Partie 10 : MDA et CSCW, Une Bibliographie

Pierre Parrend
13 Avril 2005

Table des matières

1 Si Vous êtes pressés

Notre travail se situe dans le cadre du Framework CoCSys [David05]. Celui est présenté par [David05], et expliqué dans le cadre de notre problèmatique par [MDA_CSCW_problèmatique]. Il ne sera donc pas repris ici.

1.1 Thèmes abordés

1.2 Processus MDA pour le CSCW

[Swaby99] présente un premier travail sur l'usage de MDA pour le CSCW, et fournit une vision des apports de cette approche et des problèmes posés tout à fait complète. Les manques mis en évidence dans le cadre de ce travail ont été comblé ultérieurement, par d'autres équipes.

Les points forts de l'approche orientée modèle sont donc : le développement rapide d'application, l'évolution d'application existante, l'intégration aisée de services existants, ainsi que le support d'interfaces utilisateurs variées.

1.3 Modèle de support de l'activité

[Bourguin00] ne propose pas d'étude de solution pour la réalisation de l'activité coopérative, mais se concentre sur les possibilités d'évolution de cette activité. Ce travail est donc complèmentaire à celui réalisé dans le cadre du framework CoCSys, qui lui s'attache à proposer une méthodologie de réalisation d'applications coopératives à l'aide de MDE.

L'auteur propose l'introduction de plusieurs concepts : sujet, objet, communauté, outil, conformément à la théorie de l'activité.

Il propose également l'introduction de concepts supplémentaires :

1. les rôles, qui représentent l'association entre la communauté, les sujets, et les objets. Ils sont la synthèse des règles et de la division du travail de la théorie de l'activité.
2. les tâches, qui représentent l'association entre un objet, des outils, des rôles, et les liens avec d'autres tâches.

La question se pose de la relation qu'entretiennent les tâches et les rôles au sein d'une session, ou entre des sessions différentes.

En ce qui concerne les tâches, la prudence est de mise dans leur mise en oeuvre, dans la mesure où il s'agit d'une problèmatique proche du workflow. Elles ne sont donc pas forcément pertinentes pour tous les types de groupware.

1.4 Implémentation ouverte

[Bourguin00] propose une solution pour gérer l'évolutivité du système pendant son utilisation : la mise en oeuvre de la reflexivité. Il s'inspire en celà des travaux de [Kiczales96], [Maes87] et [Kiczales91] : la mise à disposition d'un système dont les composants peuvent être manipulés (boite noire), mais également modifiés (boite blanche, ou implémentation ouverte).

La reflexivité est permise par l'utilisation du Meta-Object Protocol (MOP), qui propose l'introspection (visualisation du système par lui-même), mais également l'intercession (modification du système par lui-même).

Les modifications sont réalisables par l'utilisateur, mais peuvent être limitées en fonction des droits d'accès des utilisateurs. Il s'agit donc d'une solution moins puissante que la méta-modélisation, mais qui permet de faire évoluer le système pendant son utilisation.

1.5 Coopération de Méta-groupware

[LePallec02] propose deux outils pour la réalisation de services d'adaptation pour groupware, en mettant en oeuvre le paradigme MDE (Model Driven Engineering).

Il s'agit d'un outil devant permettre la collaboration entre groupwares hétérogènes - CAST, et d'un outil de manipulation de méta-données - RAM3.

RAM3 se distingue des autres outils par les possibilités d'extensions qu'il présente, en particulier la variation possible de l'interprétation des modèles.

1.6 Extension d'UML pour le CSCW

UML-G est une extension d'UML pour le support de la conception de groupware, et plus précisément des informations partagées [Rubart02][Rubart04].

Il ne propose donc pas de solution générale pour la conception d'un logiciel CSCW.

2 Objectif

3 Le Framework CoCSys

Le Framework CoCSys [David05] est un framework méthodologique de développement de groupware par les méthodes du Model Driven Engineering (MDE), qui est un des projets d'étude actuel du laboratoire ICTT. C'est dans le cadre de ce Framework que nous situons notre travail.

Ce document est destiné à présenter les autres travaux réalisés sur l'usage de MDE pour le CSCW. Nous ne préciserons donc pas ici les principes de CoCSys, dans la mesure où nous les présentons en détail par ailleurs [CSCW_MDA_problematique].

4 Processus MDA pour le CSCW

[Swaby99] présente la mise en oeuvre d'un processus MDE pour la réalisation de groupwares. A notre connaissance, cet article est précurseur dans ce domaine.

4.1 Résumé

[Swaby99] présente un premier travail sur l'usage de MDA pour le CSCW, et fournit une vision des apports de cette approche et des problèmes posés tout à fait complète. Les manques mis en évidence dans le cadre de ce travail ont été comblé ultérieurement, par d'autres équipes.

Les points forts de l'approche orientée modèle sont donc : le développement rapide d'application, l'évolution d'application existante, l'intégration aisée de services existants, ainsi que le support d'interfaces utilisateurs variées.

4.2 Apport

Selon l'auteur, l'introduction du CSCW dans le processus de conception d'un groupware permet de prendre en compte le dynamisme et l'hétérogénéité des équipes virtuelles.

L'objectif est double :

1. développement rapide d'applications adaptées à la collaboration concernée,
2. capacité d'évolution des applications accrue.

La nécessité d'évolution découle du besoin permanent d'adapter l'application au besoin de l'utilisateur.

Le développement orienté modèle permet donc un double gain :

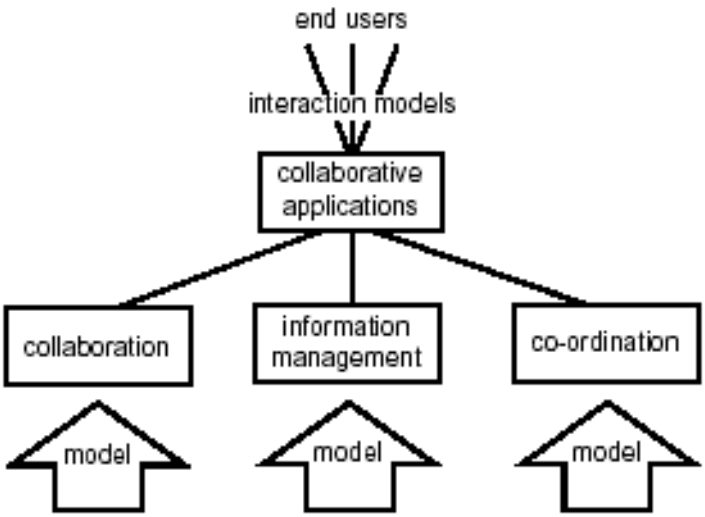
1. la création d'un modèle générique de processus, qui s'adapte aux besoins du management et la gestion des équipes,
2. l'adaptation des tâches courantes, dans l'espace de travail personnel de chaque utilisateur.

Le principe est de réutiliser autant que possible les services existants, et de fournir des fonctionnalités spécifiques à l'application au dessus de ces services.

Le développement est donc décomposé en deux domaines distincts : le développement des fonctionnalités réutilisables (les services et leur composition), et le développement de l'interaction de l'utilisateur avec le service.

L'auteur distingue nettement les domaines de gestion de l'information, de coordination et de collaboration, ainsi que les modèles d'interaction, lors de la création des modèles.

La figure suivante illustre ce processus.



Les interaction utilisateurs sont décrites par le langage DMSL (Display Metaphor Scripting Language).

L'auteur propose lui-même une évaluation de sa proposition.

Le développement d'une vue nouvelle pour le prototype réalisé est très aisée, et mois de cinq minutes sont nécessaires.

Les points forts de cette approche sont :

1. exploration rapide des idées de conception,
2. intégration dynamique de nouveaux services,
3. intégration dynamique des différents aspects collaboration, coordination, information,
4. support d'interfaces utilisateurs variées.

Les points faibles sont :

1. besoin de développeurs expérimentés,
2. manque d'abstraction de l'information,
3. manque de contrôle de session.

4.3 Limites

L'auteur propose une solution au premier problème : développement rapide d'application. L'aspect d'évolution de l'architecture est également grandement facilitée, mais pas l'adaptation des tâches courantes par l'utilisateur. Ceci est l'objet de la thèse de Bourguin [Bourguin00].

En ce qui concerne le manque d'abstraction de l'information, une solution est proposée par exemple dans [Rubart04].

En ce qui concerne le contrôle des sessions, [Moran05] propose une solution complète qui prend en compte l'intégration des utilisateurs dans la session.

4.4 Bilan

Ce premier travail sur l'usage de MDA pour le CSCW fournit une vision des apports de cette approche et des problèmes posés tout à fait complète. Les manques mis en évidence dans le cadre de ce travail ont été comblé ultérieurement, par d'autres équipes.

Les points forts de l'approche orientée modèle sont donc : le développement rapide d'application, l'évolution d'application existante, l'intégration aisée de services existants, ainsi que le support d'interfaces utilisateurs variées.

5 Modèle de support de l'activité

5.1 Résumé

[Bourguin00] ne propose pas d'étude de solution pour la réalisation de l'activité coopérative, mais se concentre sur les possibilités d'évolution de cette activité. Ce travail est donc complèmentaire à celui réalisé dans le cadre du framework CoCSys, qui lui s'attache à proposer une méthodologie de réalisation d'applications coopératives à l'aide de MDE.

L'auteur propose l'introduction de plusieurs concepts : sujet, objet, communauté, outil, conformément à la théorie de l'activité.

Il propose également l'introduction de concepts supplémentaires :

1. les rôles, qui représentent l'association entre la communauté, les sujets, et les objets. Ils sont la synthèse des règles et de la division du travail de la théorie de l'activité.

2. les tâches, qui représentent l'association entre un objet, des outils, des rôles, et les liens avec d'autres tâches.

La question se pose de la relation qu'entretiennent les tâches et les rôles au sein d'une session, ou entre des sessions différentes.

En ce qui concerne les tâches, la prudence est de mise dans leur mise en oeuvre, dans la mesure où il s'agit d'une problèmatique proche du workflow. Elles ne sont donc pas forcément pertinentes pour tous les types de groupware.

5.2 Apport

Grégory Bourguin présente, dans sa thèse [Bourguin00], l'apport de la théorie de l'activité dans le domaine du CSCW, en particulier au niveau des exigences de malléabilité. Cette malléabilité a une grande importance pour la conception des applications coopératives, car elles permettent leur modification par l'utilisateur [CSCW_services]. Cette malléabilité est supportée par des modèles comceptuels, dont l'utilisation, même s'il n'y est pas fait référence explicitement, est très proche de l'esprit du MDE.

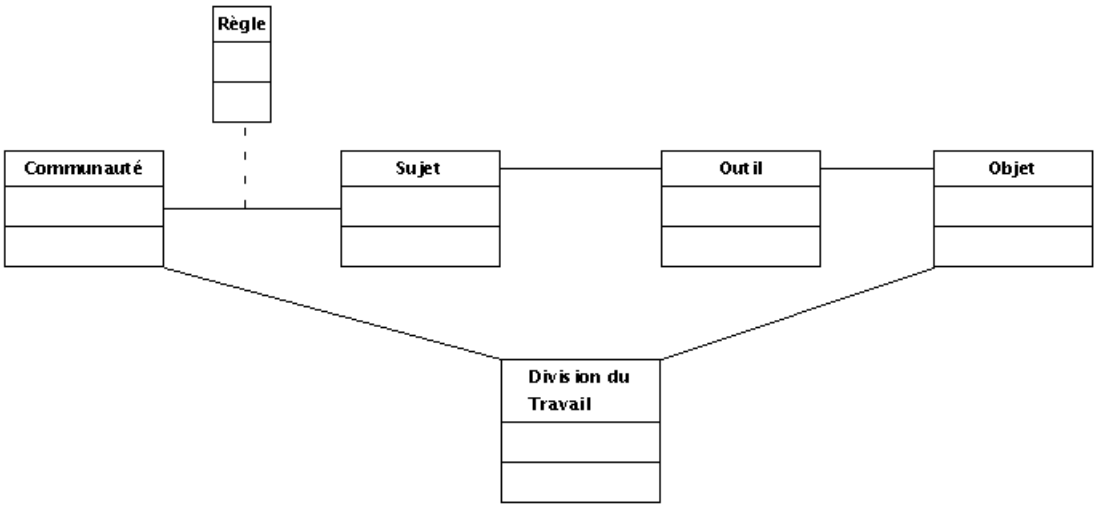
Bourguin propose un modèle conceptuel de support aux activités. Ce modèle, comme son nom l'indique, reste donc abstrait, mais permet de clarifier les besoins d'un futur modèle MDE. Il a pour objectif d'identifier les concepts et mécanismes sous-jacents qui permettront à l'environnement CSCW de fournir un support pour les activités coopératives malléables.

Les propriétés du modèles doivent être :
- compréhensible par les sujets
- compatible avec la théorie de l'activité
- en lien avec des stratégies de conception logicielle

L'auteur se concentre sur les moyens à mettre en oeuvre pour permettre aux utilisateurs de faire évoluer le support de l'activité coopérative, et non sur la réalisation de cette activité.

Les concepts fondamentaux de la théorie de l'activité sont : l'objet (motif de l'activité), sujet (qui travaille sur l'objet), outil (médiation entre l'objet et le sujet), communauté (ensemble de sujets), règles (médiatisent la relation sujet-communauté).

La figure suivante représente le modèle de la théorie de l'activité.



Le rôle Les 'règles' et la 'Division du Travail' peuvent être regroupées dans le même concept de rôle. Ceci est motivé par la familiarité des utilisateurs aux rôle, que ce soit dans le contexte informatique ou dans la vie courante. Par ailleurs, le rôle est plus intuitif dans la réalisation d'une tâche que la présence de listes de règles et de divisions du travail.

Les rôles sont composés de micro-rôles, qui contiennent chacun les droits et devoirs envers un outil.

La tâche La tâche est le composant de base de l'activité [Bedny97]. Celle-ci est donc une succession de tâches.

Dans DARE, le groupware développé par l'auteur, la tâche définit l'objet de l'activité, les outils qui vont être utilisés, les rôles explicites des sujets, ainsi que des liens avec d'autres tâches.

La tâche représente donc un patron d'activité, et contient l'expérience de la communauté. Elles peuvent contenir des sous-tâches. Les liens entre les tâches, et entre tâches et sous-tâches permettent d'organiser l'activité. Cette organisation est proche de la problèmatique du workflow, qui est un domaine de recherche à part entière. Elle ne peut donc être résolue brièvement.

5.3 Limites

On peut se poser la question si l'objectif qui est celui de l'auteur de proposer un support pour les activités malléables n'est pas susceptible de provoquer un manque dans certains domaines non directement liés à la malléabilité.

De plus, l'attention est portée sur l'évolution de l'activité plus que sur l'activité elle-même. Par conséquent, l'auteur ne prétend pas étudier en détail une solution pour la réalisation de cette activité. Il en propose toutefois un telle solution, mais dans la perspective de support de workflow, qui n'est sans doute pas assez général. En particulier, la décomposition de l'activité en tâches découle de problèmatiques workflow, et n'est pas forcément pertinente dans tous les cas.

Nous étudierons ce problème plus en détail dans la partie ' proposition d'extension'.

5.4 Bilan

L'auteur ne propose pas d'étude de solution pour la réalisation de l'activité coopérative, mais se concentre sur les possibilités d'évolution de cette activité. Ce travail est donc complèmentaire à celui réalisé dans le cadre du framework CoCSys, qui lui s'attache à proposer une méthodologie de réalisation d'applications coopératives à l'aide de MDE.

L'auteur propose l'introduction de plusieurs concepts : sujet, objet, communauté, outil, conformément à la théorie de l'activité.

Il propose également l'introduction de concepts supplémentaires :

1. les rôles, qui représentent l'association entre la communauté, les sujets, et les objets. Ils sont la synthèse des règles et de la division du travail de la théorie de l'activité.
2. les tâches, qui représentent l'association entre un objet, des outils, des rôles, et les liens avec d'autres tâches.

La question se pose de la relation qu'entretiennent les tâches et les rôles au sein d'une session, ou entre des sessions différentes.

En ce qui concerne les tâches, la prudence est de mise dans leur mise en oeuvre, dans la mesure où il s'agit d'une problèmatique proche du workflow. Elles ne sont donc pas forcément pertinentes pour tous les types de groupwares.

5.5 Proposition d'extension

Nous avons vu que le modèle proposé par G. Bourguin introduit, sans vouloir y limiter son application, des notions directement issues des problèmatiques du workflow. Ceci n'empêche nullement la conception de groupwares, tel DARE, qui est un bon exemple de la validité du modèle proposé.

Cependant, cet a priori peut avoir des conséquences sur le généralisation du modèle. En effet, celui-ci (fig 16 p 101 dans [Bourguin00]) place la tâche au centre du groupware.

Ceci ne permet pas la mise en oeuvre de solutions où la réalisation des tâches est peu formalisé, dans le cas d'organisation par un canal informel de messagerie instantannée, ou un forum, par exemple.

De plus, la restriction de la notion de communauté aux sujets travaillant sur un même objet limite la possibilité de mise en place d'équipe de projets, qui ont besoin de ressources communes (les objets), mais pour lesquelles chaque membre peut être ammené à travailler sur des objets différents, pour atteindre un objectif commun. Il est donc nécessaire de dissocier 'communauté' et 'activité'. C'est cette dernière appellation que nous conserverons.

Pour le support d'équipe de projet, le terme 'Equipe' semble approprié. Il reflète à la fois l'évolution possible des groupes de travail, et le besoin de travail en vue d'un objectif commun, sans que le travail soit réalisé systématiquement à plusieurs. Une équipe comprend donc plusieurs sujets. Elle posséde les objets nécessaires à la réalisation de ses objectifs.

La question se pose de savoir comment les membres d'une équipe peuvent travailler ensemble, sans forcément engager toute l'équipe, pour une activité donnée par exemple, ou bien comment des membres de plusieurs équipes peuvent collaborer.

Le concept qui désigne le travail en commun de plusieurs personnes est la 'Session'. Elle ne concerne pas directement les Equipes, mais regroupe des utilisateurs, quelle que soit leur équipe d'origine. Une Session peut donc accueillir des membres de plusieurs équipes, de même qu'une personne peut participer à plusieurs Sessions. Le travail réalisé au sein d'une Equipe est donc répartit entre plusieurs Sessions.

Il doit être possible de contrôler l'accès à une Session, par exemple aux membres d'une équipe donnée.

Chaque Session contient des rôles, qui sont des vues sur des activités, liées à des tâches à réaliser.

Conformément à [Bourguin00], les rôles contiennent les règles de médiation entre le sujet et l'équipe, ainsi que la division du travail. Ils peuvent exister sous trois formes :

1. implicite (ex : titre, 'Chef de projet', 'Administrateur réseau'),
2. explicite et informel (ex : liste de tâches),
3. explicite et formel (cas des rôles intégrés à un workflow).

Un rôle est composé de micro-rôles, qui indiquent les droits et devoirs de chaque sujet pour un outil donné.

Chaque Session peut contenir également plusieurs 'Activités'. Une Activité consiste en l'utilisation d'un outil. Contrairement à ce que propose Bourguin, nous ne considèrons pas qu'il soit utile de limiter une activité à la manipulation d'un objet. Toutefois, il faut que les droits d'accès et les manipulations à exécuter sur ces objets soient cohérents, et regroupé au sein d'un micro-rôle.

Si pour un même objet, plusieurs micro-rôles existent, il est possible de décomposer l'activité en tâches et en sous-tâches.

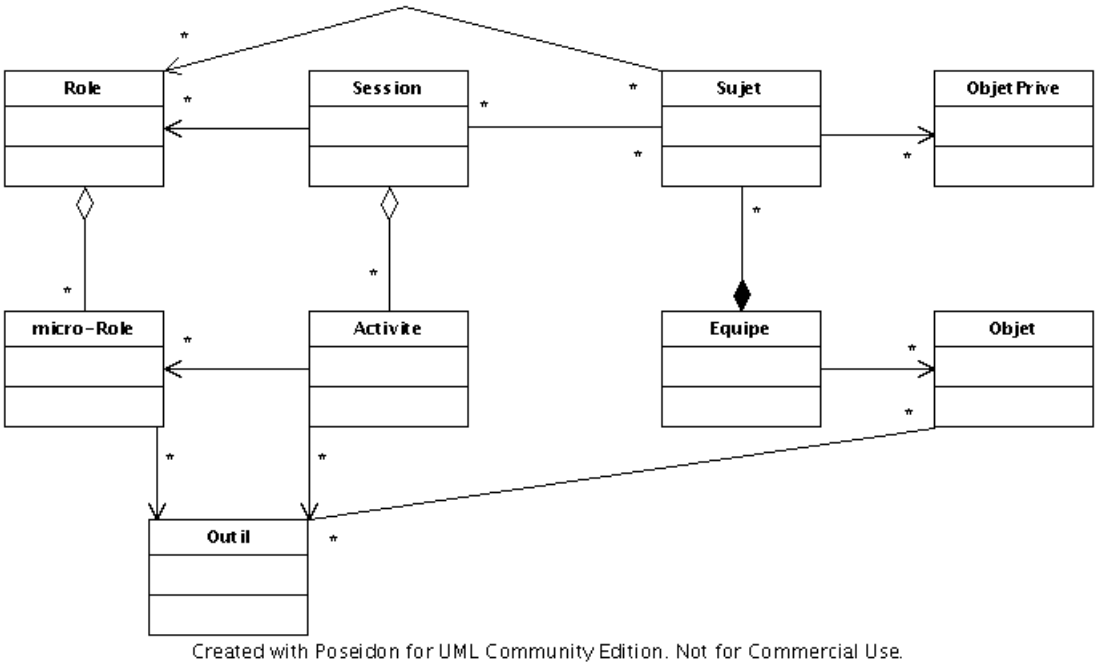
L'existence de tâches et de sous-tâches rend possible la création de système de workflow pour organiser le travail de l'Equipe, mais sans le rendre obligatoire.

Ceci répond au besoin de souplesse des groupwares, qui doivent pouvoir s'adapter à différentes situations collaboratives.

Tous les éléments intervenants dans la collaboration ont été présenté, à l'exception des utilisateurs eux-même. Ceux-ci sont représentés par les Sujets. En effet, chacun a besoin d'être représenté pour lui-même dans le système, afin de permettre la continuité entre le monde réel et le monde informatique (seamlesssness). Le Sujet est donc un avatar - représentation informatique d'une personne réelle. Il permet de conserver des données personnelles, l'historique des interactions, la configuration propre à chacun.

Ceci n'empêche naturellement nullement la présence de Sujets collectifs, qui représentent un rôle (par exemple 'invité', qui a le droit d'observer ce qui se passe mais pas d'intervenir).

La figure suivante synthétise le méta-modèle proposé, qui peut-être utilisé pour représenter une application collaborative.



6 Implémentation ouverte

6.1 Résumé

[Bourguin00] propose une solution pour gérer l'évolutivité du système pendant son utilisation : la mise en oeuvre de la reflexivité. Il s'inspire en celà des travaux de [Kiczales96], [Maes87] et [Kiczales91] : la mise à disposition d'un système dont les composants peuvent être manipulés (boite noire), mais également modifiés (boite blanche, ou implémentation ouverte).

La reflexivité est permise par l'utilisation du Meta-Object Protocol (MOP), qui propose l'introspection (visualisation du système par lui-même), mais également l'intercession (modification du système par lui-même).

Les modifications sont réalisables par l'utilisateur, mais peuvent être limitées en fonction des droits d'accès des utilisateurs. Il s'agit donc d'une solution moins puissante que la méta-modélisation, mais qui permet de faire évoluer le système pendant son utilisation.

6.2 Apport

[Bourguin00] se base sur les travaux de Kiczales [Kiczales96] pour proposer la réalisation d'un framework non pas en 'boite noire', c'est à dire mettant à disposition des fonctionnalités figées, mais en 'implémentation ouverte' (open implementation), afin de permettre aux développeurs de faire évoluer le framework lui-même.

Plus précisément, un module logiciel doit être accessible de trois manières différentes :

1. utilisation du module tel qu'il est proposé (boite noire),
2. contrôle et modification de la stratégie d'implémentation si besoin (implémentation ouverte).

Ces deux approches doivent coexister.

L'auteur s'inspire également fortement des méta-systèmes de Maes [Maes87], qui apporte la reflexivité aux systèmes informatiques par l'introspection, d'une part - c'est à dire la possibilité pour un système de s'examiner - et l'intercession, d'autre part - c'est à dire la possibilité pour un système de se modifier lui-même.

La réalisation de l'intercession est possible grâce à l'usage du langage Smalltalk, plus puissant que Java sur ce point. Le système peut donc évoluer pendant son utilisation : c'est la malléabilité.

La réalisation de la réflexivité est basé sur le Meta-Object Protocol (MOP) [Kiczales91].

6.3 Limite

L'approche présentée fournit une méthodologie de développement de programme à implémentation ouverte, qui, s'il s'agit de méta-programmation, n'est pas à proprement parler de la méta-modélisation, en ce que les éléments manipulés sont des morceaux de code, et non des modèles comme le propose l'approche MDE.

La méta-programmation s'applique a représenter un système générique, et à le faire évoluer. Elle nécessite l'adjonction de connaissances concernant le domaine.

La méta-modélisation a pour objectif de modéliser le domaine et les actions de l'utilisateur, et se concentre donc sur l'aspect fonctionnel. Elle nécessite la projection de ces modèles sur une architecture particulière.

La différence entre les deux est donc essentiellement le point de vue adopté : système pour la méta-programmation, et utilisation/domaine pour la méta-modélisation.

La méta-programmation dépend du méta-système proposé. Elle est donc fortement dépendante de l'existant.

L'ambition de la méta-modélisation est de proposer une méthodologie complète, qui permette la réalisation d'applications adaptées aux comportements de l'utilisateur (par le biais des scénarii), tout en maximisant la réutilisation de briques fonctionnelles : elle doit donc apporter beaucoup plus de souplesse en ce qui concerne les caractéristiques du nouveau système.

La méta-programmation restreint les possibilités d'innovation au méta-système. La méta-modélisation permet, en théorie, de s'affranchir de cette restriction.

La méta-programmation est donc une solution intermédiaire entre programmation objet et méta-modélisation. Elle est cependant plus puissante que la méta-programmation en ce qui concerne la malléabilité.

6.4 Bilan

La méta-programmation propose une solution d'utilisation souple de l'application. La méta-modélisation étend cette souplesse à l'ensemble du cycle de vie, de la phase de conception à l'évolution, mais perd la souplesse d'utilisation de l'application.

Une solution courament proposée, entre autre par le laboratoire ICTT, pour remédier à cette perte de souplesse, est la mise en oeuvre de systèmes de workflow flexibles, qui permettent également de faire évoluer l'activité en cours. Une question se pose donc :

1. Quel sont les points forts et les points faibles des deux approches d'évolution des activités à chaud, malléabilité et worlflow flexibles ?

Par ailleurs, pour exploiter la modification à chaud dans notre approche, c'est à dire en utilisant le langage Java, il sera indispensable de clarifier la question suivante :

1. Quelles sont les limites de la réflexivité en Java ? Quels sont les apports respectifs de MOP et des aspects ? Smalltalk est-il réellement plus puissant, comme le prétend G. Bourguin ?

Les questions concernant la reflexivité n'étant pas directement liée à l'approche MDE, il est probable que nous n'aurons pas le temps de les détailler.

En résumé En résumé, Bourguin propose donc un méta-modèle de groupware, inspiré de la théorie de l'activité, et une solution d'implémentation de ce modèle qui supporte la malléabilité, par la reflexivité.

Il ne propose pas de méthode de conception de groupware.

7 Coopération de Méta-groupware

7.1 Résumé

[LePallec02] propose deux outils pour la réalisation de services d'adaptation pour groupware, en mettant en oeuvre le paradigme MDE (Model Driven Engineering).

Il s'agit d'un outil devant permettre la collaboration entre groupwares hétérogènes - CAST, et d'un outil de manipulation de méta-données - RAM3.

RAM3 se distingue des autres outils par les possibilités d'extensions qu'il présente, en particulier la variation possible de l'interprétation des modèles.

7.2 Apport

Un travail important concernant l'utilisation des outils MDE dans le cadre de développement de groupwares a été réalisé par [LePallec02].

L'auteur se concentre sur la réalisation d'un mécanisme d'interopérabilité entre groupwares, mettant en oeuvre les principes du MDE. Les groupwares considérés sont les workflow, d'une part, et des systèmes plus ouverts du types DARE (voir les paragraphes précédents [Bourguin00]).

L'auteur présente les caractéristiques des systèmes de workflow, ainsi que les approches existantes pour y apporter de la flexibilité. Son travail n'est donc pas directement pertinent dans le cadre de notre problèmatique.

De plus, sont considérés comme groupwares tous les systèmes informatiques d'entreprise, sans prise en compte des recherches spécifiques sur les caractéristiques des applications collaboratives telles qu'elles sont définies par la communcauté de recherche CSCW, et présentées par exemple dans [Borghoff00].

En ce qui concerne la méta-modélisation, L'auteur s'intéresse à l'opérationalisation des modèles : comment passer du modèle au code exécutable. Une application mettant en oeuvre des méta-modèles doit proposer l'accès informatique, sous forme de classes instanciables, et un référentiel (ou dépot de modèles) qui contiennent les modèles existants.

Un problème abordé est celui de la conception du niveau M0 de la classification OMG, c'est à dire le niveau d'implémentation (par opposition aux niveau M1 : classes - M2 : méta-modèle - M3 : méta-méta-modèle). Sa conception nécessite d'expliciter au sein du méta-modèle des préocupations d'implémentation. Elle est nécessaire pour permettre la génération de code correspondant aux modèles.

L'apport de l'auteur se situe plus dans les outils réalisés : CAST, pour la réalisation de services d'adaptation, et RAM3, pour la manipulation de méta-données, quand dans la bibliographie proposée, qui n'apporte que peu de perspectives nouvelles pour notre problèmatique.

7.3 Limites

Ce travail est basé sur les Workflows, qui ne sont qu'un outil parmis d'autres dans le cadre de groupwares.

Il n'y a pas de données précise sur ce qu'est un méta-groupware (groupware générique), mais un travail autour des méta-groupware pour leur permettre de coopérer,

Cette thèse a aboutit à la réalisation d'un outil de manipulation de méta-modèles, RAM3, alors que d'autres ont été réalisés parallèlement. Toutefois, RAM3 est spécifique au CSCW, et donc sans doute plus adapté que d'autres outils, dans la mesure où il permet le support de la manipulation de méta-modèles et le support de la réflexivité. Il présente en outre l'avantage d'être disponible, alors que d'autres sont en cours de réalisation et donc non utilisable pour le moment.

Par ailleurs, la problèmatique d'opérationnalisation a été traitées dans de nombreux autres travaux, parrallèlement à cette thèse, et des implémentations largement diffusées, et donc proposant une meilleure compatibilité, existent (par exemple, les outils MOF d'Eclipse, EMF). L'opérationnalisation n'est donc plus un problème majeur, même si l'on peut souligner l'adéquation des travaux de l'auteur avec le domaine du CSCW, que l'on ne retrouve pas dans d'autres outils.

Les caractéristiques de l'outil RAM3 sont :
- environnement multi-fenêtré,
- prototypage,
- instanciation de modèles,
- extensibilité.

Il n'est donc pas possible d'utiliser cet outil à l'intérieur d'un programme Java, ce qui limite ses possibilités d'utilisation, et le rend comparable à d'autres outils disponibles. L'extensibilité, c'est à dire l'utilisation dynamique des méta-modèles et la possibilité de modifier leur interprétation (lors de la génération de code), est un plus par rapport aux autres outils.

7.4 Bilan

Le travail de [LePallec02] n'a pas d'influence direct sur notre travail, mais il en est très complémentaire. D'autres outils plus génériques ont été développés, simultannément à sa thèse, qui bénéficient de plus de visibilité, mais sont moins adaptés à la problèmatique traitée.

Par ailleur, on peut déplorer le manque de précisions sur ce qu'est un méta-groupware, ce que nous nous proposons de faire : l'auteur se limite à la présentation des workflows, du moins dans la partie bibliographique, et n'argumente pas la généralisation de sa démarche à des systèmes coopératifs génériques.

L'extensibilité n'est pas a priori utile dans le cadre de notre problèmatique.

Questions :

1. L'auteur se conforme-t-il à son objectif de proposer une solution d'intéropérabilité pour les workflow, mais également des systèmes plus génériques du types DARE ?

8 Extension d'UML pour le CSCW

8.1 Résumé

UML-G est une extension d'UML pour le support de la conception de groupware, et plus précisément des informations partagées [Rubart02][Rubart04].

Il ne propose donc pas de solution générale pour la conception d'un logiciel CSCW.

8.2 Apport

Le laboratoire IPSI (Integrated Publication and Information Systems) de l'institut Frauenhofer travail essentiellement dans le domaine de l'enseignement à distance. Certains de leurs travaux concernent de manière plus générique le domaine du CSCW.

[Rubart04] présente un profil UML dédié aux groupwares : UML-G. En particulier, il supporte la modélisation de données partagées. C'est un besoin fondamental des applications collaboratives, même si c'est naturellement loin de couvrir l'ensemble des caractéristiques des groupwares.

[Rubart02] présente plus précisément ce que doit être UML-G. Les besoins identifiés sont :

RD1 : représentation de l'information partagée,

RD2 : représentation de l'information partagée persistante,

RD3 : représentation des notifications des changements de l'information partagée,

RD4 : représentation du contrôle d'accès à l'information partagée, en fonction des utilisateurs,

RD5 : représentation de la possibilité de bloquer les informations partagées, indépendamment de l'utilisateur (locking),

RD6 : modèles partagés d'acteurs et rôles.

Le tableau suivant présente les ajouts à UML qui permettent de répondre à xces besoins.

Nom	Type	Complète	Description
<<shared >>	stéréotype	Elément	instances accessibles par tous les utilisateurs
<<sharedRole>>	stéréotype	Class	Sous-type de <<shared>>
<<sharedActor>>	stéréotype	Class	Sous-type de <<shared>>
{access-controlable}	Tag booléen	<<shared>>	mécanismes de contrôle d'accès possibles
{lockable}	Tag booléen	<<shared>>	blocage d'instance possible
{observable}	Tag booléen	<<shared>>	notification de changement possible

Notez que le support pour la stratégie de gestion de concurrence (optimiste ou pessimiste) n'a pas encore été formalisé.

8.3 Limites

UML-G se concentre sur les problématiques de partage et de distribution de données. Il ne concerne donc pas le groupware dans son ensemble, mais les données que les utilisateurs manipulent. On peut noter toutefois que ces données sont variées : processus, équipes et objets de la collaboration en font partie.

8.4 Bilan

UML-G permet de préciser les besoins en ce qui concerne les informations partagées. Il ne propose pas de solution générale pour la conception d'un logiciel CSCW, mais concerne spécifiquement la gestion de données.

9 Perspectives

10 Pour notre travail

10.1 Apport

Selon [Swaby99], les points forts de l'approche orientée modèle sont donc : le développement rapide d'application, l'évolution d'application existante, l'intégration aisée de services existants, ainsi que le support d'interfaces utilisateurs variées.

Ceci correspond à notre conception.

10.2 Questions

Plusieurs questions apparaissent :

1. Quels sont les besoins auxquels les sessions doivent répondre ?
2. Comment sont-elles liées aux rôles et aux tâches, tels qu'ils sont définies par [Bourguin00] ?

11 Bibliographie

[**Bedny97**] Bedny G. Meister D., The Russian Theory of Activity, Current applications to Design and Learning, lawrence Erlbaum Associates Publishers, 1997, 430p., cité par [Bourguin00].

[**Borghoff00**] Computer Supported Cooperative Work, Introduction to distributed applications, U.M. Borghoff, J.H. Schlichter, Springer-Verlag, 2000, 529 p.

[**Bourguin00**] Un support informatique à l'activité coopérative fondé sur la théorie de l'activité : le projet DARE, Grégory Bourguin, thèse de doctorat, 2000.

[**CSCW_MDA_problematique**] CSCW et MDA : une Problèmatique, Pierre Parrend, 2005.

[**CSCW_services**] CSCW : les Services, Pierre Parrend, 2005.

[**David05**] Bertrand David, René Chalon, , Olivier Delotte, Model-Driven Engineering of Cooperative Systems, 2005.

[**Kiczales91**] Kiczales G., Bobrow D.G., Des Rivieres J., The Art of the Metaobject Protocol, MIT press, August 1991, 335 p., cité par [Bourguin00].

[**Kiczales96**] Kiczales G. Beyond the black Box : open implementation, IEEE software, January 1996, cité par [Bourguin00].

[**LePallec02**] Des services d'Adaptation de modèles pour la coopération de méta.systèmes, Xavier LePallec, thèse de Doctorat, décembre 2002.

[**Maes87**] Maes P., Computational reflexion, Ph.D. Thesis, V.U.B., Brussels, 1987, cité par [Bourguin00].

[**Moran05**] Support pour la collaboration potentielle et réelle dans la cocception des collecticiels répartis, thèse de l'INPG Grenoble, Alberto Leopoldo Moran y Solares 20.01.2005.

[**Rubart02**] Jessica Rubart, Peter Dawabi, Towards UML-G : a UML-Profile for modeling Groupware, in Groupware : Design, Implementation and Use, 8th International Workshop, CRIWG 2002, p 93-113, 2002.

[**Rubart04**] Shared data modeling with UML-G Jessica Rubart and Peter Dawabi, International Journal of Computer Applications in Technology, Volume 19, Nos. 3/4, 2004, p 231-243.

[**Swaby99**] Model-based construction of collaborative systems M A Swaby, P M Dew and P J Kearney, BT Technol J Vol 17 No 4 October 1999.