Un système de patterns dédié à l'analyse, à la conception et à l'implémentation des Systèmes d'Information Coopératifs

Vincent COUTURIER

Equipe MODEME
IAE - Université Jean Moulin Lyon3 - 6 cours Albert Thomas
BP 8242 - 69355 Lyon cedex 08
Tél. 04 78 78 71 58 - Fax. 04 78 78 77 50
vincent.couturier@univ-lyon3.fr

RÉSUMÉ. Cet article s'intéresse aux systèmes d'information coopératifs (SIC), systèmes complexes permettant l'utilisation coordonnée de sources d'information hétérogènes et la résolution des conflits qui en découlent, et a pour objectif de présenter une approche d'ingénierie de ces systèmes basée sur la réutilisation de patterns. Les patterns constituent des solutions génériques à des problèmes fréquemment rencontrés. Les patterns que nous avons conçus couvrent toutes les phases (analyse, conception et implantation) du cycle de développement d'un SIC et permettent d'accélérer et de faciliter ce développement. Les patterns proposés sont tout d'abord des patterns de domaine, réutilisables lors des phases d'analyse et de conception des SIC. S'ajoutent à ces patterns des patterns de support technique génériques constituant des structures réutilisables dédiées à l'implantation des entités spécifiées dans les patterns de conception de coopération et décrivant comment construire une application. Enfin, des patterns de support d'utilisation guident le concepteur lors de la réutilisation des patterns précédents.

ABSTRACT. This paper deals with Cooperative Information Systems (CIS) which are complex systems allowing the use of heterogeneous information systems and conflict resolution. It describes a new approach based on pattern reuse that facilitates the engineering of these systems. Patterns are generic solutions to problems frequently occurring. Our patterns cover all the development stages (analysis, design and implementation) and contribute to the rapid development of CIS. They are firstly domain patterns reusable during analysis and design stages of the process of CIS engineering. Technical support patterns form reusable structures dedicated to the implementation of entities specified in our cooperation design patterns and describe how to build an application. Finally, utilization support patterns form guidelines helping designers to reuse former patterns.

MOTS-CLÉS: Systèmes d'Information Coopératifs (SIC), coopération de systèmes d'information, patterns logiciels, réutilisation.

KEYWORDS: Cooperative Information Systems (CIS), information system interoperabiliy, software patterns, reuse.

Catégorie Chercheur

1. Introduction

Les besoins en matière d'information ont beaucoup évolué ces dernières années et ont rendu nécessaire le partage des données entre systèmes d'information. Le partage des données n'est certes pas une idée nouvelle mais s'est démultiplié et complexifié avec l'avènement de l'Internet. Parallèlement, les recherches axées sur le développement de S*ystèmes d'Information Coopératifs* (SIC), i.e. systèmes permettant de résoudre les conflits qui apparaissent au niveau de la représentation hétérogène de l'information, se sont accrues.

Cet article présente un système de patterns visant à faciliter le développement des SIC. Un pattern décrit les abstractions utilisées par des experts concepteurs ou programmeurs et apporte ainsi des solutions dans diverses phases du développement d'une application. Notre objectif est de faciliter la construction des modèles d'analyse, d'architecture et de conception du SIC à développer et faciliter son implantation par réutilisation de composants de type pattern.

La section suivante explicite le concept de système d'information coopératif et justifie l'utilisation de patterns pour faciliter leur développement. La notion de pattern est plus précisément présentée en section 3. La section 4 développe le système de patterns conçu. Enfin, la section 5 présente son expérimentation.

2. Les Systèmes d'Information Coopératifs (SIC)

2.1. Définition et typologie des systèmes d'information coopératifs

Un système d'information coopératif peut être défini comme un « ensemble de composants plus ou moins autonomes, souvent préexistants qui travaillent de manière synergique en échangeant information, expertise et en coordonnant leurs activités » (Boulanger *et al.*, 1997). Dans le cadre de la coopération de sources d'information hétérogènes, un SIC vise, plus précisément, à permettre l'utilisation conjointe de ces sources et résoudre les différents conflits inhérents.

La plupart des SIC existants peuvent être classés selon deux familles ou *approches* : l'*approche fédérée* et les *approches à base de médiation*.

L'approche fédérée (ex. de SIC fédérés : MRDSM (Litwin, 1988), Mind (Dogac *et al.*, 1995),…) repose sur l'intégration. Chaque SI exporte un schéma dans un modèle pivot, les différents schémas d'export étant alors intégrés dans un (ou plusieurs schéma(s) fédéré(s). Un schéma fédéré permet ainsi un accès uniforme aux données partagées. Cette approche ne fonctionne cependant que si le nombre de SI impliqués est limité.

Les *approches à base de médiation* reposent sur deux composants, le médiateur et le wrapper. Le médiateur simplifie, abstrait, réduit, combine et décrit les données et est chargé des traitements permettant à l'utilisateur d'obtenir des informations extraites des SI locaux. Il permet principalement de résoudre les conflits sémantiques (i.e. conflits de domaine de définition, conflits dans l'interprétation d'une même donnée). Le wrapper fournit une interface d'accès homogène aux sources d'information et résout les conflits de modèle de données et de langage d'interrogation en présentant les données dans le modèle de médiation. On distingue deux types de médiation : la *médiation de schémas* (ACSIS (Dubois *et al.*, 2002), Tsimmis (Garcia-Molina *et al.*, 1997),…) qui construit au préalable une base d'informations prenant en compte les SI participants pour permettre au médiateur de faire son travail d'intégrateur et la *médiation de contextes* (InfoSleuth (Nodine *et al.*, 2000), DILEMMA (Jouanot, 2001),…) qui repose sur l'intégration dynamique des informations en fonction du contexte de l'application ou de l'utilisateur.

Les systèmes d'information coopératifs nécessitent d'être spécifiés et développés ; la notion d'ingénierie des SIC est présentée en section suivante.

2.2. Ingénierie des systèmes d'information coopératifs

L'*ingénierie des systèmes d'information coopératifs* (de type sources d'information) est un « domaine de recherche qui à trait au développement systématique de solutions interopérables pour les systèmes hétérogènes et autonomes comprenant à la fois des bases de données mais aussi d'autres types de sources d'information provenant de domaines d'application variés » (Conrad *et al.*, 2002). Elle vise ainsi à construire des solutions réutilisables dédiées à la coopération de sources d'information. Pour Fernandez & Zhao (Fernandez *et al.*, 2000), cette tâche d'ingénierie est délicate car elle repose sur une double complexité : (1) les SIC sont des systèmes très complexes à modéliser (connaissances) et (2) l'ingénierie de tels systèmes est elle-même complexe. En outre, ils remarquent que leur ingénierie ne s'appuie pas, pour la plupart des cas, sur des techniques de réutilisation qui ont fait leur preuve (framework, pattern,….) et qui permettraient de faciliter celle-ci. Il nous semble nécessaire d'utiliser des techniques de réutilisation éprouvées favorisant l'analyse, la conception et l'implémentation des SIC. Nous proposons ainsi de concevoir et de réutiliser des patterns pour prendre en compte la complexité des SIC et de leur développement. Le concept de pattern est présenté en section suivante.

3. Le concept de Pattern

Sur la base des travaux de Christopher Alexander (Alexander, 1979), les patterns ont été utilisés et adaptés à différents domaines d'application dont le génie logiciel et la conception de systèmes d'information. Ils permettent la description d'éléments de solutions réutilisables ainsi que la capitalisation des connaissances et du savoir-

faire. Un pattern comporte typiquement la description d'une partie d'un système et la façon de la construire ; il décrit ainsi les abstractions utilisées par les experts concepteurs ou programmeurs (Coplien, 1998). Les patterns peuvent être spécifiés individuellement dans un catalogue ou définis dans une structure plus complexe et mieux formalisée que constituent les systèmes ou langages de patterns.

L'engouement pour les patterns a donné naissance à une littérature abondante dans laquelle différents types de patterns sont décrits avec une terminologie ne permettant pas toujours de percevoir les recouvrements entre les concepts.

Les *patterns d'analyse* (Coad, 1992) ont pour but de faciliter et guider les étapes de la phase d'analyse. Ils permettent d'obtenir un modèle conceptuel de qualité i.e. évolutif et plus compréhensible et décrivent des structures génériques applicables et réutilisables. Les *patterns de conception* (Gamma *et al.*, 1995) donnent un nom, isolent et identifient les principes fondamentaux d'une structure générale pour en faire une ressource utile et réutilisable lors de la conception de systèmes. Par exemple, lors de la conception d'un système orienté objet, un pattern de conception va décrire les classes et les instances intervenant dans la solution, leurs attributs, leurs rôles, les modalités de collaboration et la contribution de chaque élément décrit à l'obtention de la solution ciblée dans un contexte particulier.

Les *patterns d'implantation*, ou idiomes, sont des patterns de bas niveau qui permettent de définir les bonnes façons de programmer dans un langage particulier (Coplien, 1998). Ils sont liés aux technologies choisies par le programmeur.

Les *patterns de domaine* (Fowler, 1997) se différencient des patterns précédents par une approche par domaine d'application. Ils sont principalement utilisés dans l'ingénierie de domaine : ils sont utiles pour construire un modèle d'analyse. Fowler considère ainsi que les patterns de domaine doivent représenter la manière dont les gens abordent leur métier, indépendamment de tout système informatique. Il leur associe également des *patterns de support* qui traitent « des problèmes de construction de systèmes informatiques autour des patterns de domaine ». Ces patterns décrivent comment utiliser les patterns de domaine et comment les appliquer afin de les rendre concrets.

Nous présentons dans la section suivante les différents types de patterns dédiés à l'ingénierie des systèmes d'information coopératifs que nous avons spécifiés et donnons des exemples de tels patterns.

4. Patterns dédiés à l'ingénierie des systèmes d'information coopératifs

Parmi les patterns dédiés à l'ingénierie des SIC conçus, figurent les *patterns de coopération*. Ces patterns permettent d'élaborer, après réutilisation, l'ensemble des modèles du SIC à concevoir, à différents niveaux d'abstraction. Ce sont des patterns de domaine, i.e. portant sur un domaine spécifique, celui de la coopération de systèmes d'information.

Aux patterns de coopération s'ajoutent des *patterns de support de coopération* qui s'appliquent quel que soit le domaine d'application et traitent des problèmes de construction de systèmes informatiques à partir des patterns de coopération. Ces patterns décrivent comment utiliser les patterns de coopération et comment les assembler afin d'obtenir l'architecture ciblée.

4.1. Patterns de coopération

Les patterns de coopération sont des patterns de domaine[1] centrés sur les phases amont du processus de développement logiciel, i.e. les phases d'analyse et de conception. Adaptés à un domaine spécifique, nos patterns de coopération permettent ainsi de limiter le problème de la « généricité et de l'abstraction excessives de la plupart des patterns d'analyse et de conception existants » (Port, 1998). Ces patterns sont des collections de classes qui représentent des partitions bien définies (encapsulées, aux frontières claires) d'un domaine particulier. Ils aident le concepteur à aborder les systèmes d'informations coopératifs complexes d'une manière simple sans qu'il soit nécessaire d'en étudier les détails d'implantation.

Les patterns de coopération sont, d'une part, utilisés lors de la phase d'analyse et, d'autre part, lors de la phase de conception des SIC.

4.1.1. Patterns de coopération dédiés à l'analyse des SIC

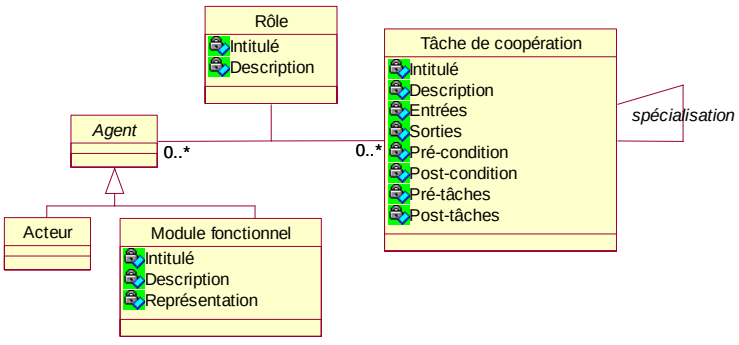
Les patterns de coopération dédiés à la phase d'analyse des SIC sont les *patterns d'analyse de coopération.* Ils mettent en jeu les connaissances de domaine détenues par les concepteurs de SIC (i.e. principalement les chercheurs du domaine de la coopération de systèmes d'information). Ils permettent de définir un vocabulaire commun propre au domaine de la coopération, classifient et organisent les connaissances du domaine afin de favoriser leur réutilisation.

Les patterns d'analyse de coopération identifient et spécifient les aspects fonctionnels du SIC, i.e. les entités du SIC en termes de classes et de relations entre les classes et fournissent des fragments de modèle d'analyse (modèle de domaine) pour représenter les SIC. Le concepteur du SIC pourra alors réaliser le modèle de son système en adaptant le fragment de modèle de domaine fourni dans la solution de chaque pattern. L'ensemble des patterns d'analyse de coopération représente un modèle d'analyse global décrivant toutes - ou si ce n'est toutes, une grande partie - les entités du SIC (structure) et leurs relations. Les solutions des patterns d'analyse permettent ainsi de constituer le modèle d'analyse de tout SIC.

Six patterns d'analyse ont été conçus et forment un sous-système de patterns, appelé *système de patterns d'analyse de coopération* (Couturier, 2005). Le pattern

[1] Notre définition d'un pattern de domaine étend celle de Fowler (Fowler, 1997) seulement limitée aux patterns d'analyse liés à un domaine d'application. Nous qualifions ainsi de pattern de domaine tout pattern d'analyse ou de conception spécifique à un domaine d'application.

"Modélisation des types de conflit" catégorise et précise les définitions des différents conflits usitées dans le système d'information coopératif à développer ; il permet également d'indiquer quels types de conflits seront traités. Le pattern *"Modélisation des sources d'information gérées"* permet de représenter le modèle et le type des sources d'information (BD…) prises en charge. Le pattern *"Modélisation du référentiel de coopération"* représente le référentiel de coopération - modèle et type (i.e. ontologie, schéma de médiation,…) - utilisé au sein du SIC. Le pattern *"Modélisation des tâches de coopération effectuées par les agents"* (Cf. tableau 1) modélise les tâches de coopération que le SIC effectue ainsi que les agents (modules fonctionnels, individus,…) intervenant lors de ces tâches. Le pattern *"Modélisation des tâches de coopération effectuées par les agents dans le cadre des approches à base de médiation de contextes et de médiation de schémas"* est un raffinement du pattern précédent dans le cadre de la conception d'un SIC de type médiation de schémas ou médiation de contextes. Enfin, le pattern *"Modélisation des processus de coopération"* permet de représenter les processus de coopération, c'est-à-dire les regroupements ordonnés de tâches de coopération spécifiées dans le SIC.

Interface
Nom
Modélisation des tâches de coopération effectuées par les agents
Classification
Pattern d'analyse de coopération
Objectif
Ce pattern permet de modéliser les tâches de coopération que le système d'information coopératif effectue et, éventuellement, les différents agents intervenant lors de ces tâches.
Solution
Modèle

Participants
<u>*Tâche de coopération :*</u> définit un événement de coopération à traiter. Une tâche de coopération peut-être décomposée en sous-tâches représentant ainsi les différents événements ou sous-événements à traiter. Elle est caractérisée par un intitulé, une

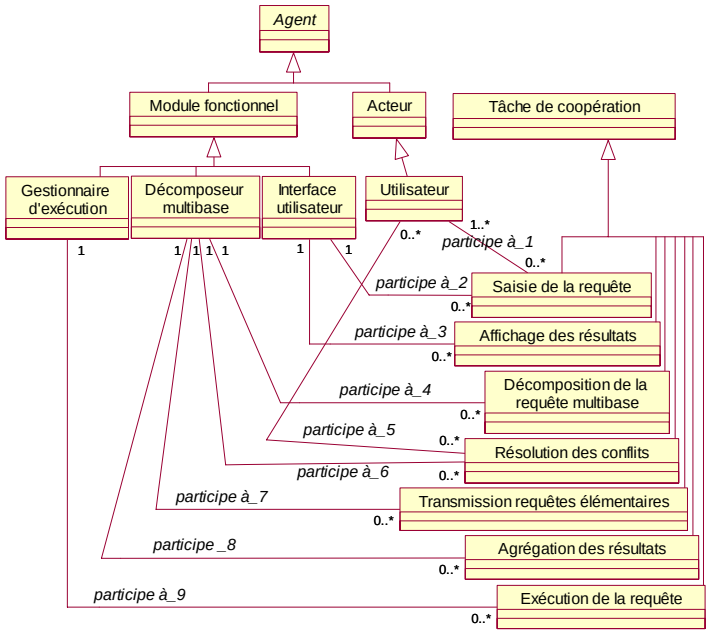
description et éventuellement les flux entrants et sortants (vis-à-vis des sources d'information ou du référentiel de coopération), les pré et post-conditions, ainsi que les pré et post-tâches.
Agent : assure la réalisation ou le contrôle d'une ou plusieurs tâches de coopération. Un agent peut-être composé ou responsable d'un ou plusieurs agents. Un agent peut être un individu (ou acteur) du fait que certaines tâches de coopération peuvent nécessiter des interventions manuelles ou un module fonctionnel réalisant une ou plusieurs tâches de coopération. Il peut avoir un ou plusieurs buts ou rôles dans un processus de coopération.
Acteur : individu (expert, utilisateur,…) intervenant dans une tâche coopérative.
Module fonctionnel : entité (conceptuelle) spécifiée dans le cadre d'un SIC réalisant une tâche de coopération. Un module fonctionnel est caractérisé par un intitulé, une description et éventuellement sa représentation (agent artificiel, objet,…).
Rôle : fonction, responsabilité, autorisation ou contrôle assigné à un agent pour l'exécution d'une ou plusieurs tâches de coopération.

Exemple d'utilisation

Nom de l'exemple

Tâches de coopération effectuées par les agents dans le cadre du SIC fédéré MRDSM (Litwin, 1988).

Modèle



Participants

Utilisateur : Acteur à l'origine de la saisie de la requête globale (ou requête multibase) soumise à la fédération et intervenant lors de la résolution des conflits.

Interface utilisateur : composant permettant la création de la requête globale à l'aide du langage de manipulation multibase dont la description est contenue dans le dictionnaire de la fédération. *Décomposeur multibase :* détermine les bases de données composantes qui doivent être ouvertes, redirige les requêtes n'impliquant qu'une seule base vers les bases correspondantes, décompose la requête multibase en requêtes élémentaires qui seront adressées aux bases locales respectives et résout les conflits de nommage, de type de données et de type de valeurs. *Gestionnaire d'exécution :* requiert et contrôle l'exécution de la requête élémentaire. *Saisie de la requête :* tâche de création de la requête utilisateur soumise à la multibase. *Décomposition de la requête multibase :* décomposition de la requête multibase (i.e., concernant plusieurs bases de données). *Résolution des conflits :* consiste principalement à utiliser et définir manuellement des concepts et techniques (« variables sémantiques », « attributs dynamiques »,…), stockées dans le référentiel de coopération, qui seront ensuite pris en compte par le système lors de la résolution des conflits. *Transmission requêtes élémentaires :* transmission des requêtes élémentaires, obtenues après décomposition de la requête multibase, au gestionnaire d'exécution. *Exécution de la requête :* traitement de la requête sur la base de données locale. *Agrégation des résultats :* consiste à appliquer les clauses « where » et les jointures définies dans les requêtes et à restructurer les résultats. *Affichage des résultats :* tâche permettant l'affichage des résultats « agrégés » à l'utilisateur.

Tableau 1. *Le pattern « Modélisation des tâches de coopération effectuées par les agents ».*

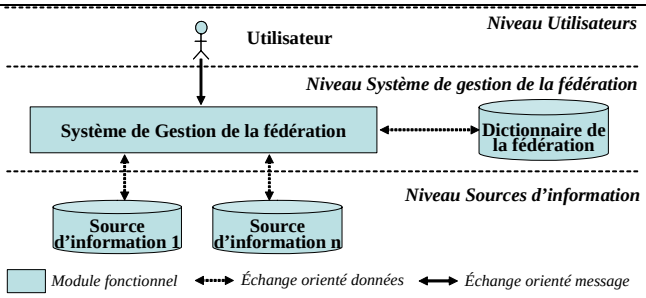
4.1.2. Patterns de coopération dédiés à la phase de conception du processus d'ingénierie des SIC

Les patterns de coopération dédiés à la conception des SIC sont les *patterns d'architecture de coopération* et les *patterns de conception de coopération.*

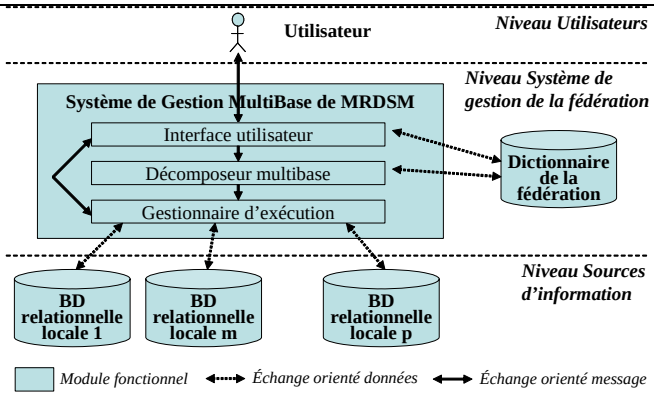
Les *patterns d'architecture de coopération* s'appliquent au début de la phase de conception et fournissent des modèles d'architecture "génériques" facilitant la détermination de l'organisation et de la structure de base du SIC à concevoir. Ils traitent des différents styles d'architecture logicielle possibles d'un SIC, d'un point de vue structurel et organisationnel et aident ainsi à déterminer sa structure de base. Ces modèles d'architecture "génériques" qui seront par la suite adaptés en fonction des entités métier spécifiques (acteurs et référentiels spécifiés,…) au SIC à concevoir - entités mises en exergue lors de la phase d'analyse par application des patterns d'analyse - dépendent de l'approche de coopération choisie (approche fédérée, approche à base de médiation de schémas et approche à base de médiation de contextes). Ainsi, en fonction de l'approche de coopération retenue pour

concevoir le SIC, différents niveaux ou couches seront spécifiés dans le pattern associé et faciliteront la détermination de l'architecture logicielle de ce SIC.

Le *système de patterns d'architecture de coopération* est constitué de trois patterns : le pattern "*Architecture d'un SIC issu de l'approche fédérée*" présente le modèle d'architecture "générique" s'appliquant aux SIC issus de l'approche fédérée (Cf. tableau 2), le pattern "*Architecture d'un SIC issu de l'approche à base de médiation de schémas*" vise à déterminer l'architecture des SIC de type médiation de schémas et le pattern "*Architecture d'un SIC issu de l'approche à base de médiation de contextes*" facilite la définition de l'architecture des SIC issus de l'approche à base de médiation de contextes.

Interface
Nom
Architecture d'un SIC issu de l'approche fédérée.
Classification
Pattern d'architecture de coopération
Objectif
Ce pattern a pour objectif de représenter l'architecture des SIC fédérés.
Solution
Modèle

Participants
Un système d'information coopératif de type fédéré est constitué de trois couches : *Niveau Sources d'information (1er niveau) :* contient les sources d'information participant à la coopération. Ces sources peuvent être de différents types (sources structurées, semi-structurées ou non structurées). *Niveau Système de gestion de la fédération (2ème niveau) :* contient le module fonctionnel en charge de la résolution de la requête globale et du traitement de certains conflits structurels et sémantiques inhérents. Il accède à la base de connaissances appelée dictionnaire de la fédération contenant la description globale de la fédération, les informations concernant l'interface du système fédéré, le ou les protocoles utilisés dans la coopération et les liens inter-bases. *Niveau Utilisateurs (3ème niveau) :* utilisateurs soumettant une requête globale.
Exemple d'utilisation
Nom de l'exemple
Architecture du SIC fédéré MRDSM (Litwin, 1988)

Modèle



Participants

Niveau Sources d'information (1er niveau) : contient les sources d'information locales. Ces sources sont des bases de données relationnelles reposant sur le système de gestion de bases de données MRDS.

Niveau Système de gestion de la fédération (2ème niveau) : contient le module fonctionnel en charge de la résolution de la requête globale et du traitement de certains conflits structurels et sémantiques inhérents. Ce module contient trois sous-modules fonctionnels : l'interface utilisateur, le décomposeur multibase et le gestionnaire d'exécution. L'interface utilisateur permet la création de la requête multibase à l'aide du langage de manipulation multibase dont la description est contenue dans le dictionnaire de la fédération (ou référentiel multibase). Le décomposeur multibase détermine les noms des bases de données composantes qui doivent être ouvertes (grâce au dictionnaire de la fédération), redirige les requêtes n'impliquant qu'une seule base vers les bases correspondantes, décompose la requête multibase en requêtes élémentaires qui seront adressées aux bases locales respectives et résout les conflits de nommage, de type de données et de type de valeur. Le gestionnaire d'exécution requiert et contrôle l'exécution de la requête élémentaire.

Le dictionnaire de la fédération contient notamment la sémantique du langage multibase, i.e. les extensions et enrichissements des langages de manipulation et de définition des données et les liens inter-bases.

Niveau Utilisateurs (3ème niveau) : utilisateurs soumettant une requête à la coopération.

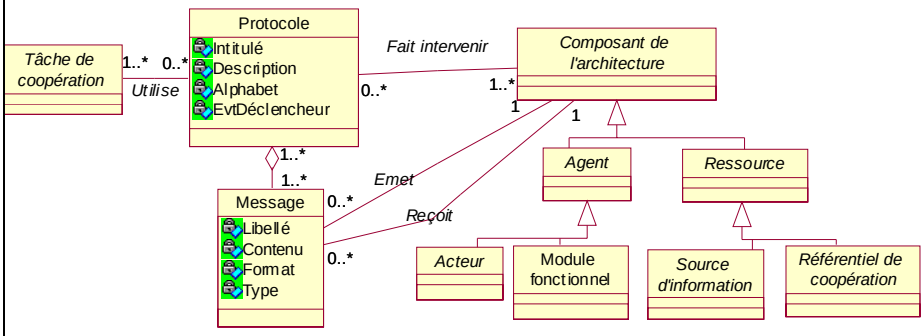
Tableau 2. *Le pattern « Architecture d'un SIC issu de l'approche fédérée ».*

Le passage d'un modèle d'analyse à un modèle de conception nécessite la décomposition et l'enrichissement des entités métier, mises en exergue par les patterns d'analyse de coopération, pour aboutir aux traitements et entités informatiques associés ainsi que la mise en évidence des collaborations entre les entités informatiques du SIC. Les *patterns de conception de coopération* décrivent les éléments conceptuels nécessaires à la mise en œuvre d'un SIC et définissent les traitements informatiques associés. Ils permettent ainsi, plus précisément, de

compléter la description des entités d'analyse identifiées par les patterns d'analyse – notamment en terme de comportement et de collaboration entre entités – et éventuellement de spécifier de nouvelles entités nécessaires à la conception du SIC. Le modèle de conception du SIC est obtenu en adaptant les modèles fournis au sein des solutions de ces patterns.

Certains de nos patterns de conception spécifient des composants réutilisables, dans un langage normalisé, l'*OMG Interface Definition Language (OMG IDL)*, et le savoir-faire associé, en utilisant la structure du pattern comme documentation de ces composants.

Le *système de patterns de conception de coopération* est constitué de six patterns: "*Objet Descriptif des Données*", "*Lien sémantique*", "*Traduction*", "*Création des liens sémantiques*", "*Contract-net protocol*" et "*Interaction d'agents*". Les premiers patterns sont présentés dans (Couturier, 2004a). L'exemple présenté en Tableau 3 est un pattern de conception générique permettant, après adaptation, de modéliser les interactions entre les composants fonctionnels d'un SIC.

Interface
Nom
Interaction d'agents
Classification
Pattern de conception de coopération
Objectif
Ce pattern permet de représenter les protocoles d'interaction faisant intervenir les agents (acteurs et modules fonctionnels) et les ressources (référentiels de coopération et sources d'information) d'un SIC. Il permet également de représenter les messages constituant ce protocole d'interaction.
Solution
Modèle

Ce diagramme de classes permet de représenter la structure statique des interactions entre les agents et ressources du SIC. Un diagramme de séquence permettant de spécifier les interactions entre des instances de classes peut également être utilisé pour décrire l'aspect chronologique de la communication et les scénarii de dialogue

entre les agents et ressources. Ce diagramme de séquence sera donc dépendant du contexte d'utilisation (typologie des agents ou ressources, messages spécifiques,…).

Participants

Tâche de coopération : définit un événement de coopération à traiter. Une tâche de coopération peut-être décomposée en sous-tâches représentant ainsi les différents événements ou sous-événements à traiter. Elle est caractérisée par un intitulé, une description et éventuellement les flux entrants et sortants, les pré et post-conditions, ainsi que les pré et post-tâches.

Agent : participe à la réalisation ou au contrôle d'une ou plusieurs tâches de coopération. Un agent peut être un individu (ou acteur) ou un module fonctionnel (ex. : agent artificiel, objet,…) réalisant une ou plusieurs tâches de coopération.

Ressource : Une ressource qualifie toute entité qui peut être créée, transformée ou utilisée par une tâche de coopération. Une ressource est soit une source d'information coopérante, soit un référentiel de coopération (ontologie, schéma de médiation,…).

Composant : caractérise toute entité de type agent ou ressource du SIC intervenant dans des interactions.

Message : un message est constitué par le contenu des informations transmises. Il est caractérisé par son libellé, son contenu (l'information transmise), son format (message KQML, par exemple), son type (synchrone, asynchrone,…).

Protocole : un protocole est la spécification d'un dialogue impliquant des agents et des ressources. Il permet d'organiser les séquences de messages échangés en de véritables conversations structurées. Un protocole est caractérisé par son intitulé, sa description (règles,…), son alphabet (série de messages ordonnancée du protocole) et l'événement déclencheur (i.e. déclenchant le protocole).

Exemple d'utilisation

Architecture du SIC fédéré MRDSM (Litwin, 1988).

Modèle

Le diagramme de séquence suivant modélise les interactions entre les composants du SIC MRDSM lors du traitement d'une requête multibase (le diagramme de classes résultant de l'adaptation du modèle précédent n'est pas présenté).

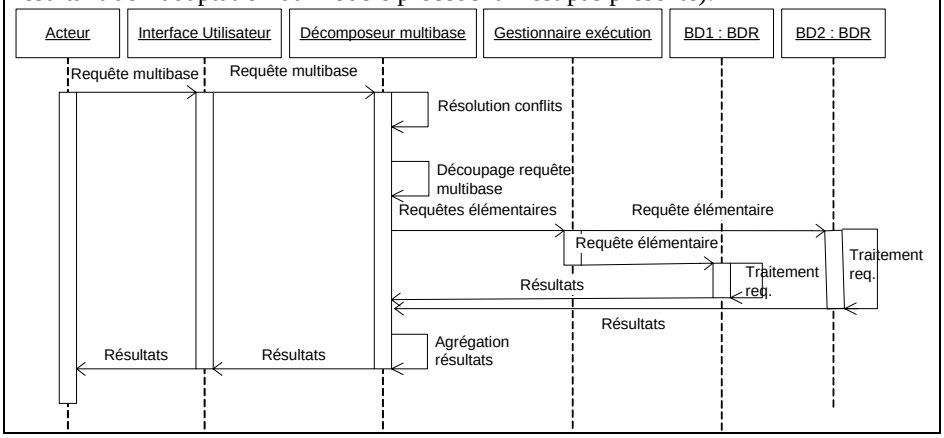


Tableau 3. *Le pattern « Interaction d'agents».*

4..2. Patterns de support de coopération

Les *patterns de support de coopération* proposés sont philosophiquement très proches des patterns de support définis par Fowler (Fowler, 1997). Nos patterns de support de coopération traitent des problèmes qui interviennent quand un système informatique doit être construit à partir des patterns de coopération. Ils décrivent comment utiliser les patterns de coopération et comment les assembler afin d'obtenir l'architecture ciblée. Nous avons spécifié deux types originaux de patterns de support de coopération, les *patterns de support technique* et les *patterns de support d'utilisation*.

4.2.1. Patterns de support technique

Les patterns de support technique sont des patterns qui décrivent comment construire une application à partir des patterns de conception. Ces patterns s'utilisent lors des phases de conception (en fin de phase) et d'implantation des SIC. Ainsi, ce sont soit des patterns dédiés à l'implémentation des entités spécifiées dans les patterns de conception de coopération soit des patterns dédiés à leur implantation.

Les patterns de support technique dédiés à l'implémentation sont, généralement, des patterns de conception spécifiques à un paradigme (objet, agent artificiel,…). Dans le cas, par exemple, d'un SIC spécifiant des agents artificiels en charge de la réalisation des traitements coopératifs, ces patterns vont permettre de construire un système multi-agents, de spécifier les interactions entre les agents (spécification des modèles de coopération ou collaboration),… D'autres patterns, explicitant par exemple les communications (synchrone, asynchrone,…) entre objets, peuvent être utilisés en fonction des paradigmes inhérents au SIC.

Les patterns de support technique dédiés à l'implantation sont spécifiques aux plates-formes d'exécution ou aux langages de programmation. Ces patterns peuvent être des patterns de conception, mais sont la plupart du temps des idiomes (idiomes Java, idiomes CORBA,…). Ils permettent notamment de générer et d'assembler les composants spécifiés dans certains patterns de conception de coopération dans l'environnement technique souhaité afin de construire une micro-architecture (i.e., un ensemble de composants logiciels) destinée à faciliter la création de processus coopératifs entre sources d'information. Ces patterns sont fonction de l'environnement de développement choisi par le développeur d'applications. Nous avons conçu deux patterns de support technique spécifiques assurant la transformation des modèles de conception, intégrés dans les patterns de conception de coopération et indépendants des plates-formes en des modèles dépendants des plates-formes techniques.

Notre démarche est ainsi compatible avec la spécification Model Driven Architecture (MDA) de l'Object Management Group (OMG) basée sur la séparation des modèles de conception (spécifiés dans nos patterns de conception) indépendants

des plates-formes techniques, des modèles d'implantation dépendants des systèmes d'exécution ciblés. Le rôle du concepteur devient alors celui d'un architecte de systèmes complexes ayant à sa disposition des composants standardisés dont l'implémentation est possible dans plusieurs environnements d'exécution, ainsi que des schémas génériques d'organisation et de documentation de ces composants, les patterns.

L'ensemble des patterns de support technique forme un sous-système de patterns, appelé *système de patterns de support technique*. Ce système intègre de nombreux patterns existants (patterns « agents » (Schelfthout *et al.*, 2002), patterns d'ontologie (Reich, 1999),…), ainsi que deux patterns originaux que nous avons conçus (présentés dans (Couturier, 2004a)) : "*Démarche de transformation de modèle*" et "*Traduction des interfaces spécifiées en OMG IDL vers Microsoft IDL*".

4.2.2. Patterns de support d'utilisation

Chaque pattern de coopération et de support technique ne donnant qu'une vision atomique des éléments réutilisables, le concepteur n'aura pas de vision globale de la collection et ne saura pas non plus comment se guider à travers cet ensemble.

Les patterns de support d'utilisation décrivent l'utilisation d'un système de patterns. Un pattern de support d'utilisation décrit la séquence de patterns devant être appliquée pour résoudre un problème. Ce dernier est défini au sein des rubriques liées à la description du problème et du contexte et la solution du pattern présente, sous la forme d'un diagramme d'activités, la séquence de patterns à appliquer pour résoudre ce problème. Ainsi, lors de l'utilisation d'un système de patterns, les utilisateurs traversent les différents chemins de ce diagramme d'activités dont les graphes représentent les patterns à appliquer, en choisissant à chaque fois un chemin spécifique au sous-problème à résoudre. Un pattern de support d'utilisation constitue donc un point d'entrée d'un système de patterns et a pour objectif de guider l'utilisateur dans la réutilisation de celui-ci.

Un ou plusieurs patterns de support d'utilisation sont associés à chaque système de patterns conçu. Un exemple est notamment présenté dans (Couturier, 2005). En complément de ces patterns, nous proposons un environnement de représentation et de support de réutilisation facilitant notamment la compréhension et la sélection des patterns conçus (Couturier, 2004b). Cet outil permet ainsi d'automatiser partiellement le processus d'ingénierie des SIC par réutilisation de patterns.

5. Expérimentation des patterns conçus

Nos patterns dédiés à l'ingénierie des SIC ont été utilisés a posteriori pour modéliser plusieurs systèmes d'information coopératifs issus d'approches de coopération variées. Ces patterns ont ainsi été utilisés pour identifier les entités métier (modèle d'analyse), définir l'architecture ou réaliser le modèle de conception

de plusieurs SIC fédérés (Omnibase, MRDSM), de type médiation de schémas (ACSIS) et de type médiation de contextes (Infosleuth, DILEMMA). Ces différentes applications, notamment présentées dans les exemples d'utilisation, constituent des exemples d'adaptation des modèles "génériques" fournis et nous ont permis de valider l'utilité des patterns proposés, mais aussi partiellement les problèmes identifiés et leurs solutions. Nous sommes actuellement en cours d'expérimentation de nos patterns dans le cadre de l'ingénierie d'un nouveau système d'information coopératif dédié à la coopération de sources d'information et d'applications issus du domaine du transport (projet TRAFIC de la région Rhône-Alpes (TRAFIC, 2005)). Dans le cadre de ce projet, les patterns d'analyse spécifiés sont notamment appliqués pour modéliser les modalités de coopération inter-systèmes de transport.

6. Conclusion

Cet article a présenté nos travaux de recherche axés sur la spécification de patterns dans le but de faciliter l'ingénierie de systèmes d'information coopératifs par réutilisation. Les patterns proposés constituent des briques conceptuelles qui, après adaptation, permettent de construire les modèles d'analyse et de conception d'un nouveau SIC, définir son architecture et favoriser son implantation. La collection présentée propose 22 patterns originaux (Couturier, 2004b) - complétés par plusieurs patterns existants - destinés à toutes les phases du développement d'un SIC : analyse, conception (architecturale et détaillée) et implantation. La plupart des propositions ayant spécifié des patterns dédiés à la coopération de SI ((Fernandez *et al.*, 2000) (Ganguly *et al.*, 2001) (Saidane, 2005)) s'est focalisée sur les seules phases de conception architecturale et/ou détaillée du processus d'ingénierie.

Nos patterns ont été utilisés a posteriori pour représenter plusieurs systèmes d'information coopératifs issus d'approches de coopération variées et sont en cours d'expérimentation dans le cadre de l'ingénierie d'un SIC dédié à la coopération de sources d'information issues du domaine du transport. A terme, nous envisageons d'étendre ces patterns à la prise en compte d'une autre facette des systèmes d'information coopératifs, la facette "travail collaboratif " qui concerne la manière dont des acteurs travaillant sur un processus métier ou un projet commun coopèrent.

7. Références

Alexander C., *The timeless way of building*, New York, Oxford University Press, 1979.

Boulanger D., Dubois G., « Objets et coopération de systèmes d'information », *Ingénierie objet : concepts, techniques et méthodes*, C. Oussalah & al, InterEditions, mai 1997.

Coad P., « Object-oriented patterns », *Com. of the ACM*, vol 35, n°9, 1992, p. 152-159.

Conrad S., Hasselbring W., James A.E., Kambur D., Kutsche R., Thiran Ph., « Report on the EFIS 2001 Workshop », *Computer Journal*, 45(2), 2002, p. 249-251.

Coplien J.O., « Software design patterns : common questions and answers », dans Rising L., *The Patterns Handbook: Techniques, Strategies, and Applications*, New-York, Cambridge University Press, 1998, p. 311-320.

Couturier V., « Patterns de coopération de systèmes d'information », *Actes du 22ème Congrès INFORSID*, Biarritz, 25-28 mai 2004, p. 495-510.

Couturier V., L'ingénierie des systèmes d'information coopératifs par réutilisation : une approche à base de patterns, Thèse de doctorat, Université Jean Moulin, Lyon, déc. 2004.

Couturier V., « Patterns d'analyse pour l'ingénierie des systèmes d'information coopératifs », *L'Objet*, Vol. 11, N°4, Hermès - Lavoisier, novembre 2005, pp. 141-175.

Dogac A. & *al.*, « METU Interoperable Database System », *ACM SIGMOD Record*, vol. 24, n° 3, septembre 1995, p. 552-562.

Dubois G., Boulanger D., Couturier V., « Coopération d'objets pour la résolution de requêtes multibases », *Actes du 20ème Congrès Inforsid*, Nantes, 4-7 juin 2002, p. 199-214.

Fernandez G., Zhao L., « A Pattern Language for Federated Architecture », *Proceedings of KoalaPLoP 2000*, Melbourne, Australie, 24-26 mai 2000, p. 21-31.

Fowler M., *Analysis Patterns*, Addison-Wesley, 1997.

Gamma E., Johnson R., Helm R., Vlissides J., *Design patterns, elements of reusable object-oriented software*, Addison-Wesley, 1995.

Ganguly P., Rabhi F.A., Ray P.K., « The Semantic Interoperability Pattern », *Second Asian-Pacific Conference of Pattern Languages of Program Design*, Nouvelle-Zélande, 2001.

Garcia-Molina H. & *al.*, « The TSIMMIS approach to mediation: Data models and Languages », *Journal of Intelligent Information Systems*, vol. 8, 1997, p. 117-132.

Jouanot F., DILEMMA : vers une coopération de systèmes d'information basée sur la médiation et la fusion d'objets, Thèse de doctorat, Université de Bourgogne, 2001.

Litwin W., « From Database Systems to Multidatabase Systems: Why and How », *Proc. of the British National Conference on Databases*, Cardiff, UK, Juillet 1988, p. 161-188.

Nodine M., Fowler J., Ksiezyk T., Perry B., Taylor M., Unruh A., « Active Information Gathering in InfoSleuth », *IJCIS*, 9:1/2, 2000, p. 3-28.

Port D., Derivation of Domain Specific Design Patterns, USC Center for software engineering, 1998.

Reich J.R., « Ontological Design Patterns for the Integration of Molecular Biological Information », proc. *German Conference on Bioinformatics*, Germany, 1999, p. 156-166.

Saidane M., Formalisation de familles d'architectures logicielles coopératives : démarches, modèles et outils, Thèse de doctorat, Université Joseph Fourier - INPG, Grenoble, 2005.

Schelfthout K. & *al.*, « Agent Implementation Patterns », Proceedings of the *OOPSLA 2002 Workshop on Agent-Oriented Methodologies*, 2002, p. 119-130.

TRAFIC, Site Web du projet TRAFIC (Transports : Réutilisation, Adaptation, Fiabilité, Intermodalité et Coopération inter-systèmes) de la région Rhône-Alpes, http://www-lsr.imag.fr/Les.Personnes/Vianney.Darmaillacq/Site_TRAFIC/, 2005.