Intellectica*, 30(1):195–222, 2000.