Table des matières

Liste des figures v

Liste des tableaux ix

Abstract 1

Résumé 3

1 Introduction générale 5
- 1.1 Contexte général . . . 5
- 1.2 Problématique . . . 6
- 1.3 Questions de recherche . . . 7
- 1.4 Contributions . . . 7
- 1.5 Structure du mémoire . . . 8

I État de l'Art 9

Introduction 11

2 Support IDM à l'alignement de modèles hétérogènes 13
- 2.1 Introduction . . . 13
- 2.2 Critères d'analyse des approches d'alignement de modèles . . . 14
  - 2.2.1 Périmètre de l'étude de la littérature . . . 14
  - 2.2.2 Critères d'analyse des approches existantes . . . 16
- 2.3 Approches d'alignement de modèles hétérogènes . . . 17
  - 2.3.1 VirtualEMF . . . 17
  - 2.3.2 AHM . . . 18
  - 2.3.3 EMF Views . . . 20
  - 2.3.4 OpenFlexo . . . 22
  - 2.3.5 Approche de Vanherpen . . . 24
  - 2.3.6 Approche de Shosha pour l'alignement d'ontologies . . . 25
  - 2.3.7 CIMA pour le matching collaboratif d'ontologies . . . 27
- 2.4 Comparaison des approches . . . 29
- 2.5 Conclusion . . . 31

3 Collaboration pour la prise de décision en groupe 33
- 3.1 Introduction . . . 33
- 3.2 Critères d'analyse des approches de modélisation des processus GDM . . . 34
  - 3.2.1 Périmètre de l'étude de la littérature . . . 34
  - 3.2.2 Critères d'analyse des approches existantes . . . 36
- 3.3 Approches de modélisation des processus GDM . . . 37
  - 3.3.1 DSO . . . 38
  - 3.3.2 MADISE . . . 39

3.3.3 OntoGDSS . . . 41
3.3.4 Approche de Malavolta . . . 43
3.3.5 Collaboro . . . 45
3.4 Comparaison des approches . . . 47
3.5 Conclusion . . . 49

Conclusion **51**

II Contributions Conceptuelles **53**

Introduction **55**

4 Formalisation de la prise de décision en groupe : MMCollab **57**
4.1 Introduction . . . 58
4.2 Principe de construction du méta-modèle MMCollab . . . 58
4.3 Méta-modèle de collaboration (MMCollab) . . . 59
4.3.1 Structuration en paquetages de MMCollab . . . 59
4.3.2 Paquetage Core Concepts . . . 61
4.3.3 Paquetage Actors . . . 64
4.3.4 Paquetage Proposals . . . 66
4.3.5 Paquetage Collective Decision . . . 68
4.3.6 Paquetage Evaluation . . . 73
4.3.7 Syntaxe concrète du méta-modèle MMCollab . . . 75
4.4 Instanciation de MMCollab . . . 75
4.4.1 Exemple de situation de collaboration : Définir des Règles de Gestion (RGs) . . . 76
4.4.2 DecisionPolicy : Instance de CollectiveDecision : :GDMPattern . . . 76
4.4.3 Politique de décision « Taking Advice » . . . 78
4.4.4 Politique de décision « Majority Deciding » . . . 80
4.4.5 Politique de décision « Consenting Together » . . . 83
4.4.6 Politique de décision « Negotiating together » . . . 84
4.4.7 Politique de décision « Delegating » . . . 88
4.5 Conclusion . . . 91

5 Alignement collaboratif de modèles hétérogènes : CAHM **93**
5.1 Introduction . . . 94
5.2 Exemple fil rouge : Conference Management System (CMS) . . . 94
5.2.1 Contexte . . . 94
5.2.2 Méta-modèles des points de vue du CMS . . . 95
5.2.3 Modèles des points de vue du CMS . . . 97
5.3 Présentation globale de l'approche CAHM . . . 99
5.3.1 Vue d'ensemble de CAHM . . . 99
5.3.2 Noyau du Méta-Modèle de Correspondances (MMC) dans CAHM . . . 100
5.3.3 Principe de CAHM - phase 1 . . . 101
5.3.4 Principe de CAHM - phase 2 . . . 102
5.4 CAHM - phase 1 : Mise en correspondance collaborative . . . 104
5.4.1 Vue d'ensemble du processus de mise en correspondance collaborative . . . 104
5.4.2 Activité 2 : Étendre MMC . . . 105
5.4.3 Activité 3 : Définir les méta-correspondances . . . 107
5.4.4 Activité 4 : Générer le modèle de correspondances . . . 110
5.5 CAHM - phase 2 : Maintien de la cohérence lors des évolutions . . . 111
5.5.1 Évolutions supportées . . . 111
5.5.2 MMC : version étendue pour supporter l'évolution . . . 112

5.5.3 Vue d'ensemble du processus de traitement des évolutions de (méta-)modèles 113
5.5.4 Activité 1 : Détecter les changements . . . 115
5.5.5 Activité 2 : Calculer l'impact et l'ordre de traitement des changements . . . 115
5.5.6 Activité 3 : Traiter les incohérences . . . 116
5.6 Conclusion . . . 120

Conclusion **123**

III Implantation et Validation **125**

Introduction **127**

6 Prototype HMCS-Collab **129**
6.1 Introduction . . . 129
6.2 Cas d'utilisation de HMCS-Collab . . . 130
6.3 Modules support à la collaboration . . . 130
6.3.1 Decision Making Tool (DMT) . . . 131
6.3.2 Collaboration Tool (CollabT) . . . 133
6.4 Modules support à l'alignement de modèles . . . 134
6.4.1 Matching Tool (MT) . . . 134
6.4.2 Consistency Management Tool (CMT) . . . 134
6.4.3 Transformation Tool (TT) . . . 135
6.5 Environnement de mise en oeuvre . . . 136
6.5.1 Architecture globale de HMCS-Collab . . . 136
6.5.2 Environnement de développement . . . 137
6.6 Conclusion . . . 141

7 Validation expérimentale **143**
7.1 Introduction . . . 144
7.2 Cas d'étude : Service d'urgence d'un hôpital . . . 144
7.2.1 Vue d'ensemble du cas d'étude . . . 144
7.2.2 Méthodologie de mise en oeuvre du cas d'étude . . . 145
7.2.3 Description des points de vue du cas d'étude . . . 147
7.3 Application de CAHM - phase 1 au système SU . . . 154
7.3.1 Illustration de la problématique de mise en correspondances collaborative . . . 154
7.3.2 Configuration du système SU sur l'outil HMCS-Collab . . . 155
7.3.3 CAHM - phase 1 - Activité 3 : Définir les méta-correspondances . . . 155
7.3.4 CAHM - phase 1 - Activité 4 : Générer le modèle de correspondances . . . 158
7.4 Application de CAHM - phase 2 au système SU . . . 159
7.4.1 CAHM - phase 2 - Activité 1 : Détecter les changements . . . 159
7.4.2 CAHM - phase 2 - Activité 2 : Calculer l'impact et l'ordre de traitement des changements . . . 160
7.4.3 CAHM - phase 2 - Activité 3 : Traiter les incohérences . . . 161
7.5 Retours sur le cas d'étude . . . 164
7.5.1 Synthèse . . . 164
7.5.2 Menaces de validité . . . 164
7.6 Conclusion . . . 165

Conclusion **167**

8 Conclusion et Perspectives **169**
8.1 Rappel des questions de recherche . . . 169
8.2 Bilan des contributions conceptuelles . . . 170
8.3 Positionnement des contributions . . . 171
8.3.1 Retour sur les questions de recherche . . . 171
8.3.2 Positionnement par rapport à la littérature . . . 172
8.4 Limites et perspectives de travail . . . 173
8.4.1 Limites des contributions . . . 173
8.4.2 Perspectives conceptuelles . . . 173
8.4.3 Perspectives techniques et applicatives . . . 174

Annexes **I**

A Méthodes de prise de décision en groupe **III**
A.1 Méthodes non ou peu structurées . . . III
A.2 Méthodes structurées . . . IV

B Compléments sur l'approche CAHM **VII**
B.1 Algorithme du calcul d'impact d'un changement . . . VII
B.2 Construction du graphe de dépendances . . . VIII
B.2.1 Construction du graphe de dépendances d'un changement partiel . . . VIII
B.2.2 Construction du graphe de dépendances d'un changement global . . . VIII
B.3 Catalogue des résolutions d'incohérences . . . X

C Compléments sur l'outil HMCS-Collab **XIII**
C.1 Support des DSLs de CAHM . . . XIII
C.2 Sémantiques des relations . . . XIV
C.2.1 Mindmap des bases de connaissances utilisées et des relations sémantiques définies . . . XIV
C.2.2 Exploitation de WordNet . . . XV
C.2.3 Exploitation de ConceptNet . . . XIX
C.2.4 Expression sémantique de la relation « Similarité » . . . XXI
C.2.5 Expression sémantique de la relation « Agrégation » . . . XXIII
C.2.6 Expression sémantique de la relation « Généralisation » . . . XXIV
C.2.7 Expression sémantique de la relation « Induction » . . . XXV
C.2.8 Expression sémantique de la relation « Déduction » . . . XXVI

Bibliographie **XXIX**

Liste des acronymes **XLIII**

Curriculum Vitae **XLVII**

Liste des publications **XLIX**

Liste des figures

2.1 Feature model de gestion de la cohérence inter-modèles. 14
2.2 Les artefacts de virtualisation de modèles dans VirtualEMF. 18
2.3 Noyau du méta-modèle de correspondances dans AHM. 19
2.4 Schéma global de l'approche EMF Views. 21
2.5 Espace de modélisation dans OpenFlexo. 23
2.6 Modèles linguistiques versus modèles ontologiques. 24
2.7 Processus de mise en correspondance collaborative dans l'approche de SHOSHA et al. [2015]. 26
2.8 Cadre collaboratif de mise en correspondance d'ontologies. 28

3.1 Modèle 3C instancié pour un travail de groupe. 34
3.2 Schéma général d'un processus GDM. 34
3.3 Feature model décrivant les caractéristiques d'un processus GDM. 35
3.4 Le modèle de décision de groupe dans DSO. 38
3.5 Decision Making Ontology (DMO) de l'approche MADISE. 40
3.6 Modèle d'argumentation multicouche dans OntoGDSS. 42
3.7 Méta-modèle générique de prise de décision de MALAVOLTA et al. [2014]. 44
3.8 Éléments clés du méta-modèle Collaboro. 46

4.1 Organisation en paquetages du méta-modèle MMCollab. 59
4.2 Vue d'ensemble du méta-modèle de collaboration (MMCollab). 60
4.3 Paquetage *CoreConcepts* de MMCollab regroupant les concepts de base. 61
4.4 Méta-classe *Collaboration* et ses relations dans MMCollab. 61
4.5 Méta-classe *Proposal* et ses relations dans MMCollab. 62
4.6 Méta-classe *CollaborativeWorkProduct* et ses relations dans MMCollab. 63
4.7 Paquetage *Actors* de MMCollab. 64
4.8 Méta-classe *InvolvedUser* et ses relations dans MMCollab. 65
4.9 Méta-classe *ElementaryProposal* et ses relations dans MMCollab. 67
4.10 Méta-classe *AlternativeProposal* et ses relations dans MMCollab. 68
4.11 Paquetage *CollectiveDecision* de MMCollab. 69
4.12 Méta-classe *GDMPattern* et ses relations dans MMCollab. 70
4.13 Méta-classe *CoDecisionMethod* et ses relations dans MMCollab. 71
4.14 Méta-classe *ParticipationMethod* et ses relations dans MMCollab. 72
4.15 Paquetage *Evaluation* de MMCollab. 73
4.16 Méta-classe *Decision* et ses relations dans MMCollab. 73
4.17 Situation de collaboration « Définir des Règles de Gestion » avant le choix de la DP. . 76
4.18 Structuration des politiques de décision 77
4.19 Diagramme d'activités de la politique « *Taking Advice* ». 79
4.20 Application de la politique *Restricted Taking Advice* à l'exemple Définir RGs. 80
4.21 Diagramme d'activités de la politique « *Majority deciding* ». 81
4.22 Évaluation de la proposition RGa_1 par 3 décideurs selon la politique *Majority Deciding*. 82
4.23 Diagramme d'activités de la politique « *Consenting Together* ». 83

4.24 Application de la politique *Consenting Together* à l'exemple Définir RGs. 85
4.25 Diagramme d'activités de la politique « *Negotiating together* ». 86
4.26 Application de la politique *Negotiating together* à l'exemple Définir RGs. 87
4.27 Diagramme d'activités de la politique « *Delegated Voting* ». 88
4.28 Application de la politique *Delegated Voting* à l'exemple Définir RGs. 90

5.1 Méta-modèle du point de vue conception logicielle (méta-modèle SD). 95
5.2 Méta-modèle du point de vue persistance (méta-modèle PS). 96
5.3 Méta-modèle du point de vue processus métier (méta-modèle BP). 96
5.4 Modèle du point de vue conception logicielle (modèle SD) du Conference Management System (CMS). 97
5.5 Modèle du point de vue persistance (modèle PS) du CMS. 98
5.6 Modèle du point de vue processus métier (modèle BP) du CMS. 99
5.7 Diagramme SADT décrivant le processus global d'alignement de modèles de CAHM. 100
5.8 Noyau du méta-modèle de correspondances dans CAHM. 101
5.9 Principe de mise en correspondance. 102
5.10 Exemple de méta-correspondance et d'une correspondance générée. 102
5.11 Exemple d'évolution de modèle du CMS et ses possibles répercussions sur le modèle de correspondances. 103
5.12 Vue d'ensemble du processus de mise en correspondance collaborative. 105
5.13 Choix de la politique de décision pour l'activité *Extend MMC*. 106
5.14 Déroulement de la collaboration *Extend MMC*. 106
5.15 Choix de la politique de décision pour l'activité *Define Meta-Correspondences*. 107
5.16 Extrait de la collaboration Define Meta-Correspondences. 109
5.17 Propagation de la méta-correspondance MC6A. 110
5.18 Modèle de correspondances M1C du CMS après filtrage. 111
5.19 Vue complète du méta-modèle de correspondances. 113
5.20 Vue d'ensemble du processus de traitement des évolutions de (méta-)modèles. . . . 114
5.21 Diagramme d'activités de Traiter les incohérences. 117
5.22 Graphe de dépendances de *Task : OutsourcePaper*. 119
5.23 Choix de la politique de décision pour l'activité *Choose resolution to apply*. 119
5.24 Déroulement de la collaboration *Choose resolution to apply*. 120

6.1 Diagramme des cas d'utilisation de HMCS-Collab. 130
6.2 Modules de HMCS-Collab dédiés au support de la collaboration. 131
6.3 Enchaînement fonctionnel de DMT. 131
6.4 Utilisation du patron *Observer* pour notifier les décideurs d'une proposition. 132
6.5 Enchaînement fonctionnel de CollabT. 133
6.6 Déroulement général du processus de filtrage des correspondances. 135
6.7 Architecture globale de HMCS-Collab. 136
6.8 Diagramme des composants de HMCS-Collab. 136
6.9 Projets Eclipse du premier niveau. 137
6.10 Architecture de EMF. 138
6.11 Fonctionnement de EMFCollab. 139
6.12 Fonctionnement de CDO. 140
6.13 Quelques plug-ins natifs de ECF. 140

7.1 Quelques points de vue métier sur le service d'urgence. 144
7.2 Processus informel de validation de l'approche par cas d'étude. 145
7.3 Méta-modèle du point de vue conception logicielle. 147
7.4 Modèle du point de vue conception logicielle. 148
7.5 Méta-modèle du point de vue processus métier. 149
7.6 Modèle du point de vue processus métier. 150

7.7 Méta-modèle du point de vue des rapports médicaux. 150
7.8 Modèle du point de vue rapports médicaux. 151
7.9 Méta-modèle du point de vue système multi-agents. 152
7.10 Modèle du point de vue système multi-agents. 153
7.11 Exemple illustrant la problématique de mise en correspondances sur le cas d'étude « service d'urgence ». 154
7.12 Interface de configuration du système SU dans HMCS-Collab. 155
7.13 Interface de configuration de la politique de décision de l'activité collaborative « Définir les méta-correspondances » dans HMCS-Collab. 156
7.14 Interface de proposition de méta-correspondance dans HMCS-Collab. 156
7.15 Interface de la liste des méta-correspondances proposées. 157
7.16 Interface d'évaluation des méta-correspondances par Alice. 157
7.17 Interface de la proposition alternative initiée par Alice. 158
7.18 Interface des décisions collectives des méta-correspondances proposées dans HMCS-Collab. 158
7.19 Extrait du modèle de correspondances M1C du service d'urgence visualisé sur HMCS-Collab. 159
7.20 Spécialisation du *Field :physio* lors de l'évolution du modèle ER. 160
7.21 Interface de classement des impacts des changements détectés. 160
7.22 Interface de visualisation du graphe de dépendances de *Task :Control*. 161
7.23 Liste des méta-correspondances après l'ajout du point de vue MAS, et avant leur évaluation. 161
7.24 Choix de la résolution à appliquer pour le changement *rename Task :Control*. 163

8.1 Schémas récapitulatifs de positionnement de CAHM par rapport à l'état de l'art. . . 172

B.1 (a) Exemple d'un modèle de correspondances et (b) du graphe de dépendances associé à l'élément « c ». IX
B.2 Graphe de dépendances associé à la suppression du modèle PS du CMS. X

C.1 Utilisation de l'éditeur de modèle Ecore pour la conception du méta-modèle MMC et l'expression des contraintes OCL. XIII
C.2 Utilisation de l'éditeur de modèle Ecore pour la conception de MMCollab et l'expression des contraintes OCL dans l'éditeur OCLInEcore. XIV
C.3 Mindmap des relations sémantiques et des fonctions qui les mettent en oeuvre. . . XV

Rapport-Gratuit.com

Liste des tableaux

2.1 Synthèse des caractéristiques supportées par les approches d'alignement de modèles hétérogènes. 30

3.1 Synthèse des caractéristiques supportées par les approches de modélisation de GDM. 48

4.1 Représentations graphiques des concepts de MMCollab. 75
4.2 Synthèse des caractéristiques supportées par les approches de modélisation de GDM incluant MMCollab. 92

5.1 Méta-correspondances proposées pour le CMS 108
5.2 Décisions collectives pour les méta-correspondances proposées pour le CMS 109
5.3 Exemples d'évolution des modèles du CMS 115
5.4 Extrait du catalogue de résolutions d'un changement global 118
5.5 Extrait du catalogue de résolutions d'un changement partiel 118
5.6 Synthèse des caractéristiques supportées par les approches d'alignement de modèles hétérogènes incluant l'approche CAHM. 121

7.1 Équipes impliquées dans l'élaboration du cas d'étude SU 146

8.1 Positionnement des contributions par rapport aux questions de recherche 172

B.1 Catalogue de résolutions des incohérences XI

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

Abstract

The design of complex systems goes through a multi-view paradigm in which separate teams, from different business viewpoints, build partial source models describing the system. As they are expressed in different languages, these partial models are called heterogeneous models. The main objective of this PhD thesis is to provide an approach that leverages collaborative engineering to ensure the overall consistency of heterogeneous source models. We first study the mechanisms offered by the existing approaches of model alignment. Then, we analyze the literature on modeling group decision-making. Based on this analysis, we have developed the meta-model MMCollab, a set of pattern-based group decision-making strategies and the two subprocesses of our approach called CAHM (Collaborative Alignment of Heterogeneous Models).
The first subprocess targets the collaborative matching of source models; it formalizes the collaborative development of inter-model correspondences by offering a semi-automatic mechanism that reduces the involvement of business actors while ensuring the relevance and quality of the established model of correspondences.
The second subprocess deals with the evolution of source models and meta-models, and mainly with the impact of these evolutions on the established model of correspondences. Consequently, this subprocess aims at maintaining the coherence of the overall system; it incorporates mechanisms to detect and calculate the impact of changes, as well as mechanisms and recommendations to collaboratively address inconsistencies that may occur.
CAHM is based on two meta-models : A meta-model describing collaboration and group decision-making in particular. This meta-model, called MMCollab, describes the fundamental concepts of a group decision-making and unfolds the phases of the constitution of collective decisions. The second meta-model is the MetaModel of Correspondences (MMC). This meta-model allows to define the structure of the model of correspondences, its correspondences and the relationships' types that characterize those latter.
The approach provides a support tool, called HMCS-Collab. Il allows unrolling CAHM's subprocesses by integrating both alignment and collaboration aspects. It is based on the Eclipse platform and provides a web application allowing a lightweight execution of the approach by distant actors. The approach has been illustrated and validated on a real example of a hospital emergency department case study.

Keywords : *Multi-View Design, Model-Based Engineering, Model Alignment, Heterogeneity, Model of correspondences (correspondence model), Collaboration, Group Decision Making, Pattern, Meta-model, Matching, Evolution, inter-model consistency.*

Résumé

L'objectif principal du travail présenté dans cette thèse est de fournir une approche tirant profit de l'ingénierie collaborative pour aligner des modèles source représentant un système complexe lors d'une conception multi-vue. Ce travail part des limitations des approches d'alignement de modèles de la littérature. Nous étudions dans un premier temps les techniques offertes, par ces approches, pour fournir la fonctionnalité de cohérence inter-modèles. Ensuite, nous analysons les travaux de la littérature concernant la modélisation de la prise de décision en groupe. A l'issue de cet état de l'art réalisé en deux temps, nous avons élaboré le méta-modèle MMCollab et l'approche CAHM (pour Collaborative Alignment of Heterogeneous Models).
CAHM fournit un processus collaboratif global composé de deux sous-processus. Le premier sous-processus concerne la mise en correspondance collaborative des modèles source. Il formalise les collaborations lors de l'élaboration des correspondances inter-modèles en offrant un mécanisme semi-automatique qui minimise l'implication des acteurs métier tout en garantissant la pertinence et la qualité du modèle de correspondances établi.
Le deuxième sous-processus traite l'évolution des modèles et méta-modèles source, et en particulier l'impact de cette évolution sur le modèle de correspondances établi, et par conséquent sur la cohérence globale du système. Il intègre des mécanismes pour détecter et calculer l'impact des évolutions, et aussi des recommandations pour réaliser le traitement collaboratif des incohérences engendrées par les changements.
CAHM s'appuie sur deux méta-modèles. Le premier, MMCollab, permet de décrire les concepts fondamentaux d'une prise de décision en groupe, notamment des patrons de prise de décision fixant les caractéristiques des méthodes d'élaboration des décisions collectives. Le deuxième méta-modèle est le Méta-Modèle de Correspondances (MMC) qui permet de définir la structure du modèle de correspondances, des correspondances et des relations typées qui les caractérisent.
Nous avons développé un outil support, HMCS-Collab qui permet de mettre en oeuvre le processus global de CAHM en intégrant à la fois les aspects alignement et collaboration. Il est basé sur la plateforme Eclipse et fournit une application web permettant la mise en oeuvre de l'approche par des acteurs géographiquement distribués. L'approche a été appliquée à une étude de cas tirée du fonctionnement d'un service d'urgence d'un hôpital français.

Mots-clés : *Conception multi-vue, Ingénierie à base de modèles, Alignement de modèles hétérogènes, Modèle de correspondances, Collaboration, Prise de décision en groupe, Patron de prise de décision, Évolution de modèles, Gestion de la cohérence inter-modèles.*

Chapitre 1

Introduction générale

> Take the first step in faith. You don't have to see the whole staircase, just take the first step.
>
> *Martin Luther King Jr.*

Sommaire

1.1 Contexte général 5
1.2 Problématique 6
1.3 Questions de recherche 7
1.4 Contributions 7
1.5 Structure du mémoire 8

1.1 Contexte général

Les systèmes complexes, tels que les systèmes cyber-physiques, sont généralement modélisés par une approche incluant divers points de vue métier [Le Moigne, 1999; Diskin et al., 2019], impliquant par conséquent plusieurs acteurs [Di Ruscio et al., 2017]. Cette conception multi-vue [Ciccozzi et al., 2018a] assure le principe de séparation des préoccupations [Dijkstra, 1976], en décomposant la modélisation du système en plusieurs unités maîtrisables par des experts de domaines métier distincts [Brichau et D'Hondt, 2005]. Cela permet de résoudre les tâches d'ingénierie de manière plus efficace en réduisant la complexité et en garantissant plus d'expressivité [Atkinson et Kühne, 2008; Neumayr et al., 2011]. La séparation des préoccupations dans un cadre d'Ingénierie Dirigée par les Modèles (IDM) [Schmidt, 2006] passe par l'utilisation de plusieurs modèles, appelés modèles *source* [Yie et al., 2009; Kramer et al., 2013]. Ces modèles sont généralement hétérogènes, c'est-à-dire décrits dans des langages différents [France et Rumpe, 2007; Broy et al., 2010]. A titre d'exemple, la conception d'un véhicule fait intervenir des concepteurs (et respectivement des modèles) des points de vues mécanique, électrique, conception logicielle, sécurité, etc. La cohabitation des modèles de ces points de vues soulève un certain nombre de problèmes, principalement liés à la gestion de la cohérence du système dans son ensemble [König et Diskin, 2016]. En effet, les modèles source décrivent parfois les mêmes éléments de façons différentes ou bien des éléments interdépendants [Reineke et Tripakis, 2014].
Les travaux de notre équipe dans le cadre de la conception multi-vue ont commencé par supposer que les modèles source étaient décrits dans un langage généraliste tel qu'UML [Nassar, 2003; Anwar et al., 2010]. Mais cette hypothèse s'étant avérée peu réaliste pour modéliser un système complexe, elle a été relâchée en considérant les modèles source comme hétérogènes, c'est-à-dire décrits dans des langages dédiés distincts appelés Domain Specific Language (DSL). Chaque DSL est décrit par un méta-modèle à part. Un méta-modèle et les modèles qui lui sont conformes

représentent un point de vue métier. Cette hypothèse d'hétérogénéité correspond tout à fait à la réalité de la conception des systèmes complexes, par exemple dans des domaines tels que la médecine, l'aéronautique, les systèmes d'information, etc.
Les travaux de notre équipe dans le cadre de la thèse de EL HAMLAOUI [2015] ont abouti à la proposition d'une approche, nommée Alignment of Heterogeneous Models (AHM), qui permet de mettre en correspondance les éléments des différents modèles source, puis de maintenir la cohérence de l'ensemble en cas de modification de ces modèles. Cependant, l'approche AHM est une approche centralisée qui dépend fortement d'un expert maîtrisant le domaine global du système. Or, en réalité, un seul acteur ne peut avoir une connaissance intégrale et profonde de tous les modèles source et la conception efficiente pour ce type de systèmes est une conception collaborative [FRANZAGO et al., 2017]. En effet, les modèles source sont souvent élaborés par des acteurs/équipes répartis sur plusieurs sites et travaillant sur des artefacts partagés, voire dans des environnements hétérogènes. Aussi, comme proposé par [BUCCHIARONE et al., 2020], l'IDM devrait être le catalyseur permettant aux acteurs métier de construire les outils dont ils ont besoin dans leurs domaines ; ils peuvent être accompagnés par les experts IDM au début pour mettre en place les bases de l'IDM, mais la définition des liens inter-modèles selon les spécificités du domaine d'application est leur responsabilité : *« Modeling by the People, for the People »* [KELLY, 2018].
Cette particularité de devoir impliquer les acteurs métier dans la définition des liens entre modèles soulève la question de la manière dont ils peuvent assurer la cohérence inter-modèles quand les modèles source sont modifiés (hétérogénéité des points de vue métier, traçage des changements, gestion de versions, etc.).

1.2 Problématique

Le travail décrit dans ce manuscrit considère l'approche AHM [EL HAMLAOUI et al., 2014; EL HAMLAOUI, 2015; EL HAMLAOUI et al., 2018] comme un point de départ. L'enjeu est de pouvoir intégrer les points de vue métier du système en optant pour une solution collaborative qui supporte la participation des acteurs des différents domaines métier afin de produire une vision globale reflétant les interdépendances et les liens existant entre leurs modèles.
L'élaboration de cette vision globale passe par une coordination et prise de décision en groupe entre des équipes qui peuvent être (géographiquement) distribuées, avec des préoccupations métier et connaissances variées. La vision globale à produire peut être exploitée dans plusieurs finalités : *synchronisation* [GIESE et al., 2010; DISKIN et al., 2019], *traçabilité* [NASLAVSKY et al., 2005; GALVAO et GOKNIL, 2007], *gestion de la cohérence globale* [TROLLMANN et al., 2011; KRAMER, 2015], etc.
Nous nous intéressons principalement à la question de la cohérence inter-modèles : l'établissement de la cohérence du système global lors de la conception dans un premier temps et ensuite le maintien de cette cohérence en cas de changements (changements de règles métier et de contraintes techniques par exemple).
Dans cette thèse, nous partons de l'hypothèse qu'un système complexe est représenté par un ensemble de modèles source de domaines métier distincts, gérés par des concepteurs (ou équipes de concepteurs) différent-e-s. D'une part, chaque modèle source, considéré séparément des autres, possède des fonctionnalités propres, ce qui justifie que ces modèles existent indépendamment les uns des autres. D'autre part, nous considérons qu'ils sont conçus selon l'IDM ; de ce fait, chaque modèle est conforme à un méta-modèle auquel est associé un DSL.
Notre problématique principale est donc : *« l'assurance de la cohérence entre les modèles source représentant un système et ce, en préservant les préoccupations métier exprimées séparément par les différents acteurs impliqués »*.

1.3 Questions de recherche

A partir de la problématique formulée ci-dessus, la question de recherche globale à laquelle répond cette thèse est :

Comment exprimer et assurer la cohérence entre les modèles hétérogènes d'un système dans le cadre d'une conception collaborative évolutive impliquant plusieurs acteurs métier ?

De cette question découle deux questions dérivées :

QR 1. Comment capturer les liens entre modèles hétérogènes (i.e., avoir une vision globale) lors d'une conception multi-vue, en impliquant les acteurs métier ? Question se décomposant elle-même en deux sous-questions :

- QR1.1. Où intervient la collaboration entre les différents acteurs métier lors de l'élaboration des liens inter-modèles ?
- QR1.2. Quelle assistance fournir aux acteurs pour définir les liens entre les modèles ?

QR 2. Comment maintenir la vision globale d'un système lors des évolutions des (méta-)modèles de points de vue métier impliqués ? Cette question se décompose elle-même en deux sous-questions :

- QR2.1. Où intervient la collaboration entre les différents acteurs métier lors du maintien de la cohérence de l'ensemble ?
- QR2.2. Quelle assistance fournir aux acteurs pour maintenir la cohérence de l'ensemble des modèles ?

1.4 Contributions

Pour répondre à la question de recherche globale de cette thèse, nous avons exploité et adapté les acquis de l'ingénierie collaborative pour élaborer une vue globale et cohérente entre les modèles hétérogènes décrivant un système donné.
Dans ce mémoire, nous décrivons les quatre contributions qui ont permis de répondre à cet objectif :

Contribution 1. La première contribution est un état de l'art consistant à délimiter le contexte du travail et analyser les travaux de la littérature en termes d'alignement de modèles et de formalisation de la prise de décision collaborative. Ceci a donné lieu à une synthèse en deux temps des travaux de la littérature.

Contribution 2. La deuxième contribution de ce travail est la définition d'un méta-modèle, nommé Méta-Modèle de Collaboration (MMCollab), structurant les prises de décision collaboratives. Nous avons élaboré aussi une liste extensible de *politiques de décision*, qui sont des instances du concept *GDMPattern* de MMCollab.

Contribution 3. L'objectif principal de la thèse étant d'élaborer et de maintenir la cohérence entre les modèles représentant un système complexe, la troisième contribution de ce travail est la définition d'une approche d'alignement collaboratif de modèles hétérogènes (Collaborative Alignment of Heterogeneous Models (CAHM)). Cette approche est axée autour d'un processus collaboratif composé de deux sous-processus : le premier pour la mise en correspondance de modèles hétérogènes et le deuxième pour le maintien de la cohérence inter-modèles. CAHM exploite MMCollab pour formaliser les prises de décision en groupe durant le processus d'alignement.

Contribution 4. La quatrième contribution est le développement d'un outil support permettant de valider l'approche. Cet outil, nommé HMCS-Collab, supporte le processus de CAHM ; il permet d'appliquer l'approche dans des domaines spécifiques en limitant l'intervention humaine aux tâches de prise de décision collective et de coordination.

1.5 Structure du mémoire

Le mémoire est organisé en trois parties.
La **première partie** est un **état de l'art** sur l'alignement de modèles hétérogènes et la prise de décision en groupe. Cette partie est organisée en deux chapitres :

Le chapitre 2 (Support IDM à l'alignement de modèles hétérogènes) étudie les approches d'alignement de modèles hétérogènes selon un ensemble de critères que nous avons jugé pertinents, puis élabore une synthèse de leurs fonctionnalités et limitations par rapport à notre problématique.

Le chapitre 3 (Collaboration pour la prise de décision en groupe) étudie les approches modélisant la prise de décision en groupe selon un ensemble de critères que nous avons identifiés, puis élabore une synthèse de leurs fonctionnalités et limitations par rapport à notre besoin.

La **seconde partie** de ce mémoire présente **notre contribution conceptuelle** répondant aux questions de recherche précédemment formulées. Cette partie est structurée en deux chapitres :

Le chapitre 4 (Formalisation de la prise de décision en groupe : MMCollab) détaille le méta-modèle MMCollab décrivant la prise de décision collaborative et expose les politiques de décision élaborées en instanciant les concepts de ce méta-modèle.

Le chapitre 5 (Alignement collaboratif de modèles hétérogènes : CAHM) décrit l'approche CAHM. Cette approche part des principales limitations des approches évoquées dans le chapitre 2 et s'appuie sur le méta-modèle MMCollab pour gérer les prises de décision en groupe durant le processus d'alignement.

La **troisième partie** de ce mémoire présente **l'implantation de l'approche et sa validation** sur un cas d'étude. Cette partie est divisée en deux chapitres :

Le chapitre 6 (Prototype HMCS-Collab) présente le prototype implémentant les contributions conceptuelles de cette thèse.

Le chapitre 7 (Validation expérimentale) illustre l'application de l'approche au cas d'étude *service d'urgence d'un hôpital*, en utilisant l'outil HMCS-Collab.

Le chapitre 8 (Conclusion et Perspectives) apporte une conclusion au document en reprenant les questions de recherche initiales, résume les contributions réalisées, aborde les limites du travail effectué et introduit des perspectives d'approfondissement.

Première partie

État de l'Art

Introduction 11

2 Support IDM à l'alignement de modèles hétérogènes 13

3 Collaboration pour la prise de décision en groupe 33

Conclusion 51

Introduction

Dans cette première partie, nous faisons une revue des travaux de recherche ciblant **l'alignement de modèles hétérogènes** et **la prise de décision en groupe** applicable pour réaliser cet alignement.

Ces deux thématiques, dont la combinaison constitue les fondements de notre travail, sont néanmoins séparées dans la littérature. Elles justifient ainsi un état de l'art composé de deux chapitres distincts.

Dans le premier chapitre de cette partie (**chapitre 2**) intitulé **Support IDM à l'alignement de modèles hétérogènes**, l'étude est focalisée sur les approches d'alignement de modèles hétérogènes. *L'alignement*, terme emprunté au domaine des schémas et ontologies [EHRIG, 2006; EUZENAT et al., 2011] consiste à (i) établir les liens inter-modèles, appelés aussi *correspondances* [BÉZIVIN et al., 2006; SELONEN et KETTUNEN, 2007] et (ii) re-évaluer ces correspondances en cas d'évolution des modèles. Nous présentons des approches d'alignement de modèles hétérogènes selon un ensemble de critères de comparaison.

Le second chapitre de cette partie (**chapitre 3**) est intitulé **Collaboration pour la prise de décision en groupe**. La prise de décision en groupe étant un cas particulier de collaboration, elle inclut, en plus des 3C de la collaboration : *Communication*, *Coordination* et *Coopération*, [ELLIS et al., 1991], un volet *prise de décision*. Nous focalisons notre revue de la littérature sur les approches de modélisation de prise de décision et les analysons selon un ensemble de critères que nous avons jugés pertinents.

Chapitre 2

Support IDM à l'alignement de modèles hétérogènes

Sommaire

2.1 Introduction 13
2.2 Critères d'analyse des approches d'alignement de modèles 14
2.2.1 Périmètre de l'étude de la littérature 14
2.2.2 Critères d'analyse des approches existantes 16
2.3 Approches d'alignement de modèles hétérogènes 17
2.3.1 VirtualEMF 17
2.3.2 AHM 18
2.3.3 EMF Views 20
2.3.4 OpenFlexo 22
2.3.5 Approche de Vanherpen 24
2.3.6 Approche de Shosha pour l'alignement d'ontologies 25
2.3.7 CIMA pour le matching collaboratif d'ontologies 27
2.4 Comparaison des approches 29
2.5 Conclusion 31

2.1 Introduction

La séparation des préoccupations lors de la conception multi-vue implique l'élaboration de différents modèles source hétérogènes spécifiques aux points de vue métier [Bruneliere et al., 2019]. Certes, elle a pour avantage d'apporter une modularité lors de la conception, mais a en revanche comme inconvénient de nécessiter un moyen d'établir une vue globale du système étudié.
Nous nous plaçons dans le cadre de l'Ingénierie Dirigée par les Modèles (IDM) pour l'élaboration de la vue globale d'un système. En effet, le domaine de l'IDM est un domaine mature (une vingtaine d'années d'existence) avec des contributions importantes pour tirer parti de l'abstraction et de l'automatisation dans les domaines de la conception et du développement de logiciels et systèmes [Mussbacher et al., 2014; Kessi et al., 2014; Hassan et Oussalah, 2018].
Diverses techniques à base d'IDM ont été proposées pour l'élaboration d'une vue globale sur un système : *fusion* [Bézivin et Kurtev, 2005; Vallecillo, 2010], *tissage* [Del Fabro et Valduriez, 2009; Jouault et al., 2010] et *fédération* [Clasen et al., 2011; Galster, 2011; Golra et al., 2016], etc. Ces techniques s'appuient sur l'IDM et offrent ainsi l'avantage de réduire l'effort requis puisqu'elles se basent sur les principes de méta-modélisation.
Nous focalisons notre étude sur les techniques (fédération, tissage, etc.) qui permettent de garder les modèles dans leur propre langage d'expression en élaborant des liens (correspondances) entre leurs éléments. Dans la suite, nous désignons ces techniques par le terme d'*alignement* par souci d'homogénéisation de nomenclature.

La restriction de l'état de l'art à cette catégorie de techniques est justifiée par plusieurs raisons : (i) les modèles source existent séparément et ont une signification et un intérêt métier, (ii) en cas d'utilisation d'une technique intrusive qui modifie les modèles source, si un nouveau point de vue métier est pris en compte (i.e., un modèle et son méta-modèle), l'élaboration de la vue globale doit repartir de zéro.
Dans ce chapitre, nous faisons une revue des approches traitant de l'alignement de modèles hétérogènes. Dans la section 2.2, nous précisons le périmètre de notre étude de l'existant et présentons par la suite les critères retenus pour notre analyse. Ensuite, nous détaillons dans la section 2.3 les particularités des approches étudiées relativement aux critères recensés. La section 2.4 présente une synthèse de la comparaison de ces approches tandis que la section 2.5 clôture ce chapitre.

2.2 Critères d'analyse des approches d'alignement de modèles

2.2.1 Périmètre de l'étude de la littérature

L'alignement de modèles hétérogènes peut être réalisé avec divers objectifs : *synchronisation* [GIESE et al., 2010; DISKIN et al., 2019], *traçabilité* [NASLAVSKY et al., 2005; GALVAO et GOKNIL, 2007], *gestion de la cohérence globale* [TROLLMANN et al., 2011; KRAMER, 2015], *mapping des modèles* [GUYCHARD et al., 2013], etc. Nous nous plaçons dans le cadre de la gestion de la cohérence lors de la conception multi-vue à base de modèles.
La Figure 2.1 présente un *feature model*[1] regroupant les concepts clés de gestion de la cohérence entre modèles. Nous nous sommes inspirés du *feature model* de CICCHETTI et al. [2019] pour la description de la cohérence inter-modèles, en lui ajoutant des éléments tirés du *feature model* de PFEIFFER et WĄSOWSKI [2015], qui précisent les caractéristiques des relations entre modèles.
Notons qu'un autre *feature model* a été proposé dans la littérature, il s'agit de celui de BRUNELIERE et al. [2019] ; nous ne l'avons pas retenu étant donné qu'il est orthogonal à celui de CICCHETTI et al. [2019] et caractérise principalement les types des modèles de points de vue, leurs langages de requête, et les fonctionnalités de gestion de la cohérence lors de l'exécution des modèles.

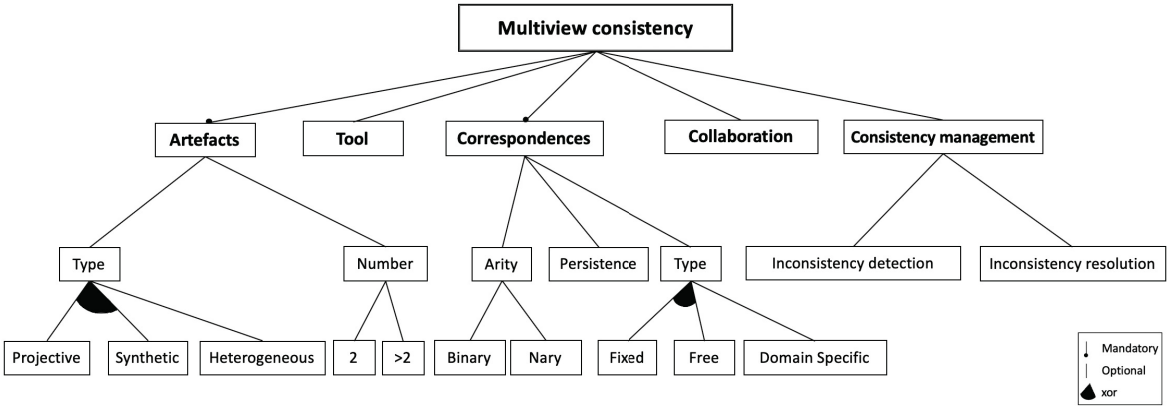


FIGURE 2.1 – Feature model de gestion de la cohérence inter-modèles.

La Figure 2.1 précise en particulier les cinq caractéristiques de la cohérence multi-vue, à savoir : **artefacts** (artefacts), **correspondances** (correspondences), **gestion de la cohérence** (consistency management), **outil support** (tool support) et **collaboration** (collaboration).

Artefacts. Les approches diffèrent selon le (i) *nombre*, le (ii) *type* et la (iii) *nature* des artefacts en entrée (modèles et méta-modèles).

1. Un feature model est un moyen d'exprimer les exigences d'un domaine à un niveau abstrait. Il permet de décrire les propriétés variables et communes des produits d'une ligne de produits, ainsi que de dériver et valider les configurations des systèmes logiciels [RIEBISCH, 2003].

Concernant le *nombre* d'artefacts, certaines approches peuvent prendre en entrée uniquement deux artefacts tandis que d'autres n'imposent pas de restriction sur ce nombre.

En ce qui concerne le *type* des artefacts, les approches peuvent être classifiées en approches *synthétiques* versus *projectives* [Cicchetti et al., 2019]. Dans les approches *synthétiques*, chaque point de vue métier est mis en oeuvre sous forme de méta-modèle distinct et le système global est obtenu par synthèse des informations véhiculées par les différents points de vue. Alors que pour les approches *projectives*, des vues virtuelles issues d'un méta-modèle de base sont construites en masquant les détails non pertinents pour le point de vue pris en compte.

Concernant la *nature* des modèles en entrée, certaines approches se limitent à des modèles homogènes tandis que d'autres supportent l'*hétérogénéité* des modèles, i.e., des modèles conformes à différents méta-modèles.

Correspondances. Une correspondance désigne une relation et un ensemble de n éléments (n>=2) reliés par cette relation.

Les caractéristiques des approches fondées sur les correspondances diffèrent selon (i) *l'arité* des correspondances (binaire versus n-aire), (ii) *le type des relations* supportées (ensemble figé de relations, relations spécifiques à un domaine, relations indépendantes du domaine (qualifiées de libre) Pfeiffer et Wąsowski [2015]) et (iii) la possibilité de rendre les correspondances *persistantes* (via un modèle de correspondances par exemple).

Gestion de la cohérence. L'existence de correspondances ne signifie pas que l'approche concernée est capable de garantir et maintenir la cohérence globale entre les modèles. Par conséquent, les approches diffèrent selon qu'elles incluent ou non des mécanismes permettant de (i) *détecter* puis (ii) *traiter* les *incohérences*.

Nous distinguons deux types d'incohérences possibles [Feldmann et al., 2019] : *incohérences intra-modèles* et *incohérences inter-modèles*. Les incohérences intra-modèles se produisent au sein d'un même modèle. Les incohérences inter-modèles existent entre un ensemble de modèles différents ; elles résultent principalement de défauts de cohérence ou de complétude entre des modèles distincts [Lange et Chaudron, 2004], par exemple des incohérences dues à des liens incorrects ou manquants entre les éléments de différents modèles.

Dans cette thèse, nous nous intéressons à **la cohérence inter-modèles** et considérons par conséquent que la responsabilité d'assurer la cohérence intra-modèle incombe aux concepteurs de chaque point de vue métier.

Collaboration. La collaboration est un caractère intrinsèque du génie logiciel piloté par les modèles. Les travaux se focalisant sur cet aspect ont considérablement augmenté ces dernières années [Franzago et al., 2017]. En effet, les approches ne pouvant pas être complètement automatisables, un effort humain est requis lors du déroulement de l'approche. Dans un cadre de conception par points de vue hétérogènes, cet effort ne peut pas venir d'un seul acteur (vu la difficulté de maîtriser tous les points de vue métier), mais plutôt d'un ensemble d'acteurs, ayant des connaissances diversifiées et complémentaires. Ainsi les approches diffèrent selon la manière dont elles supportent, le cas échéant, l'aspect collaboratif lors de la mise en cohérence des modèles.

Outil support. Les approches diffèrent selon l'outillage associé. L'outil support peut proposer des fonctionnalités pour certaines des quatre caractéristiques précédentes.

Étant donné le nombre abondant des approches traitant de cette thématique et pour limiter notre analyse de la littérature, nous considérons uniquement les approches d'alignement satisfaisant les points suivants (i.e., restriction du périmètre de la caractéristique Artefacts) :

— Elles traitent des modèles **hétérogènes** ;

— Elles peuvent traiter **plus de deux modèles** en entrée ;

— Elles sont **synthétiques** et **préservatrices**, i.e., le mécanisme d'alignement ne doit pas modifier les modèles ni les langages de modélisation correspondants.

En appliquant ces trois restrictions, plusieurs approches ont été exclues. Si nous prenons l'hétérogénéité comme exemple, une dizaine d'approches traite uniquement des diagrammes UML, en s'intéressant à la cohérence entre des diagrammes de classes, de séquence et d'activités par exemple [ATKINSON et al., 2013; TÖRNGREN et al., 2014; MANSOOR et al., 2017]. Ces approches ne sont pas exploitables pour des acteurs utilisant des DSL propres à leurs métiers. D'autres approches sont spécifiques à un seul domaine d'application [FELDMANN et al., 2019].
Pour le nombre de modèles en entrée, de nombreuses approches exploitent les grammaires de triples graphes [SCHÜRR, 1995] pour définir des liens binaires [FOSTER et al., 2007; GIESE et al., 2010; HEIN et al., 2010; TROLLMANN et ALBAYRAK, 2016; DISKIN et al., 2019], or ces liens ne permettent pas d'avoir une vue globale sur le système en question, étant donné qu'ils n'alignent que deux modèles.

2.2.2 Critères d'analyse des approches existantes

Les critères que nous considérons pour analyser et comparer les approches existantes couvrent trois caractéristiques parmi les cinq présentées dans la section précédente, à savoir : (i) L'établissement des **correspondances**, (ii) La **gestion de la cohérence**, i.e., maintien de la cohérence en cas d'évolution des modèles source, et (iii) le support de la **collaboration**.
Nous avons écarté la caractéristique **artefacts** car son périmètre a été fixé (approches préservatrices avec plus de deux modèles hétérogènes en entrée), tandis que la caractéristique **outil support** est considérée comme transverse aux différentes caractéristiques.
Pour chacune des caractéristiques retenues, nous définissons des critères d'évaluation des approches étudiées comme suit :

Établissement des correspondances. Cette caractéristique décrit les solutions techniques que l'approche offre pour définir des correspondances entre les modèles.

Cette caractéristique est qualifiée par les critères suivants :

(i) *Arité* : spécifie l'arité des correspondances établies (i.e., binaire ou n-aire) ;

(ii) *Type de relation* : les correspondances étant typées, les relations qui les spécifient peuvent être figées, spécifiques à un domaine, ou bien libres. Les relations figées ou spécifiques à un domaine contraignent l'applicabilité de l'approche tandis que les relations libres favorisent la généricité de l'approche et son application à des systèmes issus de domaines métier variés ;

(iii) *Persistance des correspondances* : évalue si les correspondances sont explicitement persistantes à l'aide d'artefacts dédiés (modèle de correspondances par exemple) ;

(iv) *Outil support* : indique si l'approche propose un outil supportant l'établissement des correspondances.

Maintien de la cohérence. Elle caractérise l'approche en matière de maintien de la cohérence entre artefacts après changement. Elle est qualifiée par les critères suivants :

(i) *Détection des changements* : indique si l'approche fournit des mécanismes pour la détection des changements dans les artefacts (modèles, méta-modèles). Les mécanismes de détection connus sont à base d'états ou d'opérations [CONRADI et WESTFECHTEL, 1998; HERRMANNSDOERFER et KOEGEL, 2010] ;

(ii) *Résolution des incohérences* : indique si l'approche propose au moins un mécanisme pour le traitement des incohérences. Ce mécanisme peut être défini de manière opérationnelle grâce à un algorithme ou comme un solveur de contraintes. Il peut inclure des stratégies de résolutions et des techniques pour la gestion des conflits pouvant résulter de l'application d'une résolution donnée ;

(iii) *Outil support* : indique si l'approche propose un outil supportant le maintien de la cohérence.

Support de la collaboration. Elle précise en quoi l'approche répond aux aspects suivants de la collaboration : *communication*, *coordination*, *coopération* [ELLIS et al., 1991] et prise de décision en groupe (*co-décision*).

Nous écartons la communication et la coopération étant donné que la coordination est l'aspect central d'une collaboration ; dans une coopération, le travail doit se faire dans un espace de travail partagé et la communication, qu'elle soit formalisée ou non, a forcément lieu si plusieurs parties prenantes interagissent. Ainsi, cette caractéristique inclut-elle des critères concernant la coordination, la co-décision, et l'outil support.

(i) *Coordination* : caractérise l'organisation des personnes, de leurs activités interdépendantes et des ressources pour qu'elles travaillent ensemble efficacement ;

(ii) *Co-décision* : reflète le support par l'approche de la prise de décision en groupe ;

(iii) *Outil support* : indique si l'approche propose un outil supportant le déroulement des activités collaboratives.

2.3 Approches d'alignement de modèles hétérogènes

Dans cette section, nous présentons les principales approches de la littérature qui répondent à notre besoin d'alignement et de mise en cohérence de modèles hétérogènes. Les approches recensées sont : VirtualEMF [CLASEN et al., 2011], Alignment of Heterogeneous Models (AHM) [EL HAMLAOUI et al., 2014, 2018], EMFViews [BRUNELIERE et al., 2015], OpenFlexo [GUYCHARD et al., 2013; GOLRA et al., 2016], l'approche de VANHERPEN et al. [2016], l'approche de SHOSHA et al. [2015], et Compact Interactive Memetic Algorithm (CIMA) [XUE et LIU, 2017]. Chaque approche est décrite par son objectif, ses concepts clés en regard des trois caractéristiques présentées dans la section 2.2.2, et ses limitations.

2.3.1 VirtualEMF

Objectif

VirtualEMF [CLASEN et al., 2011] s'appuie sur la notion de modèle virtuel pour exprimer des liens entre modèles. En effet, un modèle virtuel est un modèle dont les éléments (virtuels) sont des proxys d'éléments contenus dans d'autres modèles. Il fournit aux outils/utilisateurs l'illusion de travailler avec un modèle réel alors qu'en fait, toutes les demandes de manipulation de modèle sont redirigées de manière transparente vers les éléments contenus dans les modèles dits « virtualisés ».

Établissement des correspondances

La Figure 2.2 présente le principe de la virtualisation des modèles et les artefacts impliqués dans VirtualEMF. Les outils (éditeurs, outils d'analyse et de transformation, etc.) utilisent le modèle virtuel comme un modèle normal. Le modèle virtuel délègue les opérations de modélisation aux modèles contributeurs *Ma* et *Mb* en localisant le ou les éléments référencés et en les traduisant en éléments virtuels utilisés par l'outil.
Un modèle de correspondances relie les éléments contributeurs et identifie la règle de traduction à utiliser pour composer chaque élément.
Le modèle virtuel composé est conforme au méta-modèle de composition qui fournit les concepts de base pouvant constituer le modèle composé. Le méta-modèle de composition de VirtualEMF est généré automatiquement à l'aide d'une transformation ATL [2] [JOUAULT et al., 2006].
Celle-ci prend comme entrée les méta-modèles source et un modèle de tissage établissant des liens entre eux, et génère en sortie le méta-modèle de composition peuplé des méta-éléments source liés selon le modèle de tissage.

2. ATLAS Transformation Language est un langage de transformation de modèles plus ou moins inspiré par le standard QVT de l'Object Management Group. Il est disponible en tant que plugin dans le projet Eclipse.

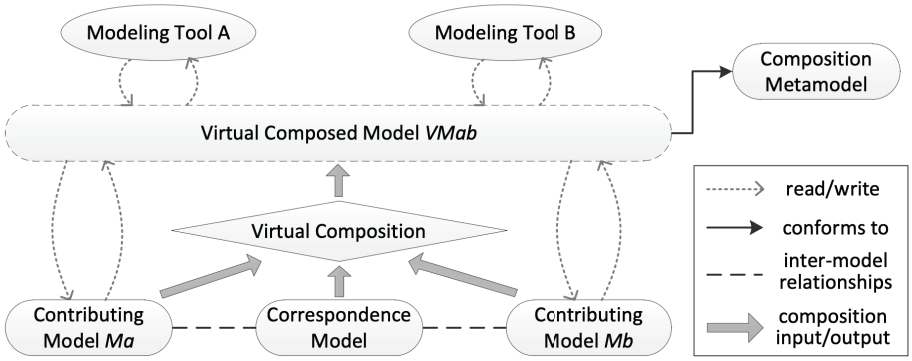


FIGURE 2.2 – Les artefacts de virtualisation de modèles dans VirtualEMF. [CLASEN et al., 2011]

L'API [3] de liaison utilise AMW [DEL FABRO et al., 2006] pour charger, enregistrer et manipuler des liens virtuels sous la forme d'éléments de modèle tissés. Les types possibles de liens virtuels sont définis dans un méta-modèle de tissage (e.g., *filter*, *merge*, *override*, *inherit*, ou *associate*) et l'API de liaison identifie comment gérer chaque type de lien.

Maintien de la cohérence

La virtualisation du modèle composé assure une synchronisation totale entre les modèles source et composé puisque le modèle virtuel utilise les mêmes instances d'éléments que les modèles source. Cela garantit la cohérence, évite d'avoir des données dupliquées (moins d'utilisation de la mémoire), et permet une création plus rapide du modèle composé (pas besoin d'éléments clonés).

Support de la collaboration

La collaboration n'est pas intégrée dans l'approche ; le modèle de tissage est défini par un seul acteur.

Limitations

Le modèle de tissage de VirtualEMF est établi de façon déclarative et la collaboration n'est pas couverte par l'approche : un seul acteur est responsable de la définition du modèle de tissage et des liens. La synchronisation entre les modèles source et le modèle composé est assurée puisque le modèle composé est virtuel, mais les incohérences inter-modèles qui peuvent apparaître suite aux évolutions conjointes des modèles source ne sont pas prises en compte.

2.3.2 AHM

Objectif

L'approche Alignment of Heterogeneous Models (AHM) [EL HAMLAOUI et al., 2014, 2018] porte sur la multi-modélisation des systèmes complexes. C'est une approche de mise en correspondance et de maintien de la cohérence entre modèles hétérogènes.

3. Une API est une interface de programmation d'application ou interface de programmation applicative. C'est un ensemble normalisé de classes, de méthodes, de fonctions et de constantes qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels.

Établissement des correspondances

L'approche AHM définit deux processus. Le premier processus permet la création d'une vue globale du système par la mise en correspondance des modèles source. La création de cette vue globale appelée dans l'approche « *modèle de correspondances* » s'appuie sur le concept de virtualisation. Ainsi, ce modèle n'inclut pas de données concrètes mais seulement les correspondances qui relient des éléments des modèles source.
L'approche AHM propose un Méta-Modèle de Correspondances (MMC) dont le noyau est présenté sur la Figure 2.3 avec des types de relations génériques, indépendantes du domaine d'application (Domain Independant Relatioship (DIR)).
MMC est extensible au sens où il prévoit l'ajout de types de relations spécifiques à un domaine d'application donné (Domain Specific Relatioship (DSR)). Une correspondance est composée d'au minimum deux éléments référencés et du type de relation (DIR ou DSR).
MMC distingue deux niveaux où le type de relation est applicable : les High Level Relationship (HLR) qui sont adaptées aux correspondances de niveau méta-modèle, et les Low Level Relationship (LLR) qui s'appliquent au niveau modèle.

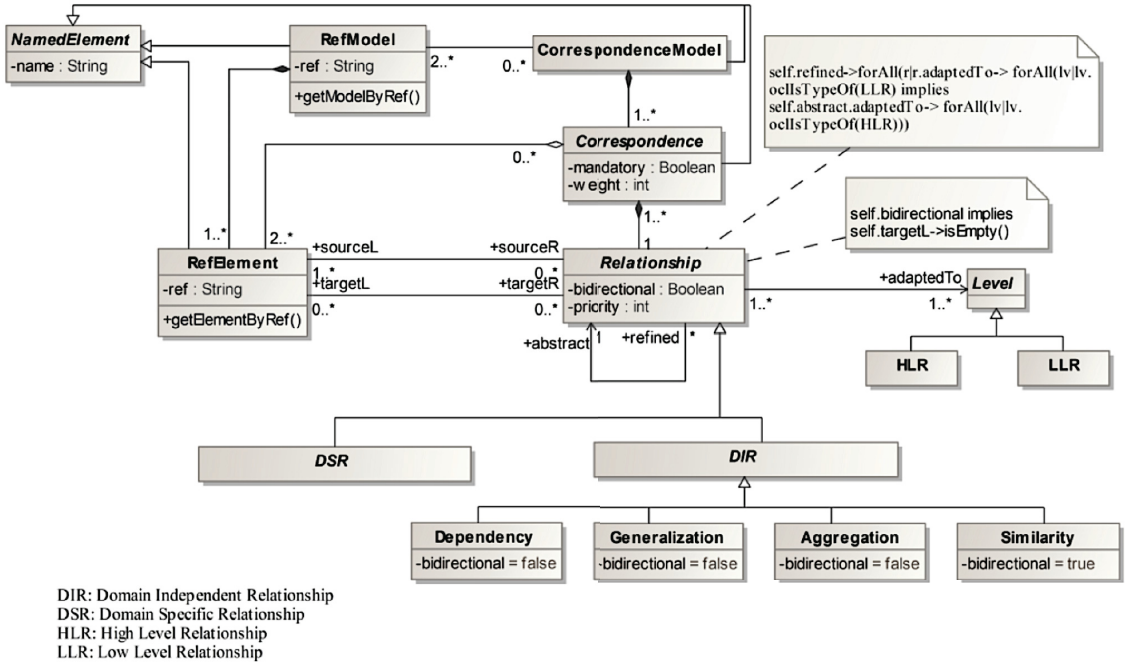


FIGURE 2.3 – Noyau du méta-modèle de correspondances dans AHM. [EL HAMLAOUI, 2015]

Maintien de la cohérence

Le second processus proposé par AHM permet de maintenir la cohérence des modèles source et du modèle de correspondances en cas d'évolution de modèles impliquant un ou plusieurs éléments dans des correspondances. En effet, l'évolution des modèles risque d'engendrer des problèmes à partir du moment où les modèles sont liés et où la modification de l'un d'eux peut entraîner l'incohérence du système entier, d'où la nécessité de répercuter les modifications, ou tout au moins d'identifier les éléments de modèles qui sont impactés par les changements.
La propagation des changements dans AHM est semi-automatisée; elle se limite à l'ajout, la modification et la suppression des éléments de modèles source, et est réalisée en définissant l'ensemble des changements pris en compte et la manière de répercuter leurs impacts sur les autres modèles.

Support de la collaboration

L'approche est mono-utilisateur en supposant qu'un rôle expert peut dérouler les deux processus de l'approche.

Limitations

Les limitations de l'approche AHM sont dues principalement au couplage fort de l'approche avec le rôle de l'expert. En effet, plus le système est large et intègre des points de vue métier diversifiés, plus l'existence de cet expert est problématique lors de la conception de systèmes complexes. Le principe de séparation des préoccupations, qui justifie la conception multi-vue, perd sa force quand on attribue à une seule personne à la fois le rôle d'établir les correspondances, de les maintenir ou d'en supprimer en cas d'évolution des modèles.
L'approche AHM considère que l'expert peut s'orienter vers les concepteurs des points de vue métier pour bénéficier de leur assistance, mais en réalité, les choix et les décisions lors de l'établissement des correspondances doivent émaner de ces concepteurs car ce sont eux qui maîtrisent les exigences et les contraintes de leurs métiers.

2.3.3 EMF Views

Objectif

EMF Views [Bruneliere et al., 2015] adopte le concept de « *vue* » des bases de données et le transpose au monde de la modélisation. Elle permet de créer des vues qui agrègent des éléments provenant de différents modèles.
EMF Views est entièrement compatible avec l'API EMF et les vues qu'elle produit agissent comme des modèles standards : elles peuvent être parcourues, interrogées et prises en tant qu'entrées pour des transformations de modèles.

Établissement des correspondances

EMF Views propose une approche générique permettant de construire des vues sur tout ensemble de modèles inter-reliés, conformes à des méta-modèles potentiellement différents. Elle fournit un mécanisme en deux étapes qui sépare explicitement la spécification des points de vue, de la réalisation et gestion des vues correspondantes.
Une vue se compose d'un ensemble d'éléments qui pointent vers des éléments concrets des méta-modèles de base référencés dans la vue, et de nouvelles relations croisées entre eux.
Au moment de la conception, les concepteurs peuvent spécifier un nouveau point de vue en choisissant le(s) méta-modèle(s) concerné(s), et en énumérant les relations qu'ils veulent représenter entre eux. Ces informations sont directement collectées auprès du concepteur manuellement ou à l'aide d'un DSL. Ce DSL permet d'exprimer les principales opérations nécessaires : sélection (*select*), projection (*project*) et jointure (*join*). Il est fortement inspiré du langage SQL ce qui facilite son adoption par les utilisateurs potentiels.
Les données sont stockées dans un modèle de tissage qui est ensuite utilisé par le mécanisme de virtualisation pour obtenir le point de vue réel. Par conséquent, le ou les méta-modèles d'origine ne sont pas modifiés par la définition du point de vue. La Figure 2.4 donne un aperçu global de l'approche EMF Views. De façon similaire aux opérations *select-project-join* en algèbre relationnelle, la définition d'un point de vue spécifie quelles classes des méta-modèles doivent faire partie (ou inversement, doivent être filtrées) de la vue (*projection*), quelles conditions doivent être satisfaites pour apparaître comme résultat dans la requête d'affichage (*sélection*) et comment les éléments des différents modèles doivent être liés lors du calcul de la vue réelle (*jointure*).

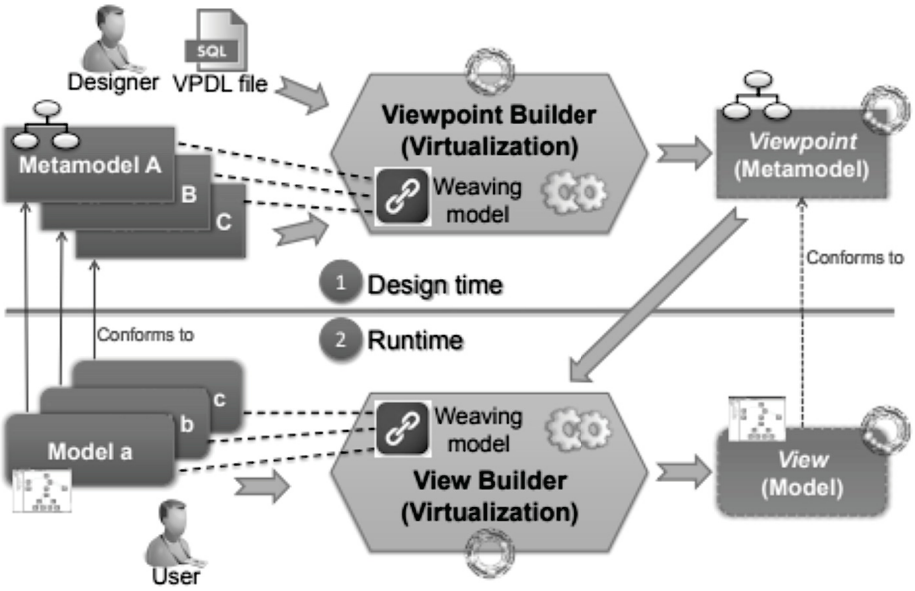


FIGURE 2.4 – Schéma global de l'approche EMF Views. [BRUNELIERE et al., 2015]

Maintien de la cohérence

Les vues établies sont en lecture seule avec la possibilité de propager aux modèles source les modifications apportées aux valeurs d'attributs dans les vues. Dans BRUNELIERE et al. [2018], les auteurs ont intégré NeoEMF [4] [DANIEL et al., 2017] et CDO [5] dans leur approche pour rendre les vues créées persistantes.

Support de la collaboration

Au moment de la conception, un seul concepteur (rôle *Designer* sur la Figure 2.4) définit les relations entre méta-modèles source afin de construire le nouveau point de vue. Il n'y a donc pas de collaboration.

Limitations

Le maintien de la cohérence est actuellement partiellement supporté. En effet, seules les modifications apportées aux valeurs d'attribut dans les vues sont propagées aux modèles d'origine. D'autre part, l'approche ne considère pas les conflits pouvant se produire en cas d'évolution conjointe des modèles, puisque la synchronisation se fait entre les modèles source d'un côté et la vue de l'autre côté. Lors de l'établissement des liens, l'approche adopte un mécanisme déclaratif (définition des liens à base du DSL fourni) où un seul acteur peut définir les liens inter-modèles.

4. NeoEMF est une solution multi-base de persistance de modèles, capable de stocker des modèles dans plusieurs types de *Datastores* NoSQL.

5. Connected Data Objects pour les objets de données connectés en français. C'est une implémentation libre d'un modèle partagé distribué dans le framework de modélisation EMF. Voir : https://wiki.eclipse.org/CDO

2.3.4 OpenFlexo

Objectif

OpenFlexo [GUYCHARD et al., 2013; GOLRA et al., 2016] fédère des modèles provenant de différents paradigmes et espaces techniques, permettant ainsi la synchronisation et la gestion de la cohérence entre les données produites par des spécialistes de différents domaines métier.

Établissement des correspondances

Le cœur de l'infrastructure (*OpenFlexo core*) contient des composants dédiés à la mise en oeuvre de la fédération de modèles. Cette technique autorise la construction de nouveaux modèles à partir d'un ensemble de sources de données (e.g., EMF, XML) qui sont vues comme des modèles hétérogènes. Les modèles en entrée sont modélisés avec des paradigmes différents qui répondent aux besoins métier de leurs concepteurs. Ils sont appelés *modèles d'espaces technologiques*.
L'approche définit des *espaces conceptuels* où un ensemble de modèles peut être fédéré pour développer un nouveau point de vue / modèle intersectoriel, appelé *modèle virtuel*.
Les modèles virtuels sont constitués : (i) d'éléments qui sont réutilisés à partir des modèles d'espaces technologiques et (ii) d'éléments qui n'appartiennent à aucun espace technologique existant. Les modèles virtuels sont conformes à un langage de modélisation générique spécialement conçu pour la fédération de modèles.
Une connexion est nécessaire entre les espaces technologiques et conceptuels pour rendre disponibles les caractéristiques d'un modèle technologique dans l'espace conceptuel. Une connexion entre un espace technologique spécifique et l'espace conceptuel est réalisée à l'aide de connecteurs technologiques. Ces connecteurs permettent un accès en lecture et en écriture aux modèles. Ils permettent aussi de synchroniser les informations entre les modèles virtuels et les modèles technologiques, car quand un connecteur technologique est disponible pour un espace technologique spécifique, tous les modèles de cet espace et leurs éléments respectifs deviennent accessibles depuis l'espace conceptuel. Les éléments d'un modèle technologique peuvent être réutilisés de deux manières différentes :

(i) Le concepteur peut créer un lien dynamique entre un concept local du modèle conceptuel et le concept de modèle technologique qu'il souhaite utiliser, comme le montre le modèle virtuel A de la Figure 2.5 ;

(ii) Le concepteur peut également choisir de créer un « proxy » local de ce concept dans le modèle virtuel en le traduisant en un concept Flexo distant. Un concept Flexo distant, comme illustré sur le modèle virtuel B de la Figure 2.5, est une substitution du concept du modèle technologique et introduit donc une certaine redondance.

Maintien de la cohérence

Une fois les éléments du modèle technologique réutilisés dans le modèle virtuel, que cela soit par des liens dynamiques ou des proxys de concepts, les connexions sont conservées pour synchroniser les modifications ultérieures.
La connexion entre les espaces conceptuels et technologiques est bidirectionnelle. Un modèle virtuel dans l'espace conceptuel peut être mis à jour lorsque le modèle technologique correspondant est modifié et inversement.
L'approche fournit deux mécanismes pour la mise à jour des informations dans les deux sens. En effet, le concepteur peut choisir de recevoir une notification lorsqu'un modèle est modifié ou bien choisir de le synchroniser automatiquement, en fonction de la sémantique définie dans le connecteur technologique.
Un rôle *expert en outillage* définit les primitives de manipulation des concepts fédérés : *opérations CRUD* (Create/Read/Update/Delete), *opérations de synchronisation*, *transformations*, *contraintes*. Des modifications simultanées des parties communes des modèles connectés peuvent provoquer des conflits. Ces conflits sont résolus manuellement par les concepteurs.

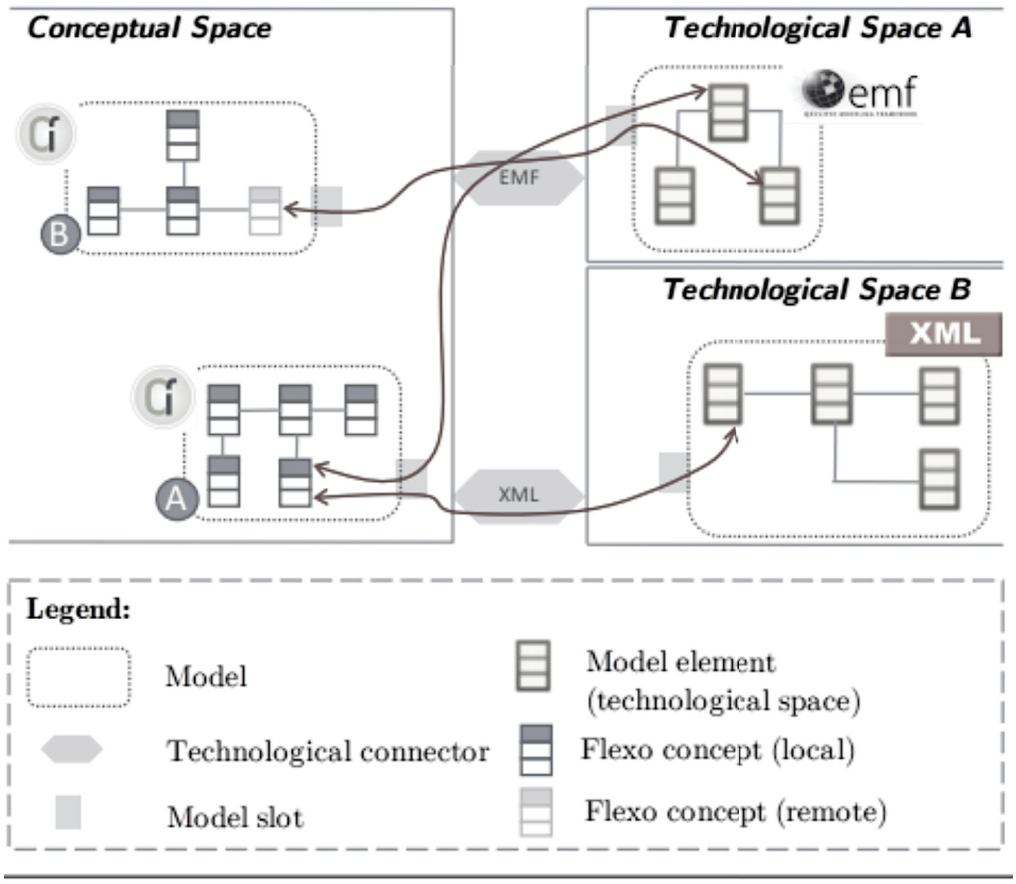


FIGURE 2.5 – Espace de modélisation dans OpenFlexo. [GOLRA et al., 2016]

Support de la collaboration

Le processus de conception des modèles virtuels dans OpenFlexo est collaboratif : les concepts fédérés sont définis par l'expert du domaine et conçus par l'expert en outillage : l'expert en outillage commence par le développement des concepts Flexo (concepts du modèle virtuel) conformément aux spécifications de l'expert du domaine. Ensuite, les concepts nécessaires pour compléter le modèle virtuel sont identifiés dans les espaces technologiques par l'expert du domaine. Enfin, les correspondances sont créées en se basant soit sur des liens dynamiques, soit sur les proxys.

Limitations

L'approche OpenFlexo propose des connecteurs technologiques pour plusieurs paradigmes (EMF, OWL, XML/XSD, MS Office, etc.). Toutefois, d'autres espaces technologiques sont à considérer et l'approche n'offre pas de langage de haut niveau pouvant remplacer le code Java définissant les interfaces des modules.
En ce qui concerne la préservation de la cohérence, les changements sont synchronisés manuellement s'il n'y a pas une sémantique derrière les liens tandis que les conflits entre les modèles en entrée résultant des changements sont gérés manuellement par les experts, ce qui est à la fois propice aux erreurs et fastidieux.
Pour le support à la collaboration, l'approche propose deux rôles distincts : expert du domaine et expert de l'outillage. Le rôle d'expert de domaine peut être joué par autant d'acteurs que d'espaces technologiques impliqués, mais l'outil proposé ne supporte pas l'interaction de ces acteurs et la prise de décision en groupe ; en effet, l'approche impose les correspondances binaires en reliant à chaque fois un élément d'un espace technologique à un élément de l'espace conceptuel.

2.3.5 Approche de Vanherpen

Objectif

VANHERPEN et al. [2016] proposent un cadre permettant de définir les relations entre différents points de vue d'un système cyber-physique en se basant sur des propriétés linguistiques et ontologiques explicitement modélisées.

Établissement des correspondances

L'approche distingue deux types de modèles : Linguistic Type Model (LTM) et Ontological Type Models (OTM) comme illustré sur la Figure 2.6.
Les modèles LTM sont les modèles issus du monde réel (i.e., modèles des points de vue métier). Les modèles OTM représentent la connaissance implicite de l'ingénieur. Un modèle OTM catégorise ou classifie des entités du monde réel en fonction de propriétés. Celles-ci sont reliées logiquement en utilisant une logique appropriée (par exemple, une logique de description). Chaque ontologie est également conforme à un modèle de type linguistique (LTM) car l'ontologie doit également être modélisée à l'aide d'un langage.

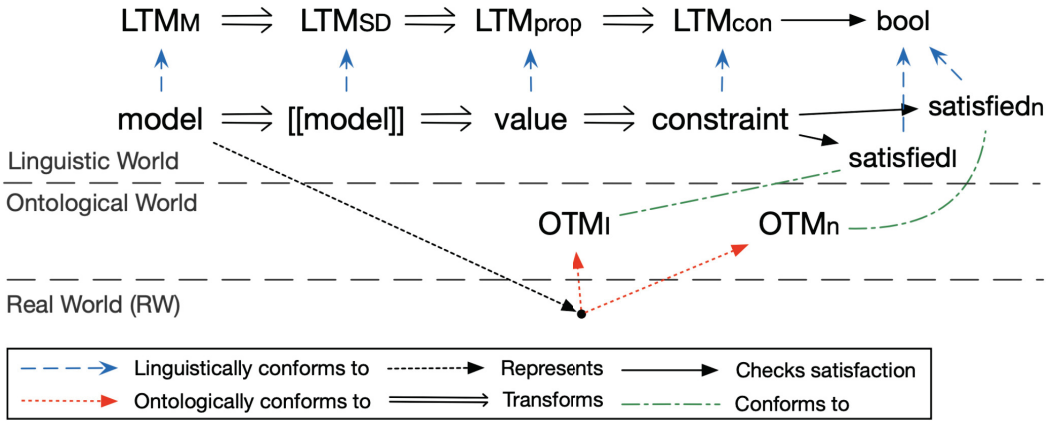


FIGURE 2.6 – Modèles linguistiques versus modèles ontologiques. [VANHERPEN et al., 2016]

Les ontologies et les modèles linguistiques sont liés les uns aux autres à travers une *relation de satisfaction* qui doit exister entre leurs propriétés respectives. En d'autres termes, chaque propriété linguistique issue d'un domaine sémantique peut être liée à une propriété ontologique. Cela implique que les propriétés linguistiques issues de différents domaines sémantiques peuvent être reliées entre elles par le biais d'une ontologie commune ou d'un ensemble d'ontologies.
En raison du chevauchement des exigences, certaines propriétés ontologiques peuvent être partagées et/ou s'influencer mutuellement. Par conséquent, le cadre proposé relie les propriétés ontologiques et définit une fonction de satisfaction pour chacune d'elles. Cette fonction prend en compte les propriétés linguistiques qui affectent la propriété ontologique.

Maintien de la cohérence

Les auteurs proposent trois patrons qui décrivent comment les propriétés linguistiques et ontologiques sont inter-reliées, à savoir : *multi-sémantique*, *multi-abstraction* et *multi-vue*. Ces patrons permettent de détecter les incohérences possibles entre les différents points de vue modélisés, et selon les incohérences identifiées, le concepteur devra résoudre le problème de manière à ce que toutes les propriétés ontologiques soient satisfaites.

Support de la collaboration

Cet aspect n'est pas pris en compte par l'approche.

Limitations

L'approche propose la notion de propriété ontologique pour relier les éléments de différents points de vue et détecter les incohérences potentielles mais c'est aux concepteurs de résoudre manuellement les incohérences détectées. De plus, les auteurs de l'approche ne décrivent pas comment les propriétés ontologiques et les valeurs de performance sont établies et vérifiées pour préserver la cohérence inter-modèles.

2.3.6 Approche de Shosha pour l'alignement d'ontologies

Nous abordons dans le reste de la section 2.3 des approches d'alignement collaboratif d'ontologies. Le choix de ces approches est justifié par deux raisons : d'une part, le passage de l'espace des ontologies vers celui des (méta-)modèles est possible grâce à des transformations [FAUCHER et al., 2008], et d'autre part, l'aspect collaboratif n'est pas au centre de la plupart des approches que nous venons de présenter alors qu'il est primordial dans notre problématique de recherche.

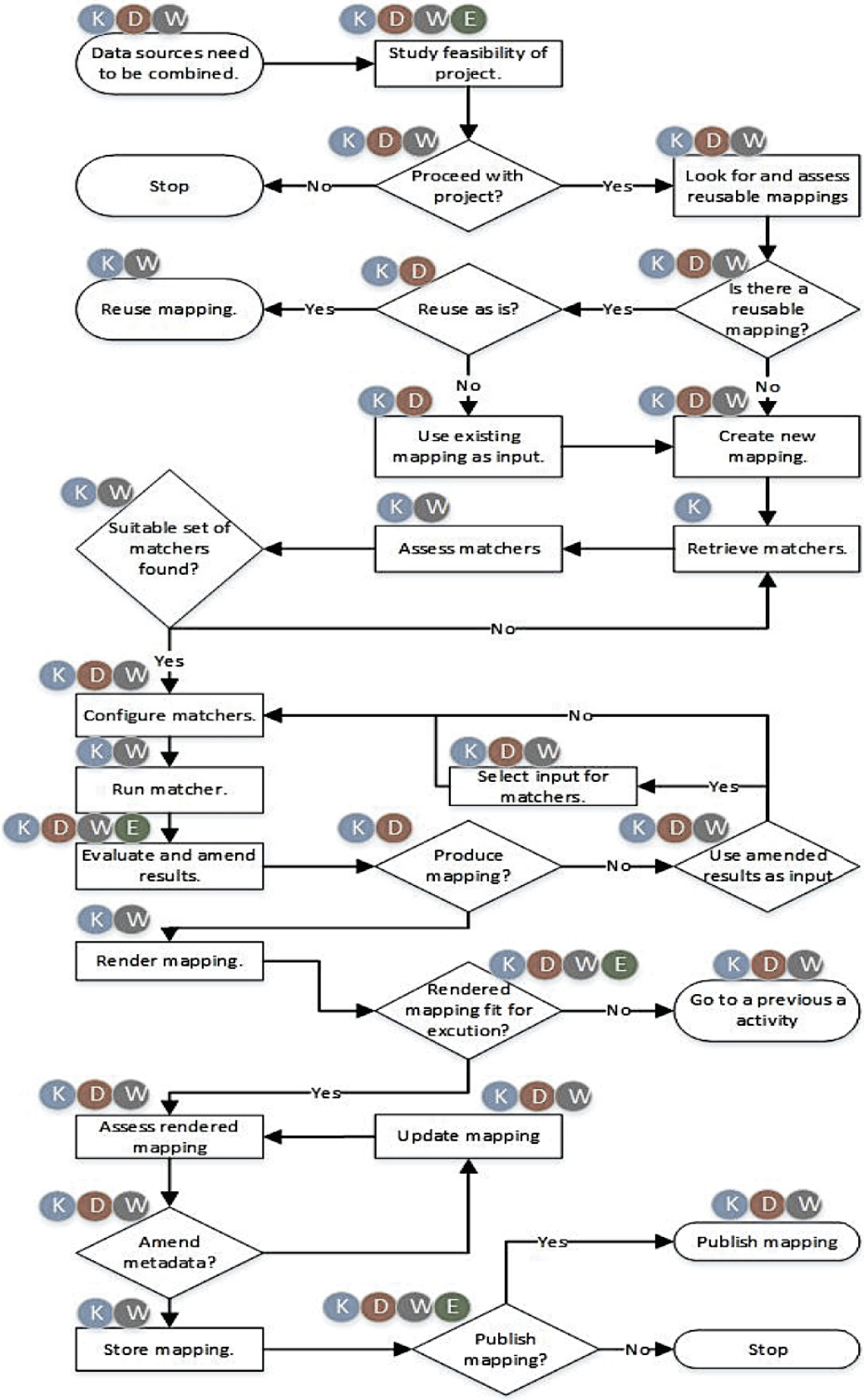
Objectif

SHOSHA et al. [2015] définissent une approche collaborative pour la mise en correspondance d'ontologies dont les données sont sémantiquement liées, et pour leur fusion dans une application composite « *mashup* » (ontologie cible du domaine).

Établissement des correspondances

Le processus de mise en correspondance, illustré sur la Figure 2.7, commence par la définition des sources de données qui doivent être combinées. Après regroupement de ces sources, une étude de faisabilité est réalisée pour identifier l'ensemble des exigences que doit satisfaire l'application composite. Ensuite, deux scénarios sont possibles pour établir les correspondances entre les ontologies : la création d'un nouvel ensemble de correspondances, ou bien la réutilisation d'un mapping [6] déjà existant.
L'une des difficultés dans ce processus est l'effort humain requis pour choisir entre réutiliser des anciens mappings ou bien tout reconstruire. C'est la responsabilité des ingénieurs en connaissances (identifiés sur la Figure 2.7 par la lettre K), des développeurs (W) et des experts du domaine (D) (ces rôles seront détaillés juste après). S'ils choisissent de réutiliser un mapping déjà existant, les ingénieurs en connaissance et les experts du domaine doivent vérifier si ce dernier satisfait les besoins métier et s'il pourra donc être réutilisé tel quel ; sinon, il sera utilisé comme point d'entrée pour produire de meilleurs résultats en impliquant aussi les développeurs.
À la fin de ce processus, le résultat du mapping est fourni à toutes les parties prenantes pour évaluation et obtention d'un consensus sur l'opportunité de procéder à la création de l'application composite ou sinon pour revenir à une étape particulière pour générer de meilleurs résultats. Les accords sont obtenus par le biais de discussions et de votes.

6. Le processus de détermination des correspondances entre des concepts de deux ontologies.



K : Ingénieur en connaissances, **D** : Expert du domaine, **W** : Développeur, **E** : Utilisateur final

FIGURE 2.7 – Processus de mise en correspondance collaborative dans l'approche de SHOSHA et al. [2015].

Le processus tel que défini garantit la séparation des préoccupations puisque chaque acteur agit dans les étapes où il a la connaissance nécessaire.

Maintien de la cohérence

L'approche permet de réutiliser un mapping déjà existant s'il répond aux besoins identifiés par les experts du domaine mais ne fournit pas de mécanisme d'adaptation de mappings existants. Elle ne s'intéresse pas au maintien de la cohérence.

Support de la collaboration

L'approche de SHOSHA et al. [2015] gère les interactions entre quatre types d'utilisateurs pour mettre en correspondance les ontologies source et définir la sémantique de l'application *mashup*, en intégrant des discussions et un système de vote. Ces quatre types d'acteurs sont : *ingénieur en connaissance* (K), *expert du domaine* (D), *développeur* (W), *utilisateur final* (E).
Les *ingénieurs en connaissance* ont une expertise dans la modélisation des connaissances sémantiques et les compétences nécessaires pour utiliser les outils de l'ontologie. Cependant, ils ne sont pas nécessairement familiers de la terminologie du domaine d'application. Ils se coordonnent avec les *experts du domaine* pour cela.
Les *développeurs* sont familiarisés avec le développement d'applications, il ne le sont pas forcément avec le domaine d'application ou les technologies sémantiques. Les *utilisateurs finaux* sont les acteurs auxquels l'application *mashup* est destinée, mais ils ne sont pas nécessairement des technophiles.

Limitations

L'approche de SHOSHA et al. [2015] n'inclut pas dans ses objectifs la gestion de la cohérence du mapping en cas d'évolution des sources de données.

2.3.7 CIMA pour le matching collaboratif d'ontologies

Objectif

XUE et LIU [2017] proposent une approche fondée sur l'algorithme Compact Interactive Memetic Algorithm (CIMA) pour la mise en correspondance collaborative d'ontologies. Cet algorithme soutient la collaboration des utilisateurs en optimisant leur implication par la détermination du moment de leur implication, en leur présentant les correspondances les plus problématiques et en les aidant à valider automatiquement plusieurs mappings conflictuels.

Établissement des correspondances

Les auteurs proposent un cadre de mise en correspondance collaborative des ontologies composé de quatre phases comme présenté sur la Figure 2.8.

(i) Partition d'ontologies : il s'agit d'une phase de pré-appariement, qui vise à réduire le nombre de correspondances candidates présentées pour validation par les utilisateurs en partitionnant deux ontologies en paires de segments similaires ;

(ii) Application de l'algorithme CIMA : il intervient dans la phase d'appariement automatique, qui vise à déterminer les correspondances potentielles et à déterminer de manière adaptative le moment de l'implication de l'utilisateur. Le processus évolutif de CIMA est susceptible de rester bloqué en raison du grand espace de recherche et parce que certaines entités ne sont pas facilement distinguées les unes des autres par le biais de la mesure de similitude ; lorsque cela se produit, l'utilisateur s'implique pour guider le processus de CIMA ;

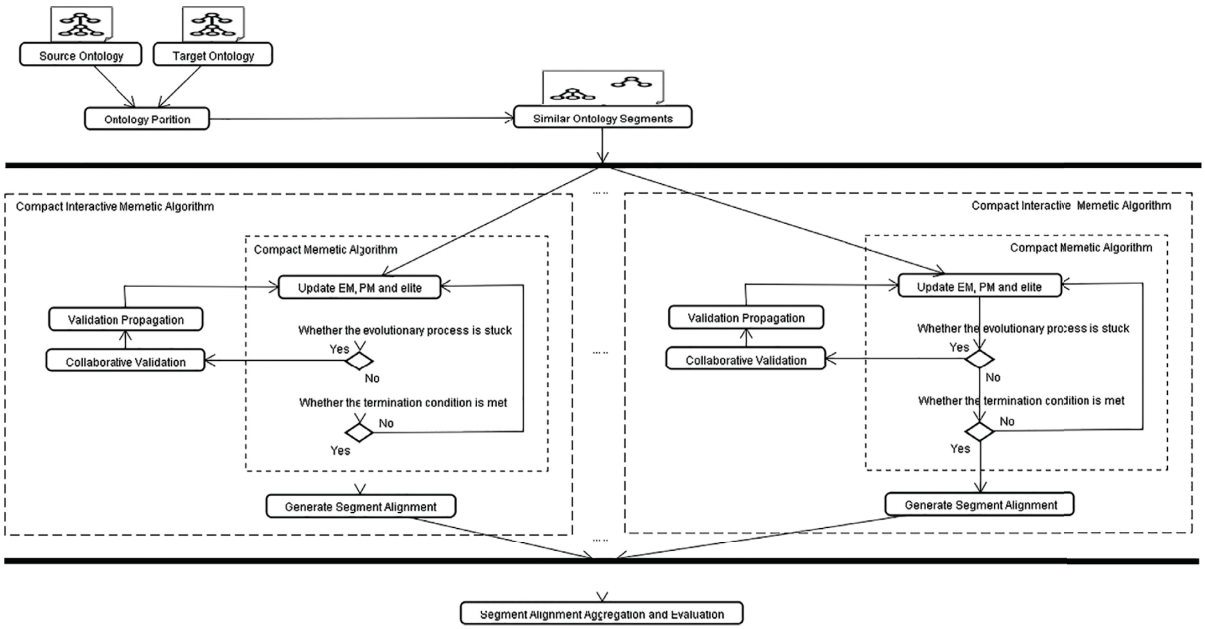


FIGURE 2.8 – Cadre collaboratif de mise en correspondance d'ontologies. [XUE et LIU, 2017]

(iii) Validation collaborative et propagation de la validation : c'est la phase interactive, les mappings problématiques sont automatiquement déterminés pour faire l'objet d'une validation collaborative par les acteurs afin d'introduire de nouveaux mappings et de filtrer ceux qui sont erronés. Après cela, la validation est faite et propagée pour maximiser l'exploitation des conseils des utilisateurs ;

(iv) Évaluation de l'agrégation d'alignement de segment : les alignements de segment sont agrégés en un alignement final qui est ensuite évalué par la mesure *F-mesure*[7].

Maintien de la cohérence

Le centre d'intérêt de l'approche est l'établissement des mappings. Elle ne s'intéresse pas au maintien de la cohérence.

Support de la collaboration

Pour réduire la charge de travail des utilisateurs, mais en même temps augmenter la valeur de leur implication, l'approche propose un mécanisme d'alignement semi-automatique qui ne fait intervenir les acteurs que quand le processus automatique bloque, i.e., il n'a pas été capable de valider les correspondances automatiquement car elles présentent des conflits. Ainsi, l'approche allie à la fois collaboration et automatisation en allégeant la participation des utilisateurs lors de la validation de l'alignement.

7. F-mesure combine la précision et le rappel. C'est leur moyenne harmonique.

Limitations

Comme pour l'approche de SHOSHA et al. [2015], CIMA n'inclut pas dans ses objectifs la gestion de la cohérence du mapping en cas d'évolution des sources de données. Elle est limitée aussi en terme de type de relations gérées entre les ontologies source (relations de similarité et d'équivalence uniquement) et en terme de cardinalité des correspondances (binaires exclusivement).

2.4 Comparaison des approches

Le tableau 2.1 résume la comparaison des approches présentées dans la section précédente, à savoir : VirtualEMF [CLASEN et al., 2011], AHM [EL HAMLAOUI et al., 2014, 2018], EMFViews [BRUNELIERE et al., 2015], l'approche de SHOSHA et al. [2015], OpenFlexo [GOLRA et al., 2016; GUYCHARD et al., 2013], l'approche de VANHERPEN et al. [2016] et CIMA [XUE et LIU, 2017].
La comparaison de ces approches a été faite selon les critères associés aux trois caractéristiques présentées dans la section 2.2.2 : *Établissement des correspondances*, *Maintien de la cohérence*, *Support de la collaboration*.

Établissement des correspondances. En ce qui concerne l'établissement des correspondances, quatre des sept approches étudiées permettent d'établir des correspondances à la fois binaires et n-aires ; alors que VirtualEMF, l'approche de Shosha et CIMA se contentent d'établir des correspondances entre deux éléments.

Les relations utilisées dans la définition des correspondances sont figées dans un ensemble prédéfini (Similarité, Transformation, etc.) sauf pour AHM et OpenFlexo.

Concernant les correspondances produites, elles ne sont persistantes que dans AHM, VirtualEMF et EMF Views (si on considère son extension BRUNELIERE et al. [2018]), alors qu'il est nécessaire d'augmenter l'exploitabilité des correspondances étant donné que leur nombre peut être très important [BRYANT et al., 2015] ; la persistance des correspondances pourrait être exploitée pour la gestion de la cohérence tout au long de l'existence des modèles source.

Maintien de la cohérence. En ce qui concerne le maintien de la cohérence entre modèles, les approches supportant cet aspect gèrent uniquement les répercussions des changements atomiques de niveau modèle.

Notons que pour la résolution des incohérences, la majorité des approches s'appuie sur l'acteur humain et n'inclut pas de mécanisme de résolution (sauf pour AHM, OpenFlexo et l'approche de Vanherpen). D'autre part, les approches n'incluent pas, dans leurs mécanismes de résolution des incohérences, de méthode(s) pour gérer les conflits. Ainsi, l'application d'une stratégie de résolution peut entraîner d'autres incohérences. AHM et OpenFlexo se distinguent sur ce volet, car les deux approches calculent les impacts des changements et les conflits qu'ils peuvent produire et délèguent leurs traitements aux acteurs effectuant l'alignement.

À part les approches présentées dans ce chapitre, il existe plusieurs approches de co-évolution méta-modèle/modèle ou même d'évolution (méta-)modèle/transformations [CICCHETTI et al., 2008a; ROSE et al., 2010; KHELLADI et al., 2018]. Ces approches permettent de faire évoluer les modèles ou les transformations en fonction des changements faits au niveau méta-modèle et de résoudre les éventuelles incohérences produites. Mais ces approches ont deux limitations majeures qui font qu'elles n'ont pas été retenues dans cette revue : (i) les liens qu'elles définissent concernent uniquement des paires de modèles deux à deux, ce qui a pour conséquence la création d'un grand nombre de modèles de correspondances séparés, sans lien entre eux, ce qui rend leur gestion difficile et presque impossible à automatiser ; (ii) l'expressivité des liens est limitée par le langage de transformation utilisé.

D'autres approches de la littérature fournissent des mécanismes pour maintenir la cohérence inter-modèles même avec des changements composites. L'intérêt de ces approches est de structurer les changements atomiques en changements à un niveau d'abstraction plus élevé

Etablissement de correspondances	*VirtualEMF*	*AHM*	*EMF Views*	*Appr. Shosha*	*OpenFlexo*	*Appr. Vanherpen*	*CIMA*
Arité binaire	●	●	●	●	●	●	●
Arité n-aire (n≥3)	-	●	●	-	●	●	-
Type de relation libre	◐	●	-	-	●	-	-
Persistance des correspondances	●	●	◐	-	-	-	-
Outil support	●	●	●	●	●	◐	◐
Maintien de cohérence							
Détection des changements	◐	◐	◐	-	◐	◐	-
Résolution des incohérences	-	◐	-	-	◐	◐	-
Outil support	◐	◐	◐	-	◐	◐	-
Support de collaboration							
Coordination	-	-	-	●	●	-	●
Codécision	-	-	-	●	-	-	●
Outil support	-	-	-	●	-	-	●

● : Critère au centre de l'approche, ◐ : Partiellement considéré, - : Non considéré

TABLEAU 2.1 – Synthèse des caractéristiques supportées par les approches d'alignement de modèles hétérogènes.

qui représentent les unités conceptuelles auxquelles les experts du domaine sont habitués. Nous citons à titre d'exemples, le travail de WIMMER et al. [2012] qui se base sur le langage MAUDE [CLAVEL et al., 2007] et les transformations couplées, et le travail de VERMOLEN et al. [2011] qui concerne les méta-modèles basés sur Ecore. Cependant, ces travaux ne détaillent pas comment les changements composites sont propagés aux modèles.

Support de la collaboration. L'aspect collaboratif dans les approches d'alignement de modèles hétérogènes est peu supporté.

Plusieurs approches ne le ciblent pas (EMF Views, AHM, OpenFlexo et l'approche de Vanherpen), en supposant qu'un expert peut effectuer toutes les tâches manuelles. D'autres n'adoptent pas de démarche dans la description de leurs mécanismes d'alignement. De ce fait, elles décrivent les techniques d'alignement mais les outils proposés ne gèrent pas la collaboration ou le multi-usage.

Seules les deux approches issues du domaine des ontologies se distinguent en supportant à la fois la coordination et la co-décision entre plusieurs acteurs. Mais ces approches comme nous l'avons vu proposent un seul type de relation (similarité) et ne traitent pas le maintien de la cohérence en cas d'évolution.

2.5 Conclusion

Dans ce chapitre, nous avons étudié les approches d'alignement de modèles hétérogènes et de gestion de la cohérence dans le cadre d'une conception multi-vue. Tout d'abord nous avons défini le périmètre de l'étude effectuée. Ensuite, nous avons élaboré une liste de critères pour l'analyse de ces approches.
L'analyse a porté sur sept approches représentatives, à savoir VirtualEMF, EMF Views, AHM, OpenFlexo, l'approche de Vanherpen, l'approche de Shosha et CIMA.
La revue des travaux réalisée montre qu'ils sont essentiellement focalisés sur les mécanismes IDM utilisés pour formaliser l'alignement, et que plusieurs approches considèrent qu'un seul rôle est capable de gérer à lui seul l'alignement. La non prise en compte de l'aspect participatif des acteurs lors de la définition des correspondances ou bien lors de leur gestion après évolution remet en cause fortement la pertinence et la qualité de l'alignement obtenu.
Par ailleurs, les modèles étant évolutifs, la modification de l'un d'eux peut entraîner l'incohérence de l'ensemble d'où la nécessité de répercuter les modifications. Or, d'après la revue de la littérature réalisée, rares sont les approches qui traitent l'aspect évolutif des modèles. Ceci vient du fait que ces approches ne rendent pas persistantes les correspondances établies ou bien qu'elles se contentent de détecter les changements sans offrir de mécanisme pour la résolution des incohérences et des conflits inter-modèles engendrés par les changements.

Dans la partie **contributions** de ce manuscrit, nous répondons à ce besoin d'alignement collaboratif qui tienne compte des évolutions des modèles. Mais avant cela, nous présentons dans le chapitre suivant une étude bibliographique des approches de description de la prise de décision collaborative. L'étude de ces approches a pour principal objectif de pouvoir allier alignement et collaboration afin d'assurer la pertinence et la complétude des alignements obtenus (étant donné l'implication des acteurs métier lors du processus d'alignement).

Chapitre 3

Collaboration pour la prise de décision en groupe

Sommaire

3.1 Introduction 33
3.2 Critères d'analyse des approches de modélisation des processus GDM 34
3.2.1 Périmètre de l'étude de la littérature 34
3.2.2 Critères d'analyse des approches existantes 36
3.3 Approches de modélisation des processus GDM 37
3.3.1 DSO 38
3.3.2 MADISE 39
3.3.3 OntoGDSS 41
3.3.4 Approche de Malavolta 43
3.3.5 Collaboro 45
3.4 Comparaison des approches 47
3.5 Conclusion 49

3.1 Introduction

La collaboration est le fait d'impliquer des participants ayant des intérêts variés mais qui opèrent ensemble pour atteindre un objectif commun [De Vreede et Briggs, 2005]. En effet, chaque membre du groupe s'associe aux autres et apporte ses connaissances et expertises pour être plus efficace et contribuer à un résultat collectif qui soit meilleur que l'adjonction des résultats individuels.
Ellis et al. [1991] ont proposé une décomposition de la collaboration avec le modèle 3C selon trois axes : *communication, coordination* et *coopération*. La communication est liée à l'échange de messages et d'informations entre les personnes; la coordination caractérise l'organisation des personnes, de leurs activités et des ressources partagées pour qu'elles travaillent ensemble efficacement; et la coopération est l'action de travailler en groupe sur un espace partagé et dans un objectif commun. La Figure 3.1 présente un exemple d'instanciation du modèle 3C dans le cas d'un travail en groupe [Fuks et al., 2005]. Cet exemple illustre comment les 3C combinées favorisent la conscience partagée (« *awareness* »), terme qui désigne la compréhension des activités des autres.
La prise de décision en groupe (Group Decision Making (GDM)) est une activité collaborative focalisée sur l'élaboration d'une décision collective [Harris, 1998].
Plusieurs approches dans la littérature décrivent la collaboration sans intégrer le volet GDM. Nous citons à titre d'exemples le méta-modèle CMSPEM [Kedji et al., 2012], l'ontologie de Santos et al. [2011] et le méta-modèle de Gallardo et al. [2013]. Dans la suite de ce chapitre, nous écartons cette catégorie d'approches puisqu'elles ciblent la modélisation des interactions et les enchaînements des activités des acteurs sans proposer de mécanismes ou de concepts pour la prise de décision collective.

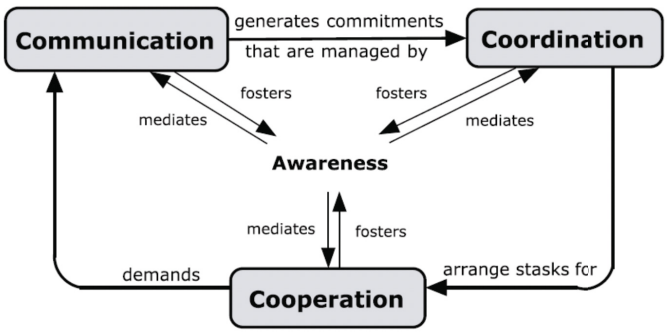


FIGURE 3.1 – Modèle 3C instancié pour un travail de groupe. [FUKS et al., 2005]

Dans la section 3.2, nous précisons le périmètre de l'étude de la littérature et les critères d'analyse que nous avons utilisés pour conduire notre étude. La section 3.3 décrit les approches modélisant la prise de décision en groupe recensées, la section 3.4 présente une synthèse de la comparaison de ces approches à base des critères susmentionnés, tandis que la section 3.5 clôt ce chapitre.

3.2 Critères d'analyse des approches de modélisation des processus GDM

3.2.1 Périmètre de l'étude de la littérature

FIGURE 3.2 – Schéma général d'un processus GDM (adapté de [CARRASCOSA, 2018; MORENTE-MOLINERA et al., 2020]).

Plusieurs travaux proposent des descriptions des processus GDM sous forme d'enchaînements d'étapes [ALDAG et FULLER, 1993; BELTON et PICTET, 1997; BAKER et al., 2002; CHAI et al., 2012]. Ces descriptions s'articulent autour des caractéristiques reprises dans la Figure 3.2. En effet, un processus GDM implique plusieurs **participants** qui expriment leurs *préférences* par rapport à des **propositions** selon un ensemble de **critères** [HARRIS, 1998; KILGOUR et EDEN, 2010]. Si le processus GDM inclut plusieurs critères de sélection entre les propositions, il s'agit d'un processus GDM multi-critères (Multi Criteria Group Decision Making (MCGDM))[MORENTE-MOLINERA et al., 2020]. L'objectif d'un processus GDM est d'obtenir une décision du groupe qui consiste à exploiter des **méthodes** de prise de décision pour classer les propositions selon les critères en prenant en considération les préférences des participants.
Dans l'exemple de la Figure 3.2, quatre participants sont impliqués dans la prise de décision en groupe, ils expriment leurs préférences par rapport aux quatre propositions A, B, C, et D. La Figure illustre comment les préférences peuvent être exprimées (i) à base de multi-critères (élaboration d'une matrice de préférences par proposition et par critère) et (ii) à base d'un critère unique.
Le *feature model* de la Figure 3.3 trace le périmètre de chacune des caractéristiques des processus GDM, à savoir : les participants, les propositions, les critères d'expression des préférences et les méthodes de prise de décision.

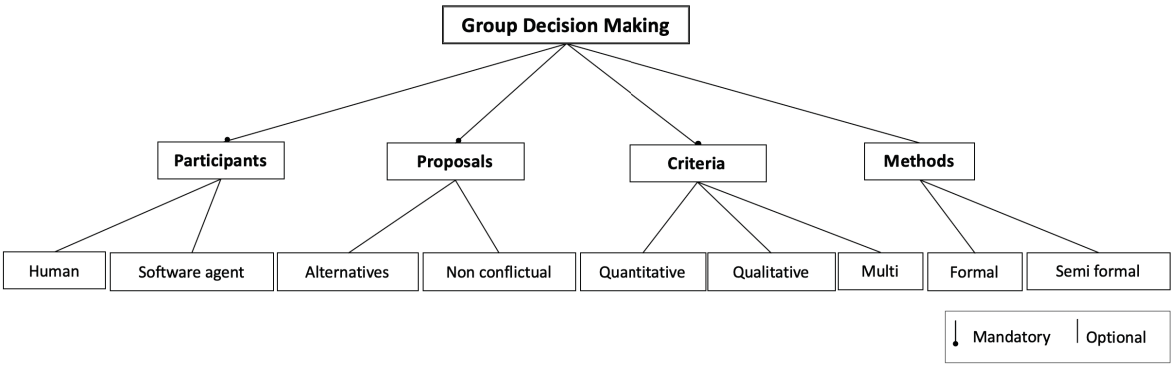


FIGURE 3.3 – Feature model décrivant les caractéristiques d'un processus GDM.

Participants. Ils représentent les *acteurs* impliqués dans la prise de décision, chacun possédant des compétences, une expérience et des connaissances particulières relatives à différents aspects du problème de GDM. Ils peuvent être assistés par des *agents logiciels*. Ces participants peuvent avoir plusieurs rôles dans le processus GDM : définition de l'objectif du GDM, définition des propositions, élaboration des critères de choix entre les propositions et expression des préférences individuelles sur les propositions.

Propositions. Elles désignent les solutions possibles fournies par les participants (en général) pour répondre aux objectifs fixés pour la prise de décision. Ces propositions peuvent être mutuellement exclusives en cas d'*alternatives* ou bien complémentaires en cas de propositions *non conflictuelles* [BELTON et PICTET, 1997].

Critères d'expression des préférences. Ils permettent d'argumenter les choix des participants. Ils doivent être utiles, indépendants et fiables pour le problème donné [BELTON et PICTET, 1997].

Dans des situations traditionnelles, les participants doivent classer un ensemble de propositions *qualitativement* en fonction de leurs perceptions.

Dans les méthodes de prise de décision en groupe multi-critères, le concept de « critère » est introduit. De cette façon, les participants sont invités à classer les propositions en fonction d'un ensemble de normes pré-établi.

L'introduction de critères dans les processus de prise de décision en groupe aide les participants à prendre la décision de manière moins subjective. En effet, ils doivent se concentrer sur la façon dont chaque proposition satisfait l'ensemble des critères requis au lieu d'utiliser leurs sentiments personnels. Par exemple, le choix d'un téléphone portable effectué avec une

méthode traditionnelle se basera sur la perception de l'acheteur, tandis qu'avec une méthode MCGDM, le choix se basera sur plusieurs critères dont la marque, le prix, les caractéristiques du matériel, la batterie, la caméra, etc. De cette manière, plusieurs participants peuvent évaluer avec l'acheteur les différents choix de portables qui s'offrent à lui.

Par la suite, nous nous référons aux critères d'expression des préférences par le terme « expression des préférences » pour éviter de les confondre avec les critères utilisés pour analyser les approches.

Méthodes de prise de décision. Elles aident à spécifier comment les préférences des différents participants sont prises en compte. Nous distinguons deux catégories de méthodes : méthodes *structurées* versus méthodes *non structurées* [Malavolta et al., 2014]. Dans le cas de méthodes structurées, les préférences individuelles sont agrégées comme dans Analytic Hierarchy Process (AHP) [Saaty, 1988, 2008] grâce à des calculs mathématiques qui sont parfois complexes. Dans le cas de méthodes moins structurées, comme le Brainstorming ou le vote, les règles d'unanimité, de majorité, de pluralité et de minorité sont utilisées.

Les méthodes de décision peuvent garantir que les décisions sont prises en temps opportun et que les préférences des participants sont prises en compte. Dans l'annexe A, nous décrivons en détail les méthodes de prise de décision en groupe les plus fréquemment utilisées.

3.2.2 Critères d'analyse des approches existantes

Pour définir les critères d'analyse des approches existantes, nous nous basons sur les critères d'analyse discutés dans les travaux de Saaty et al. [2006] et Rekha et Muccini [2014] et sur les quatre caractéristiques du GDM présentées ci-dessus : *Propositions*, *Participants*, *Expression des préférences* et *Méthodes*.

Propositions

Pour la caractéristique *Propositions*, les critères concernent l'*élaboration des propositions* et les *dépendances entre propositions*.

— L'élaboration des propositions concerne la manière dont l'approche établit l'ensemble des propositions. Elle est caractérisée par deux critères de plus bas niveau : identification et évolution.

  (i) *Identification des propositions* : ce critère indique si la définition des propositions fait partie intégrante du processus GDM ; dans le cas opposé, le processus GDM doit partir d'un ensemble prédéterminé de propositions.

  (ii) *Évolution des propositions* : ce critère indique si le processus GDM permet la considération de nouvelles propositions lors de sa mise en oeuvre. Les réflexions humaines peuvent s'enrichir itérativement durant le processus de prise de décision, notamment si l'ensemble de propositions de départ est incomplet.

— Les dépendances entre propositions concernent les solutions offertes pour exprimer les dépendances entre les propositions. Deux types de dépendances nous semblent à privilégier :

  (i) *Relation de spécialisation* puisqu'elle permet entre autres de gérer l'aspect évolutif des propositions au cours de l'exécution du processus GDM, en traçant les liens d'héritage entre propositions.

  (ii) *Relation de conflit* puisqu'elle permet de dégager les incompatibilités entre propositions, ce qui est utile pour le cas d'alternatives mutuellement exclusives.

Participants

Pour la caractéristique *Participants*, les critères concernent la *priorisation des participants*, i.e., la manière dont les participants sont gérés et priorisés. En effet, dans un processus GDM, les participants n'ont pas forcément le même niveau d'expertise ni les mêmes compétences. Ainsi, ce critère peut être évalué en vérifiant si l'approche permet une :

(i) *Sélection d'un sous-ensemble des participants* pour réduire le groupe des décideurs et n'impliquer que ceux ayant la compétence requise.

(ii) *Pondération des participants* en associant des poids différents à leurs préférences selon leur degré d'expertise.

Expression des préférences

Pour la caractéristique *Expression des préférences*, les critères concernent l'*inclusion de MCGDM* et la *pondération* en cas de critères multiples.
Le cœur de tout processus GDM est l'indication des préférences des participants. Il s'agit de spécifier la nature des critères de sélection entre les propositions supportées par l'approche. Ainsi, ce critère peut être évalué en vérifiant si l'approche permet une :

(i) *Sélection à base de critères multiples et mesurables* au lieu de considérer un seul critère, qui correspond généralement à l'avis du participant.

(ii) *Association de poids aux critères* (*critères pondérés*) : ces poids reflètent l'importance et la pertinence des critères.

Méthodes de GDM

Pour la caractéristique *Méthodes*, les critères concernent *l'adaptabilité des méthodes de GDM*.
Une méthode de décision est dite *adaptable* si l'approche permet de la personnaliser (e.g., configurer les seuils d'acceptation, offrir plusieurs techniques selon le contexte du GDM).

Outil support

Nous nous intéressons enfin à l'*existence d'un outil* et son *support* des autres caractéristiques afin d'évaluer sa *complétude*.

3.3 Approches de modélisation des processus GDM

La prise de décision en groupe concerne une vaste gamme de disciplines, y compris la gestion, l'économie, l'enseignement, l'ingénierie, et la médecine, pour n'en citer que quelques-unes. En raison de cette variété de domaines d'application, une multitude de modèles de prise de décision en groupe ont été définis. Dans cette thèse, nous considérons les modèles applicables au domaine de l'ingénierie des logiciels et systèmes en privilégiant les modèles génériques qui ne traitent pas d'un seul domaine métier, sauf si ce domaine est étroitement lié à notre thématique (développement de DSL, conception de système, etc.).
Les approches recensées sont la Decision Support Ontology (DSO) [Rockwell et al., 2009], MAke Decisions in Information Systems Engineering (MADISE) [Kornyshova et Deneckère, 2010; Kornyshova, 2011], l'Ontology based Group Decision Support System (OntoGDSS) [Chai et Liu, 2010; Chai, 2013], l'approche de Malavolta et al. [2014] et Collaboro [Izquierdo et Cabot, 2016]. Chaque approche est décrite par son objectif, ses concepts clés en regard des caractéristiques présentées dans la section 3.2.2 et ses limitations.

3.3.1 DSO

Objectif

Decision Support Ontology (DSO) [ROCKWELL et al., 2009] est une ontologie d'aide à la décision développée dans l'objectif de faciliter la prise de décision dans la conception collaborative et de fournir une taxonomie de concepts pour décrire les processus GDM. Bien que le modèle d'information mis au point soit relatif au domaine de la conception, DSO capture des informations de décision génériques et peut donc être adaptée pour être utilisée au-delà du domaine de la conception. DSO est basée sur le langage d'ontologie Web OWL [1], qui facilite le partage et l'intégration des informations de prise de décision entre plusieurs collaborateurs, via le Web.

Élaboration des propositions

DSO s'intéresse aux propositions de type alternatives qui sont établies par les membres du groupe durant les processus GDM. La Figure 3.4 reprend les concepts clés de DSO et les relations qui les lient. Les alternatives adressent un problème de GDM et sont mesurées par des informations d'évaluation recueillies à partir des connaissances préalables.

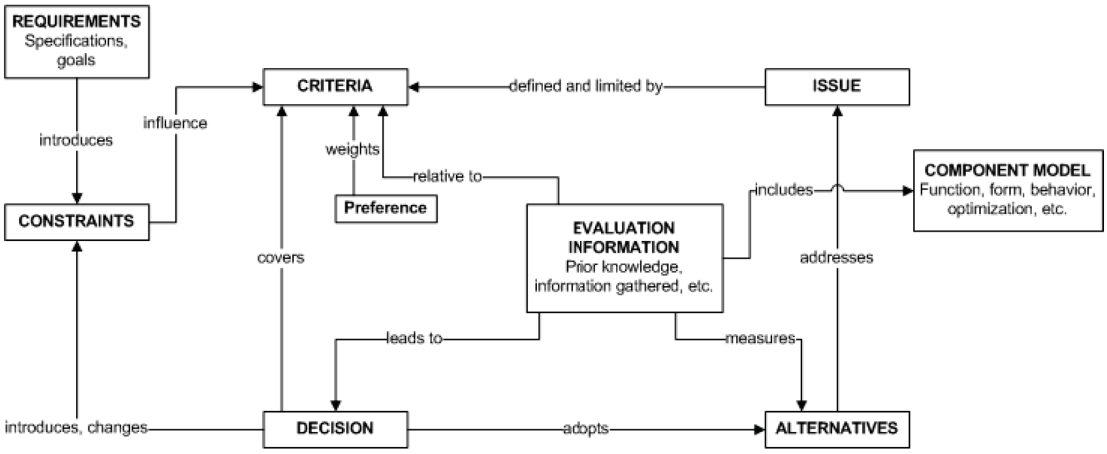


FIGURE 3.4 – Le modèle de décision de groupe dans DSO. [ROCKWELL et al., 2009]

Dépendances entre propositions

DSO ne détaille pas les types de relations pouvant exister entre les propositions. Mais comme elle intègre uniquement les alternatives mutuellement exclusives, une relation de conflit peut être identifiée entre les propositions.

Priorisation des participants

DSO n'intègre pas la caractéristique Participants d'un processus GDM.

1. OWL : Web Ontology Language est un langage de représentation des connaissances construit sur le modèle de données de RDF. Il fournit les moyens pour définir des ontologies web structurées. [MCGUINNESS et al., 2004]

Expression des préférences

Le problème de GDM est défini et limité par un ensemble de critères, tirés des informations d'évaluation. Ces critères sont influencés par un ensemble de contraintes.
Les participants expriment leurs préférences sur les alternatives par rapport à ces critères et la décision de groupe prend en compte l'ensemble de ces critères en essayant de trouver l'alternative qui satisfait le plus de critères.

Méthodes de GDM

DSO n'intègre pas de méthode de GDM. En effet, le modèle d'information mis au point est indépendante de toute méthode de décision.

Outil support

Le framework ne fournit pas d'outil support pour mettre en oeuvre la prise de décision.

Limitations

DSO permet de modéliser les concepts clés d'un processus de prise de décision en groupe. Cependant, DSO n'intègre pas de méthode de décision de groupe et ses auteurs ne proposent pas d'outil support.

3.3.2 MADISE

Objectif

L'objectif principal de l'approche MAke Decisions in Information Systems Engineering (MADISE) [KORNYSHOVA et DENECKÈRE, 2010; KORNYSHOVA, 2011] est de guider les ingénieurs informaticiens dans les activités de gestion des données. Elle comprend trois éléments : l'ontologie Decision Making Ontology (DMO), le processus MADISE, et le référentiel de méthodes de prise de décision. L'ontologie DMO est une représentation des concepts clés de la prise de décision en groupe. Le processus MADISE est un processus générique comprenant les principales activités utilisées pour la prise de décision et expliquant comment utiliser DMO. Le référentiel de méthodes fournit un ensemble de méthodes pour réaliser des activités de prise de décision en exploitant DMO.
Nous présentons DMO et le référentiel de méthodes puisque ce sont les deux éléments de l'approche MADISE qui concordent avec nos critères.

Élaboration des propositions

Le diagramme de classe UML de la Figure 3.5 modélise DMO en représentant les concepts fondamentaux de la prise de décision, les relations qui les lient ainsi que leurs attributs.
La racine de l'ontologie DMO est la situation de prise de décision (*DMSituation*) qui regroupe les conditions spécifiques traitant un objet de prise de décision donné. L'objet de prise de décision (*DMObject*) représente l'artefact sujet de la prise de décision ; il peut être un processus (*ProcessElement*) ou un produit (*ProductElement*).
Une situation donnée contient un problème, un ensemble de solutions alternatives et un ensemble de critères.
AlternativeSet est un ensemble d'alternatives valables dans une situation donnée ; il est composé d'au moins deux alternatives et les auteurs distinguent l'ensemble stable de l'ensemble évolutif grâce à l'attribut *nature* du concept *AlternativeSet*. La *nature* est stable si on conserve les mêmes alternatives pendant une durée assez longue tandis qu'elle est évolutive si des alternatives sont supprimées et d'autres ajoutées assez fréquemment. Les alternatives ne sont pas forcément produites par les acteurs participant au processus GDM, les décideurs pouvant partir d'un ensemble d'alternatives initial.

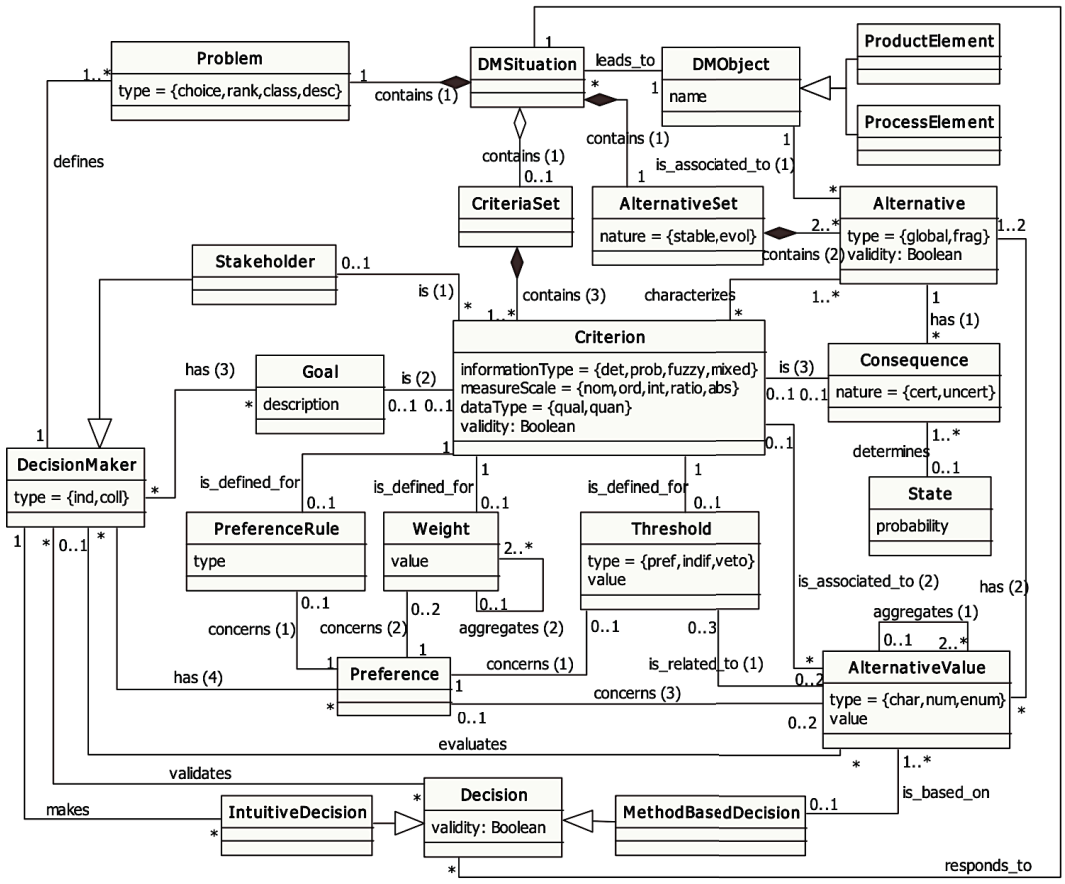


FIGURE 3.5 – Decision Making Ontology (DMO) de l'approche MADISE. [KORNYSHOVA et DENECKÈRE, 2010]

Dépendances entre propositions

La relation de conflit entre propositions est intégrée à DMO grâce à l'attribut *type* du concept *Alternative*. Si le *type* d'une proposition est *frag* (*fragmented*) alors cette alternative peut être appliquée avec d'autres ; sinon (i.e., *type* = *global*) l'application de l'alternative se fait en excluant les autres.

Priorisation des participants

DMO permet de prendre en compte la disparité des compétences dans le groupe de décideurs ; à cet effet, elle propose de considérer les participants comme critère de sélection entre les propositions, en leur attribuant des poids selon leur niveau d'expertise dans le sujet de la prise de décision.

Expression des préférences

La situation de prise de décision peut également contenir un ensemble de critères qui permettent aux participants d'exprimer leurs préférences. Un critère représente des informations de tout type permettant d'évaluer des alternatives et de les comparer. Les critères sont décrits par un nom et les attributs suivants :

— *informationType* : Le type d'information décrivant le type de critère : déterministe (*det*), probabiliste (*prob*), flou ou mixte ;

— *measureScale* : L'échelle de mesure indiquant le type d'échelle utilisé pour mesurer les alternatives en fonction du critère donné : nominal (*nom*), ordinal (*ord*), intervalle (*int*), ratio et absolu (*abs*) ;

— *dataType* : Le type de données qui peut être qualitatif (*qual*) ou quantitatif (*quan*) ;

— *validity* : La validité indique si le critère initialement identifié est sélectionné pour évaluer les alternatives.

Les critères peuvent avoir différentes origines : les caractéristiques intrinsèques des alternatives, les conséquences futures des alternatives, les objectifs et même les parties prenantes.

Méthodes de GDM

Dans DMO, nous trouvons deux types de décisions : les décisions intuitives (*IntuitiveDecision*) et les décisions à base de méthodes (*MethodBasedDecision*).
Les décisions intuitives sont prises à la volée sans utilisation de méthodes contrairement aux décisions à base de méthodes qui permettent généralement de définir une liste de critères, d'évaluer les propositions selon ces critères et d'agréger les évaluations.
Le type de décision dépend de la situation de prise de décision. Il peut consister à : sélectionner une alternative (sel_alt), sélectionner plusieurs alternatives (sel_alts), classer des alternatives ($rank_alt$), trier des alternatives ($sort_alt$) et décrire des alternatives ($desc_alt$). Il peut également être *NULL* si aucune décision n'est prise (dans le cas de non aboutissement de la prise de décision ou si la situation de prise de décision a juste pour objectif de collecter les alternatives par exemple).

Outil support

Le référentiel des méthodes MADISE a pour objectif de fournir le support méthodologique nécessaire à la réalisation des activités de prise de décision. En effet, il permet de stocker les méthodes de prise de décision et de prendre en charge le processus de configuration de ces méthodes.
Le référentiel permet de composer les méthodes et de les réutiliser dans la construction d'autres méthodes. Ainsi, les méthodes deviennent des modules pour la construction de nouvelles méthodes et peuvent également être utilisées pour enrichir celles déjà existantes.

Limitations

L'approche MADISE et plus spécifiquement son ontologie DMO n'offre pas de mécanisme pour organiser les dépendances entre propositions (à part la relation de conflit).
En ce qui concerne la priorisation des participants, l'approche permet de considérer les perceptions des participants de façon non équitable, mais n'intègre pas de mécanisme pour restreindre le groupe de décideurs une fois la situation de prise de décision définie (i.e., il n'est pas possible de sélectionner un sous-ensemble de décideurs par groupe d'alternatives).

3.3.3 OntoGDSS

Objectif

Ontology based Group Decision Support System (OntoGDSS) [Chai et Liu, 2010; Chai, 2013] est un framework de système d'aide à la prise de décision en groupe basé sur une ontologie sur le Web. Les auteurs définissent un modèle d'argumentation de groupe multicouche, comme illustré sur la Figure 3.6. Ce framework a pour objectif de faciliter la gestion du processus de décision dans son ensemble. Il intègre la construction du groupe des décideurs, l'élaboration des alternatives et de leurs évaluations ainsi que l'agrégation de ces évaluations.

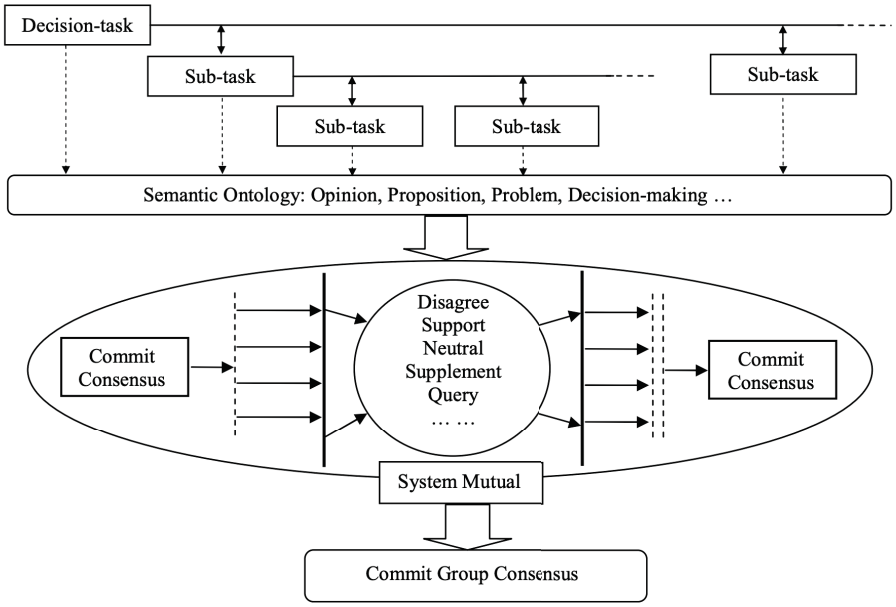


FIGURE 3.6 – Modèle d'argumentation multicouche dans OntoGDSS. [CHAI, 2013]

Élaboration des propositions

Le modèle d'argumentation de groupe proposé permet d'abord de construire la structure du groupe et la décomposition des tâches. Ensuite, les auteurs présentent une approche basée sur une ontologie pour représenter un problème de décision et extraire les concepts sémantiques sous forme d'opinions, de propositions, de problèmes, etc.

Dépendances entre propositions

Le framework s'intéresse aux alternatives mutuellement exclusives qui sont établies par les membres du groupe durant le processus de GDM et non pas prédéterminées. Étant donné que ces alternatives sont des propositions mutuellement exclusives, on peut considérer que le framework intègre la relation de conflit.

Priorisation des participants

Le framework n'inclut pas de concept répondant à cet aspect.

Expression des préférences

Le framework n'inclut pas de concept répondant à cet aspect.

Méthodes de GDM

Le framework vise un consensus pour valider les propositions. Son modèle d'argumentation de groupe propose cinq types de décisions de base, appelées *relations* dans l'approche, pour exprimer les préférences des décideurs par rapport aux propositions. Ces relations sont le *désaccord*, l'*approbation*, l'*indifférence*, le *besoin de supplément* et les *requêtes*. Ces méthodes sont non structurées pour évaluer les propositions et ne permettent pas de formaliser les évaluations des acteurs.

Outil support

Comme DSO, OntoGDSS n'est pas supportée par un outil et ne permet pas par exemple de pouvoir tirer profit de la pondération des critères proposée.

Limitations

Ce framework permet de modéliser un processus de prise de décision en groupe dans son ensemble. Son ontologie offre un vocabulaire pour les concepts clés de la prise de décision en groupe. Cependant, il ne permet pas d'exécuter la prise de décision en elle-même.
Par ailleurs, à notre connaissance, les relations entre les concepts de l'ontologie ne sont pas développées, ce qui laisse une ambiguïté pour certains concepts. Notons aussi que la taxonomie proposée n'intègre pas tous les volets du GDM, comme l'expression des préférences et la priorisation des participants par exemple et même en ce qui concerne les méthodes de GDM, elles sont non structurées et ne permettent pas de formaliser les évaluations des propositions.

3.3.4 Approche de Malavolta

Objectif

MALAVOLTA et al. [2014] proposent une approche pour incorporer explicitement les stratégies de GDM dans les décisions d'évolution d'architecture logicielle. Pour cela, les auteurs proposent un méta-modèle de référence décrivant les éléments de raisonnement minimaux nécessaires pour mettre en œuvre la prise de décision en groupe dans le cadre d'un processus de conception logicielle.

Élaboration des propositions

Le méta-modèle de Malavolta, illustré sur la Figure 3.7, inclut les principes de GDM lors d'un processus de conception logicielle. *DesignDecision* représente une proposition et contient des attributs tels que la description, l'état qui représente l'état actuel de la décision (par exemple, *idée, provisoire, décidé, rejeté, etc.*), un horodatage qui spécifie le moment où la décision de conception a été créée, l'historique de l'évolution de la décision (principalement, l'auteur, l'horodatage et le statut de chaque version antérieure de la décision).

Dépendances entre propositions

Chaque proposition, appelée dans l'approche *décision de conception*, peut être liée à d'autres décisions de conception au moyen de relations typées, telles que les compositions, les contraintes, les activations, les dépendances, les relations de conflit, etc.

Priorisation des participants

Les *GroupMemberships* mettent en relation chaque groupe avec ses parties prenantes. Le méta-modèle inclut un concept *ranking* au sein de la méta-classe *GroupMembership* qui permet de prioriser les membres du groupe et leurs préférences lors de l'élaboration de la décision du groupe. Ce concept est optionnel étant donné que certaines organisations peuvent avoir des structures plates où toutes les parties prenantes bénéficient d'une priorité égale.

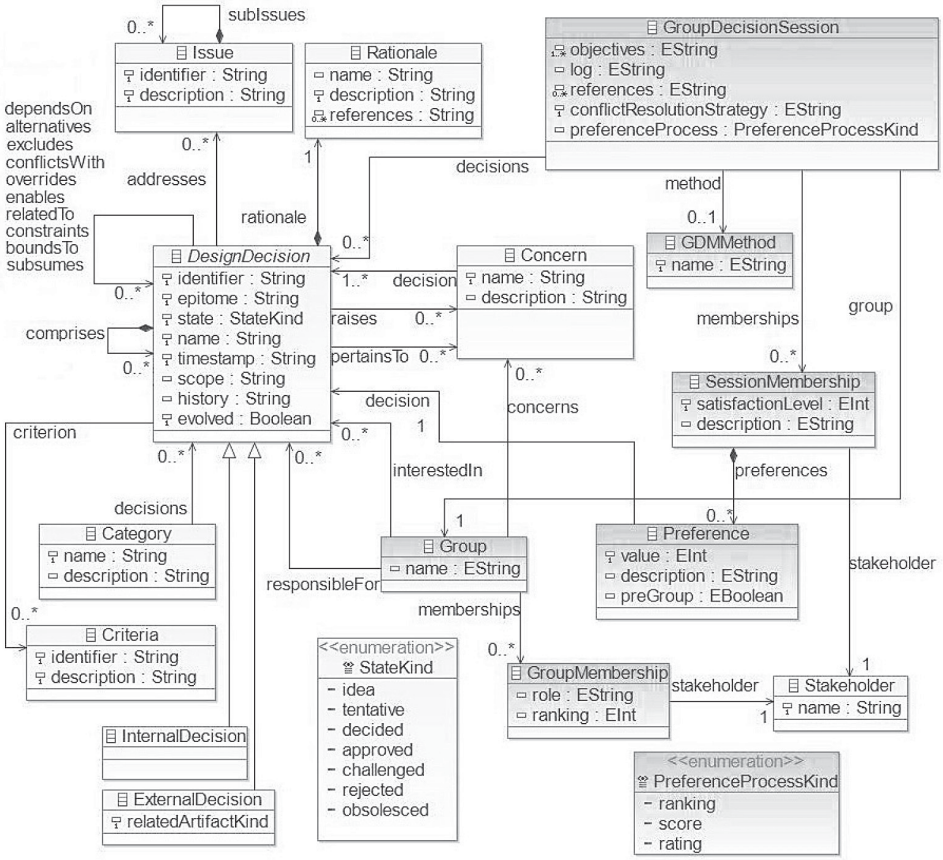


FIGURE 3.7 – Méta-modèle générique de prise de décision de MALAVOLTA et al. [2014].

Expression des préférences

Le concept session de décision de groupe *GroupDecisionSession* englobe des décisions relatives aux *DesignDecision*. Il possède un attribut *preferenceProcess* décrivant le processus utilisé pour enregistrer les préférences de chaque partie prenante. Les processus de préférence peuvent être basés sur un classement, une notation ou un système de notation.
Ce méta-modèle inclut aussi le concept *Criteria* qui permet d'exprimer les critères utilisés pour argumenter les *DesignDecision*. Les concepteurs doivent associer un motif (*rationale*) à chaque décision de conception afin de garder une trace des justifications des décisions.
Une préférence est composée de sa valeur cardinale (afin qu'elle puisse être facilement comparée aux préférences des autres parties prenantes au sein du groupe) et de sa description.
Le méta-modèle incorpore un concept pour gérer le changement des préférences des parties prenantes au cours du processus GDM, puisque les parties prenantes peuvent revoir leurs préférences au fur et à mesure de l'échange d'informations et des discussions. L'attribut booléen *preGroup* est mis à vrai lorsque la préférence a été exprimée avant la session de décision du groupe.

Méthodes de GDM

Le méta-modèle inclut aussi un concept *GDMMethod* qui représente la méthode suivie pour évaluer les diverses décisions de conception. Certaines méthodes populaires peuvent être utilisées comme le brainstorming, le vote, la technique de groupe nominal (NGT), la technique Delphi, l'analyse hiérarchique des procédés (AHP). Ces méthodes sont présentées dans l'annexe A.

Outil support

Les auteurs proposent un outil pour supporter leur approche. Cet outil a été implémenté comme un ensemble de plugins Eclipse dont deux composants concernent la mise en oeuvre des concepts du méta-modèle.
Le composant *GDM Model Editor* fournit toutes les capacités de modélisation aux parties prenantes, afin qu'elles puissent définir graphiquement les modèles GDM. Il fournit des fonctionnalités telles que la création d'éléments, la gestion de leurs propriétés, le copier-coller, l'exportation vers un fichier image, etc.
Le composant *GDM Methods Manager* fournit un certain degré d'automatisation au processus GDM. Plus précisément, il permet aux parties prenantes de calculer automatiquement le classement de toutes les décisions de conception impliquées dans une session de décision de groupe, en fonction de la méthode GDM spécifiée. Ainsi, par exemple, si la session de décision de groupe applique AHP comme méthode de GDM, le gestionnaire de méthodes GDM exécute la comparaison et le classement par paire des alternatives et calcule la meilleure solution au moment de l'exécution.
Ce composant contient un ensemble de classes Java génériques pour mettre à jour le modèle GDM au moment de l'exécution ; il est complété par un ensemble de classes Java spécifiques mettant en œuvre chacune une méthode de GDM.

Limitations

Ce méta-modèle inclut le concept *Criteria* qui permet d'exprimer les critères utilisés pour argumenter les *DesignDecision*. Cependant, aucun élément du méta-modèle ne permet de déduire que ces critères sont mesurables.
Le méta-modèle n'incorpore pas de méthodes de prise de décision précises. Ainsi pour utiliser une méthode donnée, avant la mise en oeuvre du processus GDM, une classe Java implémentant la méthode doit être ajoutée à l'outil proposé par les auteurs.

3.3.5 Collaboro

Objectif

Le méta-modèle Collaboro [IZQUIERDO et CABOT, 2016] a pour but de décrire les concepts et relations permettant de représenter l'élaboration collaborative d'un DSL.

Élaboration des propositions

La Figure 3.8 résume les principaux concepts de ce méta-modèle. Collaboro permet à chaque membre du groupe d'initier des sessions collaboratives. Pour cela, chaque membre peut élaborer des *propositions*. Les propositions traitant un même problème sont réunies dans des *solutions*. Les membres donnent leur opinion (*commentaires*) au sujet des solutions proposées.
Les *propositions*, les *solutions* ainsi que les *commentaires* sont soumis aux *votes* des autres membres.

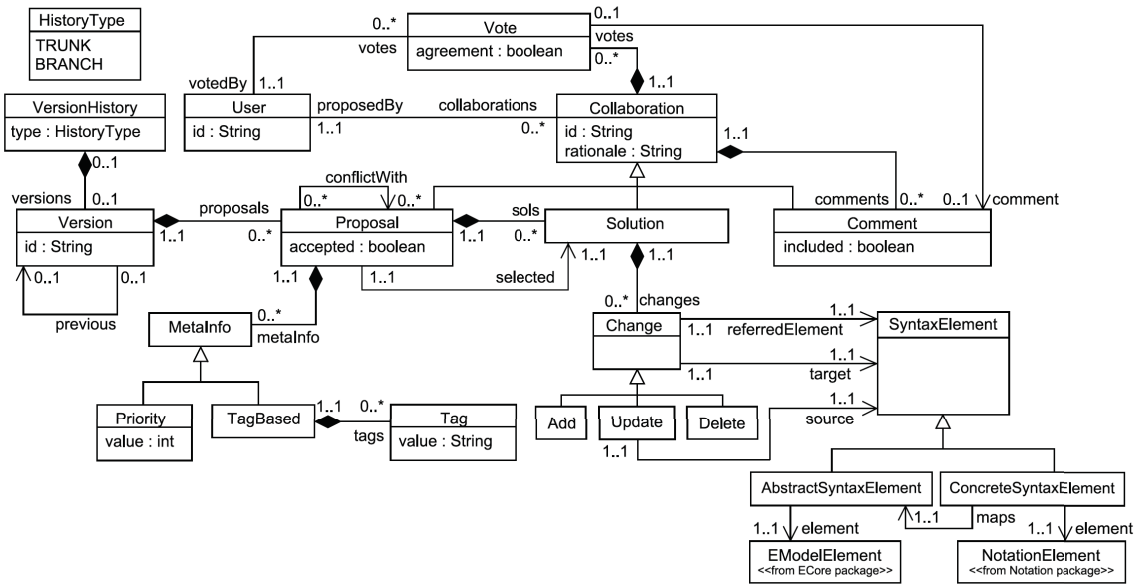


FIGURE 3.8 – Éléments clés du méta-modèle Collaboro. [IZQUIERDO et CABOT, 2016]

Dépendances entre propositions

Les propositions conflictuelles sont tracées grâce à la relation *conflictWith* qui relie une proposition à celles qui lui sont conflictuelles.

Priorisation des participants

Cette caractéristique n'est pas prise en compte dans Collaboro. En effet, les participants sont considérés d'égale importance et ont ainsi le même poids décisionnel.

Expression des préférences

Les critères de sélection entre propositions sont des critères qualitatifs basés sur les opinions et perceptions des décideurs.

Méthodes de GDM

L'approche Collaboro propose un moteur pour la prise de décision. Le rôle de ce moteur est double : d'une part, il permet aux participants de définir la stratégie de prise de décision (par exemple, comment voter, nombre de votes nécessaires pour arriver à un accord, etc.) ; d'autre part, il permet de filtrer les solutions selon le nombre de votes, et d'incorporer les solutions acceptées dans le répertoire des collaborations, qui garde trace des versions des propositions (le membre l'ayant proposée, les membres ayant participé au vote).
Le moteur de décision suit un processus semi-automatisé. Cependant, les auteurs de l'approche proposent d'intégrer à la prise de décision un coordinateur dont le rôle est d'établir un consensus entre les membres en cas de divergence des opinions individuelles.

Outil support

L'outil proposé par Collaboro supporte à la fois l'élaboration des propositions, l'expression des préférences individuelles et leur agrégation grâce au moteur de décision.

Limitations

Collaboro met en œuvre une formalisation de la prise de décision en groupe. Toutefois, le méta-modèle ne permet pas de répertorier les liens entre les propositions. Par exemple si une proposition a été raffinée après un refus, cette information est perdue et il n'est pas possible non plus de tracer les dépendances entre propositions au sein d'une même session collaborative.
Le méta-modèle favorise une prise de décision par consensus. Cependant, le fait que ses principaux concepts (proposition, solution, commentaire) soient soumis au vote rend la collaboration fastidieuse surtout dans le cas de groupes de grande taille.

3.4 Comparaison des approches

Le tableau 3.1 propose une comparaison des approches présentées dans la section précédente, à savoir : la Decision Support Ontology (DSO) [ROCKWELL et al., 2009], MAke Decisions in Information Systems Engineering (MADISE) [KORNYSHOVA et DENECKÈRE, 2010; KORNYSHOVA, 2011], l'Ontology based Group Decision Support System (OntoGDSS) [CHAI et LIU, 2010; CHAI, 2013], l'approche de MALAVOLTA et al. [2014] et Collaboro [IZQUIERDO et CABOT, 2016].
La comparaison de ces approches est basée sur les critères présentés dans la section 3.2.2 et qui sont pour rappel : *élaboration des propositions*, *dépendances entre propositions*, *priorisation des participants*, *expression des préférences*, *méthodes de décision de groupe* et *outil support*.

Élaboration des propositions. En terme d'identification des propositions, les cinq approches étudiées permettent aux membres du groupe d'identifier l'ensemble des propositions de départ. Cela les implique davantage que lorsque les parties prenantes reçoivent une liste prédéterminée de propositions (par la personne qui a initié la collaboration par exemple).

Pour l'évolution des propositions, deux approches sur cinq - Collaboro et MADISE - permettent de faire évoluer les propositions au cours du processus GDM. Ceci est possible dans la première approche puisque tous ses concepts (proposition, solution et commentaire) sont soumis au vote, alors que dans la deuxième approche, l'attribut nature du concept *Alternative* de l'ontologie DMO permet de spécifier les alternatives qui ont évolué.

Pour les autres approches - DSO, OntoGDSS et Malavolta - il n'est pas possible de faire évoluer les propositions au cours du processus GDM.

Dépendances entre propositions. Les approches étudiées permettent d'identifier les relations de conflit entre propositions soit parce qu'elles traitent uniquement des alternatives mutuellement exclusives (DSO, MADISE et OntoGDSS), soit parce qu'elles incluent des concepts ou des relations spécifiques (approche de Malavolta et Collaboro).

En ce qui concerne la relation de spécialisation, seul le méta-modèle de MALAVOLTA et al. [2014] permet d'identifier ce type de relation. Or les dépendances entre propositions offrent aux membres une vision globale et claire de l'ensemble des propositions et limitent ainsi les risques de décisions prématurées [EKLUND et ARTS, 2010].

Priorisation des participants. Chaque participant doit être traité en fonction de son expertise. MADISE et l'approche de Malavolta reconnaissent cet aspect et proposent des mécanismes pour pondérer les préférences des participants en fonction de leur expertise. Ces mécanismes sont optionnels pour que les approches puissent considérer aussi les groupes de participants ayant la même expertise. Cependant, aucune approche n'inclut une fonctionnalité permettant de restreindre le groupe des participants en fonction des propositions. Ceci pourrait être utile quand une session de GDM comporte plusieurs propositions mais que certaines de ces propositions ne concernent pas forcément tous les participants à la prise de décision.

Élaboration des propositions	***DSO***	***MADISE***	***OntoGDSS***	***Appr. Malavolta***	***Collaboro***
Identification des propositions	●	●	●	●	●
Evolution des propositions	-	●	-	◐	●
Dépendances entre propositions					
Relation de conflit	●	●	●	●	●
Relation de spécialisation	-	-	-	●	-
Priorisation des participants					
Sélection de participants	-	-	-	-	-
Pondération de participants	-	●	-	●	-
Expression des préférences					
Critères mesurables	●	●	-	-	-
Critères pondérables	●	●	-	◐	-
Méthode décision du groupe					
Adaptabilité	-	●	-	-	◐
Outil support					
Complétude	-	●	-	◐	●

● : Critère au centre de l'approche, ◐ : Partiellement considéré, - : Non considéré

TABLEAU 3.1 – Synthèse des caractéristiques supportées par les approches de modélisation de GDM.

La restriction du groupe de décideurs pourrait faire gagner du temps, fluidifier le processus GDM et assurer même la protection et la confidentialité des données.

Expression des préférences. Les parties prenantes participant au GDM doivent pouvoir indiquer leurs préférences.

Dans Collaboro, les alternatives sont évaluées grâce aux votes des parties prenantes dans le but de parvenir à un consensus. Les décideurs évaluent les propositions en s'appuyant sur des critères intrinsèques (préférences, perceptions personnelles).

MADISE répertorie un ensemble de critères en les subdivisant en critères subjectifs comme les critères intrinsèques aux décideurs, et en d'autres critères objectifs et quantifiables.

Les autres approches - DSO, OntoGDSS et Malavolta - ne clarifient pas comment les propositions sont évaluées. Elles ne semblent pas inclure de critères quantifiables puisque leurs modèles n'apportent pas de précision là dessus.

DMO (de l'approche MADISE) et DSO proposent la pondération des critères, en attribuant une valeur à chaque critère. Tandis que le méta-modèle de Malavolta permet d'attribuer des poids décisionnels distincts aux participants selon leur expertise.

Méthodes de GDM. DSO et le méta-modèle de Malavolta ont été conçus indépendamment de toute méthode de décision. Ils définissent un concept générique pour la méthode d'agrégation mais ne proposent pas de méthodes concrètes. Ainsi, les prises de décision en groupe ne peuvent pas être déroulées avec ces approches telles qu'elles sont.

Dans Collaboro et OntoGDSS, les scénarios alternatifs sont notés et les parties prenantes votent les propositions. La méthode de vote a été choisie pour permettre aux groupes de parvenir à un consensus en temps opportun. Toutefois, dans Collaboro, le rôle modérateur peut intervenir pour conclure le processus même si le consensus n'est pas atteint.

MADISE offre une liste de méthodes d'agrégation utilisables dans les processus GDM. Ces méthodes sont implémentées dans le référentiel de méthodes et sont donc directement exploitables pour exécuter les processus GDM.

Outil support. Les ontologies OntoGDSS et DSO n'offrent pas d'outil support pour la mise en oeuvre des processus GDM car elles sont focalisées sur la construction de la taxonomie nécessaire lors d'un processus GDM.

L'outil proposé par MALAVOLTA et al. [2014] n'intègre pas en lui-même les méthodes de décision de groupe. Il doit être complété en définissant des classes Java implémentant les méthodes de GDM que l'on veut exploiter lors de l'exécution d'un processus GDM.

L'outil proposé par Collaboro supporte à la fois l'élaboration des propositions, l'expression des préférences individuelles et leur agrégation grâce au moteur de décision.

MADISE fournit un référentiel de méthodes de prise de décision, ce qui lui permet de compléter son ontologie DMO et de pouvoir automatiser des processus de prise de décision en groupe.

3.5 Conclusion

Dans ce chapitre, nous avons étudié les approches de modélisation de GDM. Nous avons d'abord défini le périmètre de l'étude. Ensuite, nous avons élaboré une liste de critères d'analyse des approches en nous basant à la fois sur les éléments clés d'un processus GDM et les critères fréquemment utilisés dans la littérature.
L'analyse a porté sur cinq approches qui peuvent être considérées comme complémentaires pour modéliser la prise de décision en groupe du fait qu'aucune d'entre elles ne couvre la totalité des éléments d'un processus GDM. Ceci est dû principalement au fait qu'elles ont été conçues pour des besoins spécifiques et répondent ainsi à certains critères et pas à d'autres.
La limitation majeure de ces approches concerne le support de l'évolution des propositions au cours d'un processus GDM. Nous considérons que ce point est important, pour pouvoir enrichir

l'ensemble des propositions pendant le déroulement du processus GDM, en fonction de l'avancement des réflexions du groupe et ce sans devoir relancer, depuis le début, la session de prise de décision.
D'un autre côté, certaines approches ne permettent pas de mettre en oeuvre les processus GDM. Ceci vient principalement du fait qu'elles n'intègrent pas de méthodes concrètes de prise de décision en groupe dans leurs modèles ou dans les outils qu'elles fournissent. Nous pensons que des contributions doivent être fournies dans cette direction, et qu'il est utile de proposer des outils pour la prise de décision en groupe génériques et configurables selon le contexte et l'objectif de la prise de décision. Ces outils doivent permettre, par exemple, de choisir la nature des critères utilisés pour exprimer les préférences, de choisir aussi entre des méthodes multi-critères et des méthodes à critère unique et de choisir les règles de GDM à appliquer (vote, AHP, etc.).

L'étude de ces approches nous a permis de définir les briques essentielles pour le méta-modèle de prise de décision qui fait l'objet du chapitre suivant (chapitre 4).

Conclusion

Dans cette première partie, nous avons réalisé une étude bibliographique concernant les deux thématiques complémentaires de notre problématique de recherche : (i) l'alignement de modèles et la gestion de la cohérence lors d'une conception multi-vue et (ii) la description et la modélisation d'un processus de prise de décision en groupe.

Dans le chapitre 2, nous avons analysé les travaux de la littérature relatifs à la gestion de la cohérence lors d'une conception multi-vue, favorisant la séparation des préoccupations métier.
Les approches d'alignement étudiées présentent des limitations concernant deux aspects : (i) le support à la collaboration lors de l'alignement, et (ii) la réutilisabilité de cet alignement lors des évolutions.
En ce qui concerne le premier aspect, nous constatons que la majorité des approches (mis à part les approches Shosha et CIMA) ne supportent pas la collaboration lors de la définition des correspondances. En effet, elles supposent qu'un expert peut - à lui seul - exécuter les activités non automatisables lors de l'alignement. D'une part, cette supposition n'est pas réaliste car dans les systèmes complexes la connaissance est répartie sur des équipes différentes. D'autre part, même si les approches stipulent que cet expert pourrait faire appel aux concepteurs métier pour l'assister en cas de besoin, ceci reste non formalisé et risque d'être fastidieux puisque les outils fournis ne supportent pas la collaboration et les interactions des différentes parties prenantes.
Concernant le deuxième aspect, l'évolution des modèles est un caractère intrinsèque à l'ingénierie logicielle collaborative dirigée par les modèles (*Collaborative MDSE*), et la modification d'un modèle peut entraîner l'incohérence de l'ensemble. Répercuter les modifications, ou tout au moins identifier les éléments de modèles qui seront impactés par les changements s'avère donc nécessaire.

L'étude bibliographique du chapitre 3 a porté sur les approches de modélisation de la prise de décision en groupe (GDM) et les éléments clés du processus GDM qu'elles incluent.
Les approches étudiées ne couvrent pas toutes les caractéristiques d'un processus GDM, surtout en termes de considération de nouvelles propositions durant ce processus (évolution des propositions) et d'existence d'un outil support pour le déroulement de la prise de décision en groupe. Ainsi, ces approches ne répondent pas complètement à notre besoin. C'est pour cela que la première contribution de notre travail (chapitre 4) consiste en un méta-modèle formalisant les processus GDM.

Deuxième partie

Contributions Conceptuelles

Introduction 55

4 Formalisation de la prise de décision en groupe : MMCollab 57

5 Alignement collaboratif de modèles hétérogènes : CAHM 93

Conclusion 123

Introduction

Cette partie rassemble le coeur de la thèse. Elle est structurée en deux chapitres. Nous proposons dans le **chapitre 4 un méta-modèle**, appelé **MMCollab, décrivant la prise de décision collaborative** et, sur la base de ce méta-modèle, nous définissons un ensemble de **politiques de prise de décision configurables et réutilisables** qui permettent de mettre en oeuvre des processus de GDM.

Dans le **chapitre 5**, intitulé **Alignement collaboratif de modèles hétérogènes : CAHM**, nous proposons un **processus collaboratif d'alignement de modèles hétérogènes** dans le cadre d'une conception multi-vue. Ce processus précise comment et quand les concepteurs métier interagissent pour créer une vue globale du système considéré, qui prend en compte des préoccupations métier diversifiées tout en favorisant la séparation de ces préoccupations.
Ce processus s'appuie sur MMCollab ce qui lui permet de gérer collaborativement les décisions à prendre lors de la définition et de la maintenance des correspondances inter-modèles.

Chapitre 4

Formalisation de la prise de décision en groupe : MMCollab

Sommaire

4.1 Introduction . . . 58
4.2 Principe de construction du méta-modèle MMCollab . . . 58
4.3 Méta-modèle de collaboration (MMCollab) . . . 59
4.3.1 Structuration en paquetages de MMCollab . . . 59
4.3.2 Paquetage Core Concepts . . . 61
4.3.2.1 Collaboration . . . 61
4.3.2.2 Proposal . . . 62
4.3.2.3 CollaborativeWorkProduct . . . 63
4.3.3 Paquetage Actors . . . 64
4.3.3.1 User . . . 65
4.3.3.2 InvolvedUser . . . 65
4.3.4 Paquetage Proposals . . . 66
4.3.4.1 CompositeProposal . . . 66
4.3.4.2 ElementaryProposal . . . 67
4.3.4.3 AlternativeProposal . . . 68
4.3.5 Paquetage Collective Decision . . . 68
4.3.5.1 Pattern . . . 68
4.3.5.2 GDMPattern . . . 70
4.3.5.3 CoDecisionMethod . . . 70
4.3.5.4 ParticipationMethod . . . 71
4.3.6 Paquetage Evaluation . . . 73
4.3.6.1 Decision . . . 73
4.3.6.2 Comment . . . 74
4.3.7 Syntaxe concrète du méta-modèle MMCollab . . . 75
4.4 Instanciation de MMCollab . . . 75
4.4.1 Exemple de situation de collaboration : Définir des Règles de Gestion (RGs) 76
4.4.2 DecisionPolicy : Instance de CollectiveDecision : :GDMPattern . . . 76
4.4.3 Politique de décision « Taking Advice » . . . 78
4.4.3.1 Description de « Taking Advice » . . . 78
4.4.3.2 Application de « Taking Advice » à l'exemple « Définir des Règles de Gestion » . . . 80
4.4.4 Politique de décision « Majority Deciding » . . . 80
4.4.4.1 Description de « Majority Deciding » . . . 80
4.4.4.2 Application de « Majority Deciding » à l'exemple « Définir des Règles de Gestion » . . . 82

4.4.5 Politique de décision « Consenting Together » 83
4.4.5.1 Description de « Consenting Together » 83
4.4.5.2 Application de « Consenting Together » à l'exemple « Définir des Règles de Gestion » 84
4.4.6 Politique de décision « Negotiating together » 84
4.4.6.1 Description de « Negotiating together » 84
4.4.6.2 Application de « Negotiating together » à l'exemple « Définir des Règles de Gestion » 87
4.4.7 Politique de décision « Delegating » 88
4.4.7.1 Description de « Delegating » 88
4.4.7.2 Application de « Delegating » à l'exemple « Définir des Règles de Gestion » 89
4.5 Conclusion 91

4.1 Introduction

Dans ce chapitre, nous proposons une formalisation de la prise de décision en groupe. Pour cela, nous proposons un langage de description des processus de prise de décision en groupe spécifié sous la forme d'un méta-modèle appelé MMCollab. Nous commençons par expliquer le principe de la construction du méta-modèle (section 4.2), puis nous décrivons ses concepts dans la section 4.3. Nous détaillons les politiques de décision, instances du concept *GDMPattern* dans la section 4.4 en introduisant un exemple de collaboration. Dans la section 4.5, nous concluons le chapitre en positionnant notre contribution par rapport aux travaux analysés dans le chapitre 3.

4.2 Principe de construction du méta-modèle MMCollab

Notre intention première dans cette thèse est de fournir une approche d'alignement collaboratif de modèles hétérogènes, élaborés dans le cadre d'une conception multi-vue. Ceci passe par la mise en œuvre de mécanismes de gestion de la prise de décision en groupe car le processus d'alignement requiert l'élaboration de décisions collectives (concernant les correspondances à mettre en place par exemple). Ainsi, nous avons décidé de construire un méta-modèle décrivant les concepts fondamentaux de la prise de décision en groupe.
Ce méta-modèle de collaboration, appelé MMCollab, est inspiré de Collaboro (voir section 3.3.5) dans la définition des propositions d'une collaboration. Cependant, MMCollab est indépendant de tout domaine d'application et peut être utilisé dans divers contextes de prise de décision en groupe. Deux concepts de MMCollab étendent des concepts provenant des paquetages *Process Structure* de SPEM [1] [Object Management Group (OMG), 2008] et CMSPEM [Kedji et al., 2012]. Le choix de SPEM et son extension CMSPEM est justifié par le fait que SPEM est un standard orienté description de processus logiciels (contrairement à BPMN [Silver, 2009] par exemple qui est orienté protocoles métier) et que CMSPEM est dédié aux processus collaboratifs.

1. https://www.omg.org/spec/SPEM/2.0/PDF

4.3 Méta-modèle de collaboration (MMCollab)

4.3.1 Structuration en paquetages de MMCollab

MMCollab est un méta-modèle support à la formalisation de la collaboration en général et de la prise de décision en groupe en particulier. Il permet de décrire les concepts nécessaires lors d'une prise de décision collaborative ; ces concepts concernent les acteurs, le choix de la politique de prise de décision, la collecte des propositions sur lesquelles vont porter les évaluations individuelles et les décisions collectives relatives à ces propositions.
MMCollab importe les paquetages *Process Structure* de SPEM et CMSPEM. Il est conforme à MOF [2] [SOLEY et al., 2000] et est structuré en cinq paquetages :

— *Actors* : concerne les acteurs de la prise de décision ;

— *Proposals* : concerne la collecte des propositions ;

— *Evaluation* : concerne l'élaboration des évaluations individuelles des propositions ;

— *CollectiveDecision* : concerne la politique d'élaboration des décisions collectives ;

— *CoreConcepts* : contient les concepts fondamentaux de MMCollab et importe les quatre autres paquetages comme décrit sur la Figure 4.1.

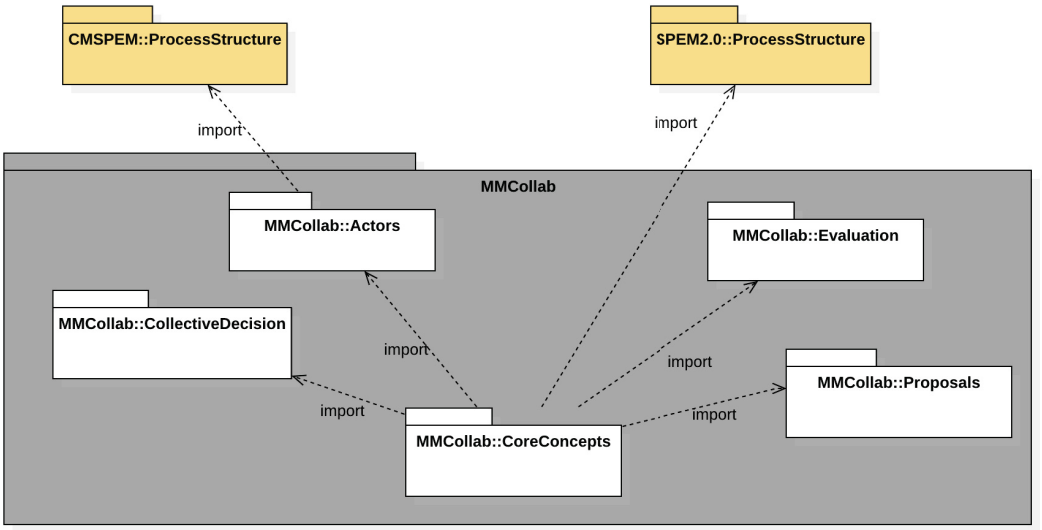


FIGURE 4.1 – Organisation en paquetages du méta-modèle MMCollab.

Une vue globale de MMCollab est donnée dans la Figure 4.2. Dans la suite de cette section, nous présentons les paquetages de MMCollab et leurs principaux concepts. Pour ces derniers nous adoptons le template de présentation classique :

Description décrit la méta-classe. Une figure peut accompagner cette description. Elle met l'accent sur les relations de la méta-classe (relations au sein du paquetage et externes) et ignore volontairement les attributs des concepts en relation avec la méta-classe en question ;

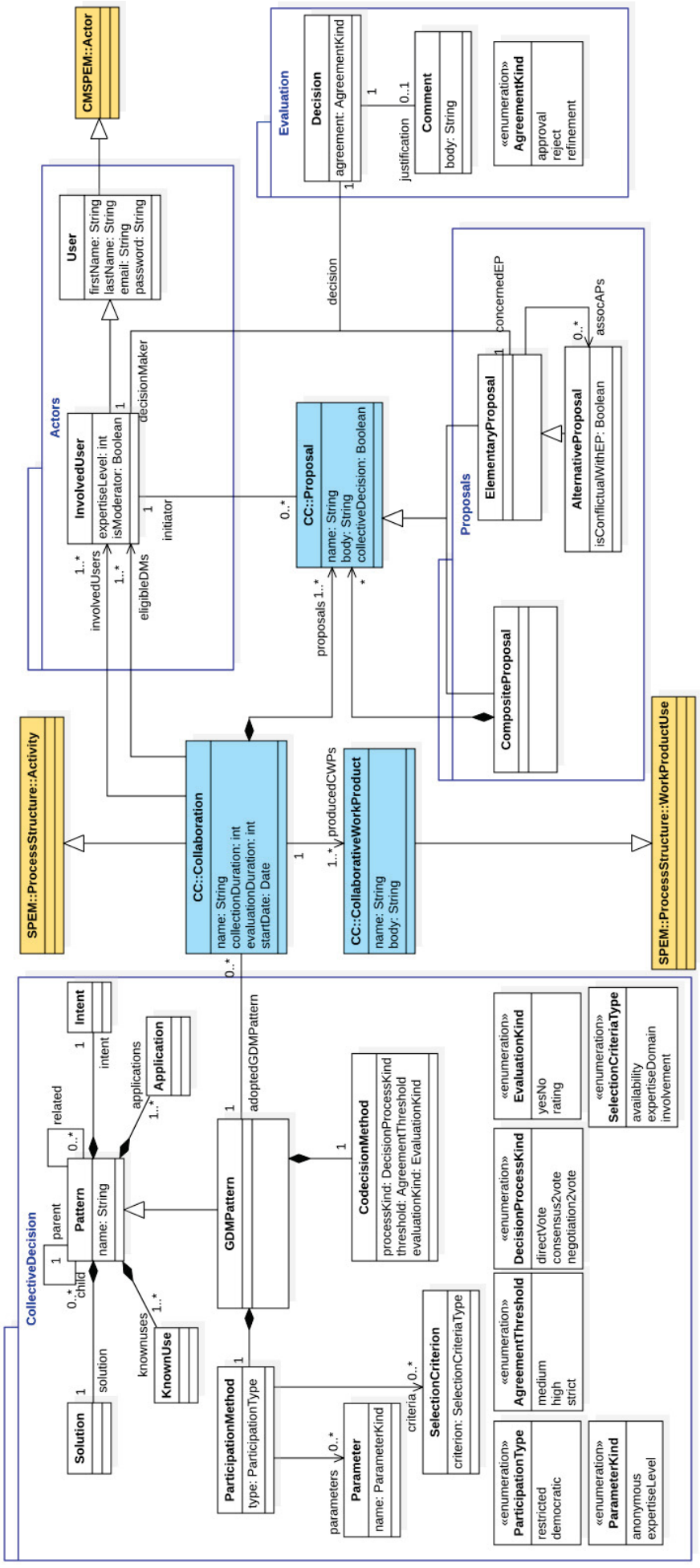
Classe(s) parente(s) précise, le cas échéant, la (les) méta-classe(s) mère(s) du concept ;

Attributs décrit les attributs de la méta-classe ;

Relations au sein du paquetage décrit les associations qui lient cette méta-classe aux concepts du même paquetage ;

Relations externes décrit les associations qui lient cette méta-classe aux concepts des autres paquetages ;

2. Meta Object Facility.



En jaune : Paquetages importés, **En bleu** : Paquetage CoreConcepts (CC) de MMCollab

FIGURE 4.2 – Vue d'ensemble du méta-modèle de collaboration (MMCollab).

Règles de bonne formation liste les règles de bonne formation en langage naturel et en OCL ;

Sémantique liste les autres contraintes (règles de gestion) qui régissent la méta-classe.

Notons que le terme de « méta-classe » est utilisé quand on présente toutes les caractéristiques d'un concept (attributs, relations, etc.), sinon on utilise le terme de « concept ».

4.3.2 Paquetage Core Concepts

Ce paquetage, présenté sur la Figure 4.3, contient les concepts de base définissant la structure d'une collaboration. Il importe le paquetage *Process Structure de SPEM 2.0* en spécialisant ses méta-classes *Activity* et *WorkProductUse* par, respectivement, *Collaboration* et *CollaborativeWorkProduct*.

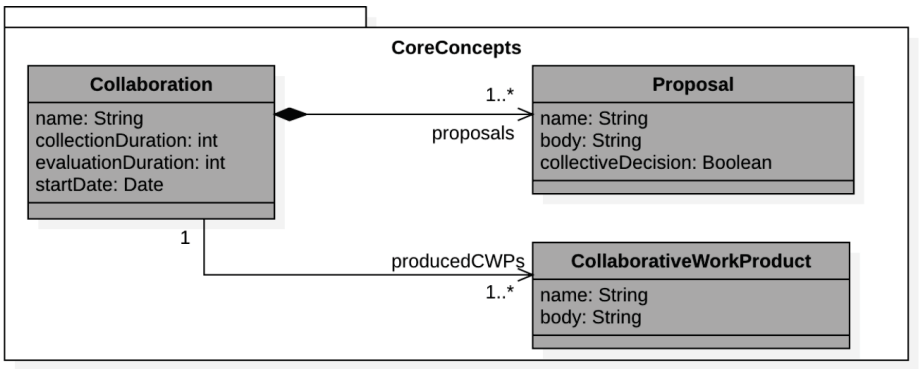


FIGURE 4.3 – Paquetage *CoreConcepts* de MMCollab regroupant les concepts de base.

4.3.2.1 Collaboration

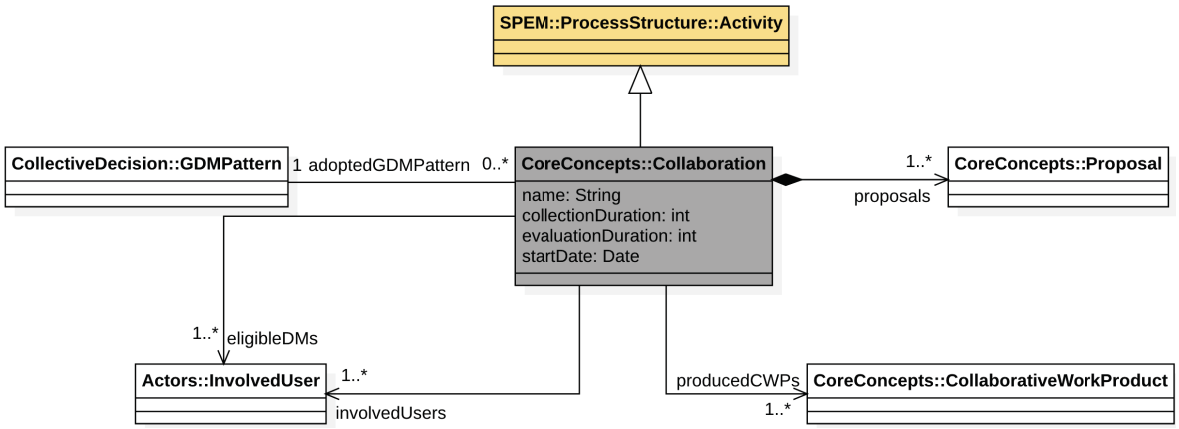


FIGURE 4.4 – Méta-classe *Collaboration* et ses relations dans MMCollab.

Description

Collaboration (Figure 4.4) est la méta-classe centrale de MMCollab. Elle spécialise le concept *Activity* de SPEM pris dans son sens élémentaire et consiste en l'élaboration et l'évaluation d'un ensemble de propositions (*proposals*).

Une collaboration produit un ou plusieurs artefacts (*CollaborativeWorkProduct*) qui rassemblent les propositions (*proposals*) approuvées par les parties prenantes (*InvolvedUser*) selon le patron de prise de décision adopté (*adoptedGDMPattern*).

Classe(s) parente(s)

Activity (from SPEM 2.0 Process Structure).

Attributs

name : désignation de la collaboration (de façon unique).

collectionDuration : durée maximum fixée en jours pour la collecte des propositions.

evaluationDuration : durée maximum fixée en jours pour la collecte des évaluations.

startdate : date de début de la collaboration.

Relations au sein du paquetage

Collaboration se compose d'un ensemble de *Proposal*.

Collaboration produit un ensemble de *CollaborativeWorkProduct*. L'association entre ces deux concepts est uni-directionnelle et est concrétisée par le rôle *producedCWPs*. Les CollaborativeWorkProducts contiennent les propositions approuvées.

Relations externes

Collaboration est liée au concept *GDMPattern* du paquetage *CollectiveDecision* via l'association *adoptedGDMPattern*. Une *Collaboration* est mise en oeuvre via un *GDMPattern*.

Collaboration fait intervenir un ensemble d'utilisateurs via le rôle *involvedUsers* de l'association reliant *Collaboration* et *InvolvedUser*.

Collaboration permet à un sous-ensemble des *involvedUsers*, à savoir les *eligibleDMs*, d'évaluer les propositions faites.

4.3.2.2 Proposal

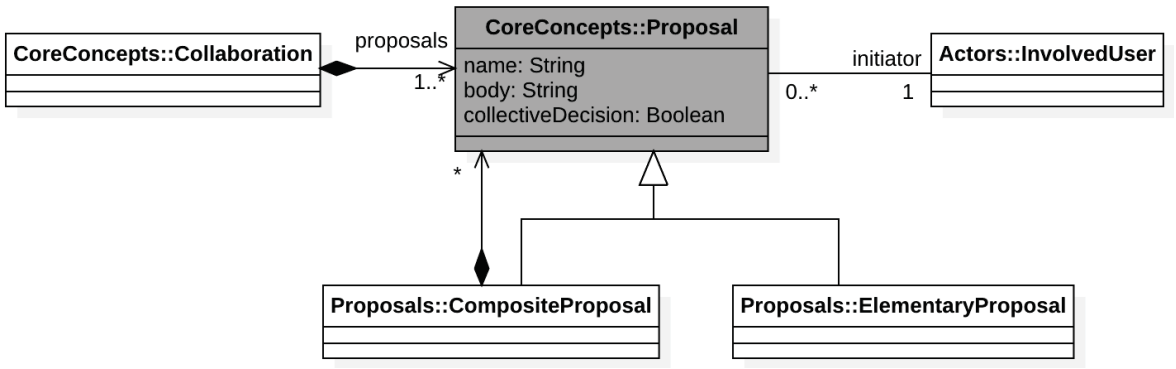


FIGURE 4.5 – Méta-classe *Proposal* et ses relations dans MMCollab.

Description

Proposal (Figure 4.5) est une méta-classe qui décrit une solution proposée au problème devant faire l'objet d'une prise de décision. Elle est caractérisée par un corps (*body*) et une décision collective (*collectiveDecision*), qui représente la décision finale du groupe par rapport à la proposition.

Classe(s) parente(s)

Néant.

Attributs

name : désignation de la proposition (de façon unique).

body : corps de la proposition décrit par une chaîne de caractères qui peut varier d'un simple texte à une expression structurée, qui peut être analysée pour extraire ces éléments (e.g., expression régulière, expression à base d'un langage déclaratif : XML, JSON, etc.).

collectiveDecision : booléen représentant la décision collective (acceptation/refus) prise pour la proposition, il dépend des évaluations individuelles et du patron de prise de décision en groupe adopté.

Relations au sein du paquetage

Association avec le concept *Collaboration* (rôle *proposals* - vu dans la section 4.3.2.1).

Relations externes

Proposal est associé au concept *InvolvedUser* : chaque *Proposal* a un unique initiateur (rôle *initiator*).

Proposal peut être composite (*CompositeProposal* - cf. section 4.3.4.1) ou élémentaire (*ElementaryProposal* - cf. section 4.3.4.2).

Règles de bonne formation

[WF1] Il n'est pas possible d'avoir deux instances de *Proposal* avec le même *body*. i.e, avec un contenu identique.

```
1  context Proposal
2  inv distinctProposals:
3  Proposal.allInstances()->forAll(p1, p2 | p1 <> p2 implies p1.body <> p2.
   body);
```

Listing 4.1 – Contrainte OCL WF1

[WF2] L'initiateur d'une *Proposal* appartient à la liste des *involvedUsers* de la collaboration.

```
1  context Collaboration
2  inv initiatorIsInvolvedUser:
3  self.involvedUsers ->includesAll(self.proposals.initiator);
```

Listing 4.2 – Contrainte OCL WF2

Sémantique

Les décideurs d'une proposition sont constitués à partir de la liste *eligibleDMs* associée à une Collaboration.

L'attribut *collectiveDecision* d'une proposition est mis à jour une fois que tous les décideurs de la proposition ont fait leur évaluation si la durée d'évaluation de la Collaboration (*evaluationDuration*) n'est pas renseignée. Sinon, il est mis à jour quand *evaluationDuration* arrive à échéance. Il est mis à vrai si l'agrégation des évaluations individuelles satisfait les conditions du patron de prise de décision (voir section 4.3.5.3) ; il prend en compte uniquement les évaluations qui ont été faites dans le délai de *evaluationDuration* si ce champ est renseigné.

4.3.2.3 CollaborativeWorkProduct

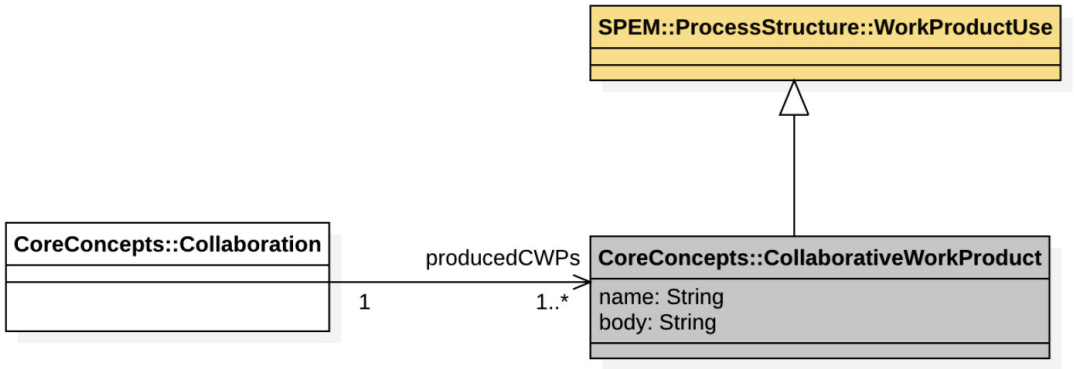


FIGURE 4.6 – Méta-classe *CollaborativeWorkProduct* et ses relations dans MMCollab.

Description

CollaborativeWorkProduct (Figure 4.6) est une méta-classe qui permet de représenter les artefacts produits par une collaboration.

Classe(s) parente(s)

WorkProductUse (from SPEM 2.0 Process Structure).

Attributs

name : désignation du CollaborativeWorkProduct (de façon unique).

body : Cet attribut représente le corps de l'artefact produit par la collaboration. C'est l'ensemble des propositions qui ont été approuvées par le groupe. Le *body* est une expression structurée qui peut être analysée pour extraire ces éléments (e.g., expression régulière, expression à base d'un langage déclaratif : XML, JSON, etc.).

Relations au sein du paquetage

Une *Collaboration* produit un ensemble de *CollaborativeWorkProduct*.

Règles de bonne formation

[WF3] Le produit de la collaboration est fourni une fois qu'une décision collective a été prise par proposition. La méthode *generateCWP()* du concept Collaboration se charge de générer le produit de la collaboration.

```
1  context   Collaboration::generateCWP()
2  pre allProposalsHaveFinalDecision:
3  Proposal.allInstances()->forAll(p| not(p.collectiveDecision.
   oclIsUndefined()));
```

Listing 4.3 – Contrainte OCL WF3

4.3.3 Paquetage Actors

Le paquetage *Actors* (voir Figure 4.7) contient deux concepts relatifs à l'organisation des parties prenantes d'une collaboration : *User* et *InvolvedUser*.
Ce paquetage importe le paquetage *Process Structure* de CMSPEM. Il spécialise le concept *Actor* par le concept *User* ce qui permet de désigner un participant humain, jouant un ou plusieurs rôles dans la collaboration (ex. : *moderator*, *initiator*, *decisionMaker*).

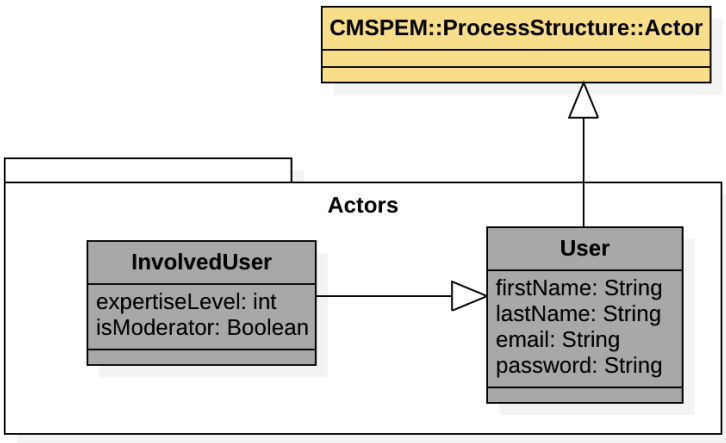


FIGURE 4.7 – Paquetage *Actors* de MMCollab.

4.3.3.1 User

Description

User est une méta-classe qui permet de caractériser les acteurs impliqués dans une collaboration.

Classe(s) parente(s)

Actor (from CMSPEM ProcessStructure).

Attributs

firstName : prénom de l'utilisateur.

lastName : nom de l'utilisateur.

email : adresse e-mail de l'utilisateur.

password : mot de passe de l'utilisateur.

4.3.3.2 InvolvedUser

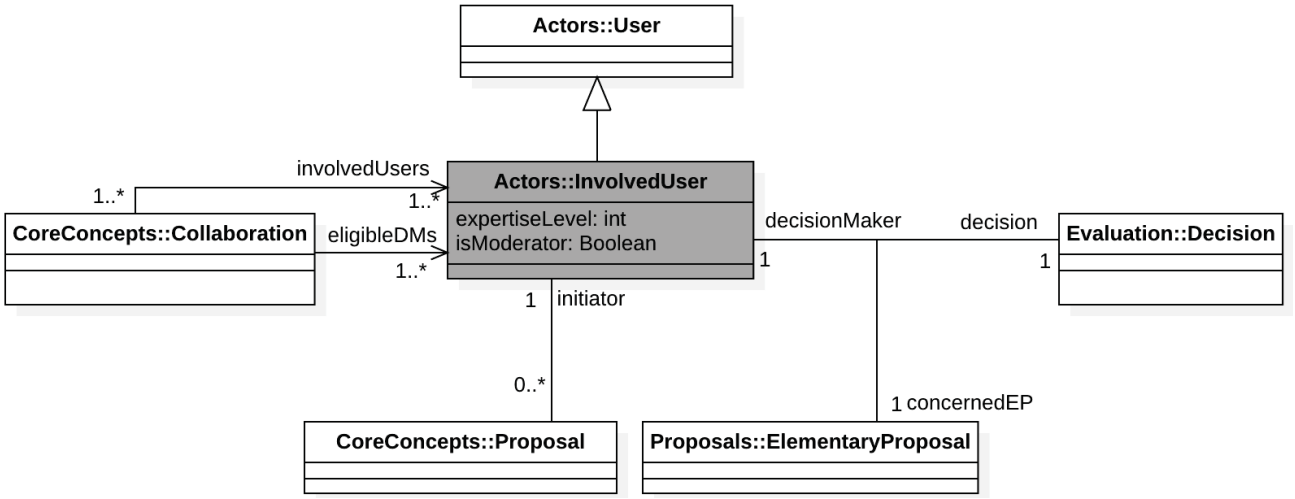


FIGURE 4.8 – Méta-classe *InvolvedUser* et ses relations dans MMCollab.

Description

InvolvedUser (Figure 4.8) est une méta-classe qui permet de représenter les acteurs impliqués dans une collaboration et de spécifier leurs rôles.

Classe(s) parente(s)

Classe parente : *User* (from MMCollab Actors)

Attributs

expertiseLevel : entier compris entre 1 et 5, indiquant le niveau d'expertise de l'acteur dans la collaboration ; la valeur 1 est associée à un niveau débutant alors que la valeur 5 est associée à un niveau d'expertise très avancé. Quand les acteurs de la collaboration ne sont pas considérés d'importance égale, la valeur de cet attribut est utilisée comme poids décisionnel.

isModerator : booléen qui indique si l'acteur a le rôle de modérateur. Le modérateur intervient pour faciliter la collaboration des acteurs, il précise l'objectif de la collaboration et fixe les durées *collectionDuration* et *evaluationDuration*. Il peut être désigné par les acteurs impliqués dans la collaboration s'ils se mettent d'accord, sinon c'est l'acteur ayant initié la première proposition.

Relations externes

InvolvedUser est lié par une association au concept Collaboration, puisqu'une Collaboration fait intervenir un ou plusieurs *involvedUsers*.

InvolvedUser est l'extrémité, rôle *eligibleDMs*, d'une association dirigée vers le concept *Collaboration*. *eligibleDMs* constitue l'ensemble des acteurs éligibles pour évaluer les propositions (cardinalité 1..*).

InvolvedUser possède une association 0..* avec la classe *Proposal* : chaque *Proposal* a un unique initiateur (rôle *initiator*).

InvolvedUser est aussi utilisé dans une relation ternaire avec le concept *ElementaryProposal* du paquetage *Proposals* et le concept *Decision* du paquetage *Evaluation*. Cette relation permet d'associer à chaque couple *ElementaryProposal* (rôle *concernedEp*) et *InvolvedUser* (rôle *decisionMaker*), une *Decision* (concept détaillé dans la section 4.3.6.1).

Règles de bonne formation

[WF4] Un seul modérateur est admis par collaboration.

```
1  context   Collaboration
2  inv moderatorIsUnique:
3  self.involvedUsers->select(m|m.isModerator)->size()=1;
```

Listing 4.4 – Contrainte OCL WF4

[WF5] *eligibleDMs* est un sous-ensemble des *involvedUsers* d'une Collaboration. Il représente les acteurs éligibles pour l'évaluation des propositions, en fonction du patron de prise de décision en groupe adopté.

```
1  context Collaboration
2  inv eligibleDecisionMakers:
3  self.involvedUsers ->includesAll(self.eligibleDMs);
```

Listing 4.5 – Contrainte OCL WF5

[WF6] *decisionMaker* est un membre de *eligibleDMs*.

```
1  context Collaboration
2  inv eligibleDecisionMakers_2:
3  self.eligibleDMs->includesAll(ElementaryProposal.allInstances().decision
   .decisionMaker);
```

Listing 4.6 – Contrainte OCL WF6

4.3.4 Paquetage Proposals

Ce paquetage contient les concepts qui structurent l'organisation des propositions élaborées lors d'une collaboration. Ses trois concepts : propositions composites (section 4.3.4.1), propositions élémentaires (section 4.3.4.2) et propositions alternatives (section 4.3.4.3) spécialisent le concept *Proposal*. Ce dernier appartient au paquetage *CoreConcepts* puisqu'il fait partie des concepts centraux de MMCollab.

4.3.4.1 CompositeProposal

Description

CompositeProposal (voir Figure 4.9) est une méta-classe décrite par l'intermédiaire du patron de conception « composite ».

Classe(s) parente(s)

Proposal (from MMCollab Core Concepts).

Attributs

Néant.

Relations externes

CompositeProposal est lié à *Proposal* par le patron de conception « composite ».

Sémantique

L'initiateur d'une proposition composite est le même pour les propositions élémentaires qu'elle contient.

Une *CompositeProposal* est approuvée si toutes les *ElementaryProposal* qui la composent ont été approuvées puisqu'elle est vue comme une transaction atomique.

4.3.4.2 ElementaryProposal

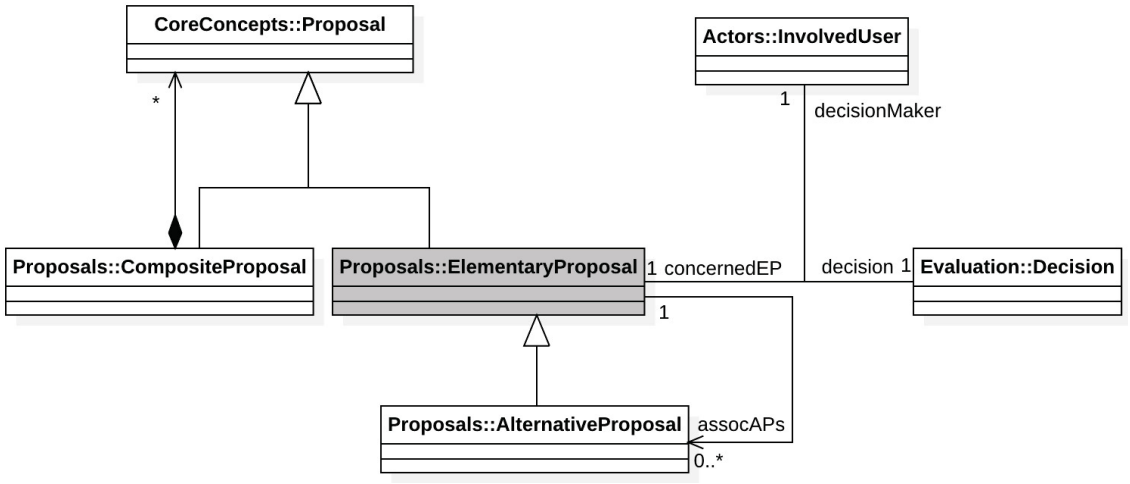


FIGURE 4.9 – Méta-classe *ElementaryProposal* et ses relations dans MMCollab.

Description

ElementaryProposal (Figure 4.9) est une méta-classe qui représente les feuilles du patron de conception *CompositeProposal*.

Classe(s) parente(s)

Proposal (from MMCollab Core Concepts).

Attributs

Néant.

Relations au sein du paquetage

ElementaryProposal est une feuille de *CompositeProposal*.

ElementaryProposal peut donner lieu à des propositions alternatives *AlternativeProposal* (association avec le rôle *assocAPs* du côté de *AlternativeProposal*).

Relations externes

ElementaryProposal est lié à *Proposal* par le patron composite.

La relation ternaire entre *InvolvedUser, ElementaryProposal* et *Decision* permet de préciser pour chaque couple *decisionMaker* (rôle de *InvolvedUser*) et *concernedEP* (rôle de *ElementaryProposal*), la decision prise (rôle de *Evaluation : :Decision*) (i.e. l'évaluation individuelle de la proposition élémentaire - le concept *Evaluation : :Decision* est détaillé dans la section 4.3.6.1).

4.3.4.3 AlternativeProposal

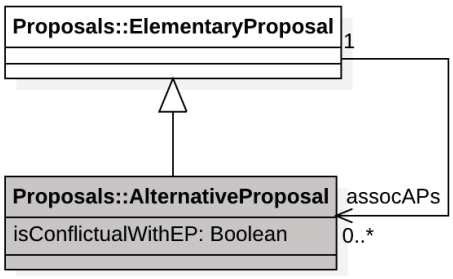


FIGURE 4.10 – Méta-classe *AlternativeProposal* et ses relations dans MMCollab.

Description

AlternativeProposal (Figure 4.10) est une méta-classe qui spécialise *ElementaryProposal*. On recourt à des propositions alternatives pour raffiner des propositions élémentaires durant une collaboration donnée.

Classe(s) parente(s)

ElementaryProposal (from MMCollab Core Concepts).

Attributs

isConflictualWithEP : booléen qui indique si une AlternativeProposal est conflictuelle avec la proposition élémentaire dont elle découle.

Relations au sein du paquetage

AlternativeProposal résulte d'une *ElementaryProposal* qui nécessite d'être raffinée.

Sémantique

L'attribut *isConflictualWithEP* permet de détecter les propositions en conflit.
S'il est positionné à Vrai par l'initiateur de la proposition alternative, l'*AlternativeProposal* et l'*ElementaryProposal* dont elle découle ne peuvent pas être approuvées conjointement par le groupe, puisque chaque décideur peut approuver uniquement l'une des deux.

4.3.5 Paquetage Collective Decision

Ce paquetage, illustré sur la Figure 4.11, contient les concepts qui décrivent les éléments nécessaires à l'établissement des décisions collectives sur les propositions. Le concept principal de ce paquetage est *GDMPattern* qui est une spécialisation de la méta-classe *Pattern* du même paquetage.
GDMPattern est caractérisé par la méthode de prise de décision collective (*CoDecisionMethod*) et la méthode de participation des membres de la collaboration (*ParticipationMethod*). C'est le *GDMPattern* qui permet d'agréger les évaluations individuelles des propositions et de parvenir ainsi aux décisions collectives qui permettent de clore les collaborations.

4.3.5.1 Pattern

Description

Pattern (voir Figure 4.11) est une méta-classe qui représente la structure adoptée pour présenter le patron de prise de décision décrit plus loin. Cette structure est largement inspirée de celle des patrons de conception [GAMMA, 1995].

Classe(s) parente(s)

Néant.

Attributs

name : caractérise le patron en le distinguant des autres patrons.

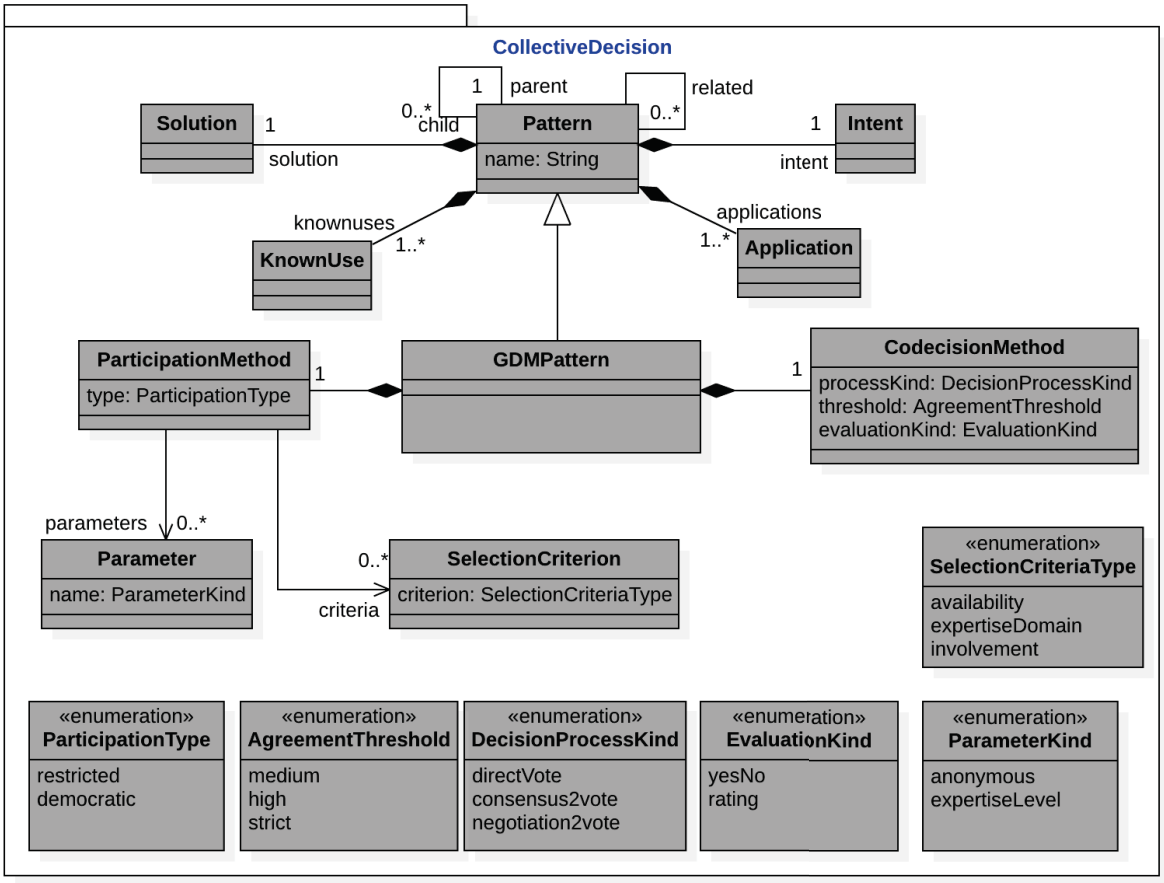


FIGURE 4.11 – Paquetage *CollectiveDecision* de MMCollab.

Relations au sein du paquetage

child-parent sont les deux rôles de l'association réflexive qui permet de décrire les relations de spécialisation entre les patrons.

related est une association réflexive qui permet de lister les patrons répondant à des situations comparables.

Le rôle *intent* relie *Pattern* à la méta-classe *Intent* qui représente le problème récurrent résolu par le patron.

Le rôle *applications* relie *Pattern* à la méta-classe *Application* qui représente les contextes dans lesquels le patron est destiné à être utilisé.

Le rôle *knownuses* relie *Pattern* à la méta-classe *KnownUse* qui permet de donner des exemples concrets d'applications du patron.

Le rôle *solution* relie *Pattern* à la méta-classe *Solution* qui décrit comment le patron répond au problème défini (*intent*) et comment il est appliqué.

4.3.5.2 GDMPattern

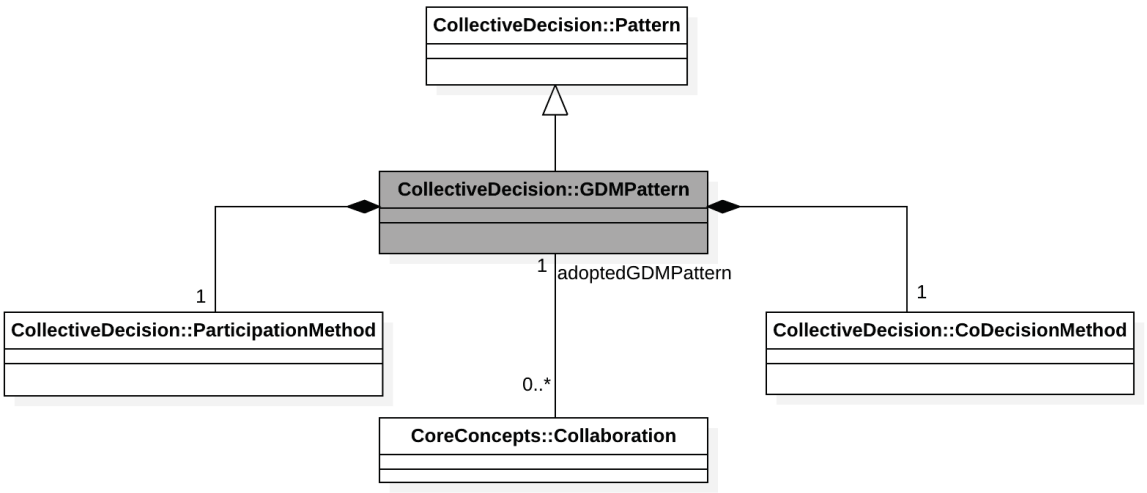


FIGURE 4.12 – Méta-classe *GDMPattern* et ses relations dans MMCollab.

Description

GDMPattern (Figure 4.12) est une spécialisation de *Pattern*. C'est un patron dédié à la prise de décision collective au sein d'une collaboration. Il permet de préciser la manière selon laquelle la décision de groupe est élaborée en matière de méthodes de participation et de co-décision. La méthode de participation précise le type de participation des parties prenantes et le cas échéant précise aussi les critères sur lesquels la restriction est basée. La méthode de co-décision précise à la fois comment les évaluations individuelles sont élaborées et agrégées (le type de processus adopté et le seuil d'agrément prédéfini).

Classe(s) parente(s)

Pattern (from MMCollab CollectiveDecision).

Relations au sein du paquetage

GDMPattern est composé d'une méthode de co-décision *CoDecisionMethod* (section 4.3.5.3) et d'une méthode de participation *ParticipationMethod* (section 4.3.5.4).

Relations externes

Un *GDMPattern* est adopté par *Collaboration*; il permet d'aboutir aux décisions collectives relatives aux propositions. Le rôle *adoptedGDMPattern* de GDMPattern précise ce lien.

4.3.5.3 CoDecisionMethod

Description

CoDecisionMethod (Figure 4.13) est une méta-classe qui représente la méthode de prise de décision collective. Elle précise comment les évaluations individuelles sont élaborées et agrégées.

Classe(s) parente(s)

Néant.

Attributs

processKind : indique le type de processus de décision suivi. Les types possibles sont définis par l'énumération *DecisionProcessKind* : on procède soit à un vote direct (*directVote*), soit à la recherche d'un consensus suivi d'un vote non contraignant (*consensus2vote*) soit à une négociation suivie d'un vote (*negotiation2vote*).

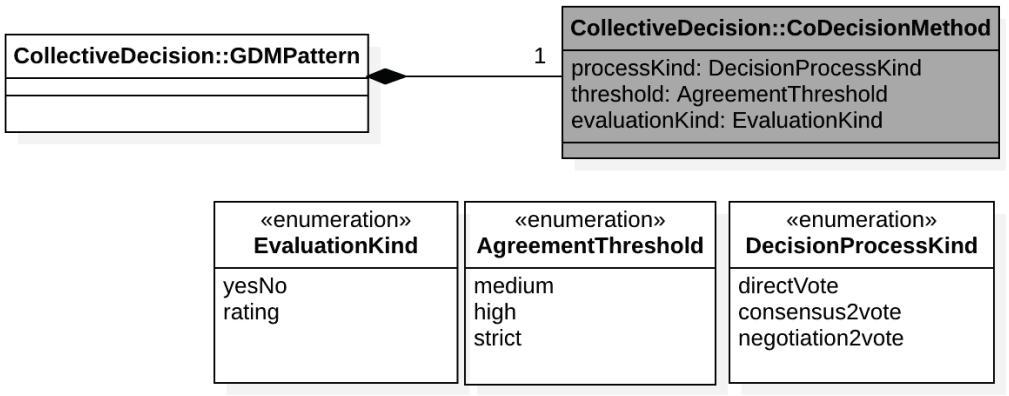


FIGURE 4.13 – Méta-classe *CoDecisionMethod* et ses relations dans MMCollab.

threshold : représente le seuil d'agrément prédéfini par le modérateur de la collaboration pour qu'une proposition soit acceptée. C'est une valeur parmi celles fixées dans l'énumération *AgreementThreshold* du paquetage *CollectiveDecision* : *medium* pour un agrément entre 50% et 66%, *high* pour un agrément entre 67% et 99% et *strict* pour un agrément de 100%. Notons que ces valeurs ne sont pas figées, elles peuvent être adaptées selon les particularités des situations de collaboration.

evaluationKind : indique la manière d'évaluer les propositions. Elle prend une valeur parmi celles fixées dans l'énumération *EvaluationKind*, i.e. une évaluation à base de notes (*rating*), ou par approbation/rejet (*YesNo*).

Relations au sein du paquetage

CoDecisionMethod est un composant de *GDMPattern*.

Sémantique

Pour l'attribut *processKind*, le choix de la majorité est favorisé dans le cas de vote direct. Le consensus quant à lui exige que les membres du groupe affinent les propositions jusqu'à ce qu'un seuil d'agrément strict de 100% soit atteint. Ainsi, le consensus s'efforce d'intégrer les perspectives, les besoins, et finalement l'approbation de chacun. La négociation suivie de vote est semblable au consensus, mais au lieu de rechercher un seuil strict, elle pousse le groupe à adopter une solution acceptable (agrément<100%).

4.3.5.4 ParticipationMethod

Description

ParticipationMethod (Figure 4.14) est une méta-classe qui représente la façon dont les parties prenantes interviennent dans une collaboration. Elle précise aussi les critères de sélection des parties prenantes dans le cas où des restrictions sont faites.

Classe(s) parente(s)

Néant.

Attributs

type : désigne le type de participation des acteurs. Il prend une valeur parmi celles définies dans l'énumération *ParticipationType*, à savoir *restricted* ou *democratic*.

Une participation est caractérisée de *democratic* quand tous les *involvedUsers* participent, sinon elle est *restricted*.

Relations au sein du paquetage

ParticipationMethod est un composant de *GDMPattern*.

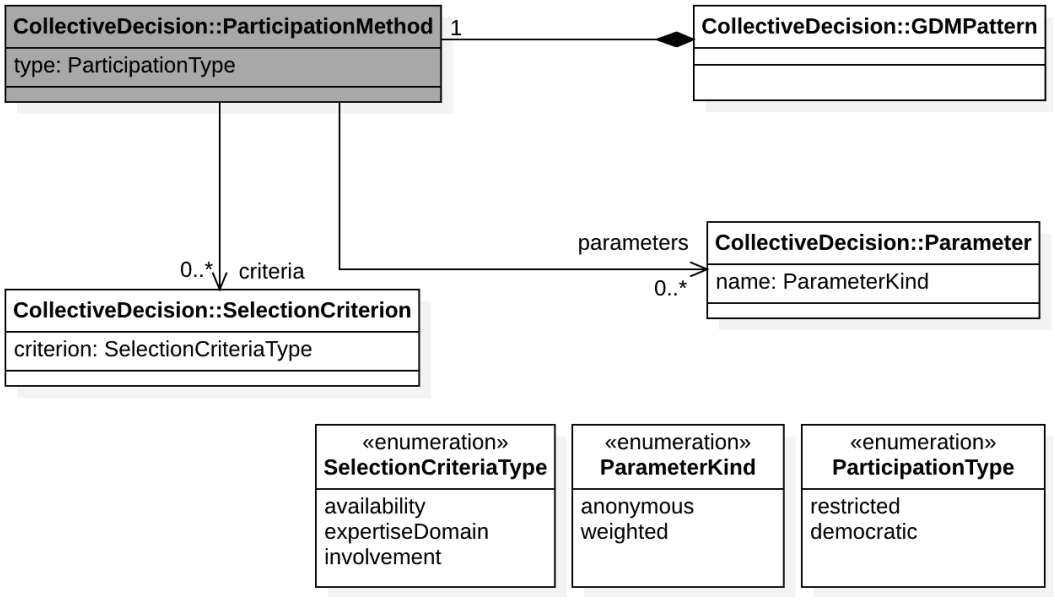


FIGURE 4.14 – Méta-classe *ParticipationMethod* et ses relations dans MMCollab.

Pour le type de participation qui choisit un sous-ensemble des *involvedUsers* (à savoir *restricted*), le critère de sélection de ce sous-ensemble est spécifié grâce à l'association entre *ParticipationMethod* et *SelectionCriterion* (rôle *criteria* du côté de *SelectionCriterion*).

ParticipationMethod peut être précisée grâce à l'association qui la lie à *Parameter*.

Sémantique

Les valeurs possibles pour *criterion* sont définies dans *SelectionCriteriaType* (*availability, expertiseDomain* ou *involvement*). Ainsi les acteurs sont choisis selon leur disponibilité, leur niveau d'expérience ou leur implication dans les propositions.

Les paramètres possibles sont spécifiés dans l'énumération *ParameterKind* (*weighted* ou *anonymous*). Ces deux paramètres n'étant pas contradictoires, ils peuvent être appliqués conjointement (cardinalité 0..*).

Le paramètre *weighted* permet de spécifier que les acteurs sont discriminés sur la base de leurs expérience et connaissances. Dans ce cas, les choix individuels des membres sont pondérés selon les poids décisionnels qui leur sont conférés par la valeur de l'attribut *expertiseLevel* du concept *InvolvedUser*.

Le paramètre *anonymous* spécifie que la décision est prise anonymement.

Règles de bonne formation

[WF7] *criteria* doivent être précisés dans le cas d'une *ParticipationMethod restricted*.

```
1  context ParticipationMethod
2  inv restrictedParticipation:
3  self.type = ParticipationType::restricted implies self.criteria->size()
   >=1;
4
```

Listing 4.7 – Contrainte OCL WF7

[WF8] *criteria* ne sont pas applicables dans une *ParticipationMethod democratic*.

```
1  context ParticipationMethod
2  inv democraticParticipation:
3  self.type = ParticipationType::democratic implies self.criteria->size()
   =0;
```

Listing 4.8 – Contrainte OCL WF8

4.3.6 Paquetage Evaluation

FIGURE 4.15 – Paquetage *Evaluation* de MMCollab.

Ce paquetage (voir Figure 4.15) contient les concepts qui permettent de structurer les évaluations individuelles (associées aux différentes propositions) afin de les agréger pour aboutir à une décision collective par proposition (ou par groupe de propositions s'il s'agit d'une proposition composite). Nous utilisons deux concepts pour décrire les évaluations individuelles : *Decision* et *Comment*. L'énumération *AgreementKind* décrit les différents types d'évaluations possibles : accord (*approval*), rejet (*reject*), raffinement (*refinement*).

4.3.6.1 Decision

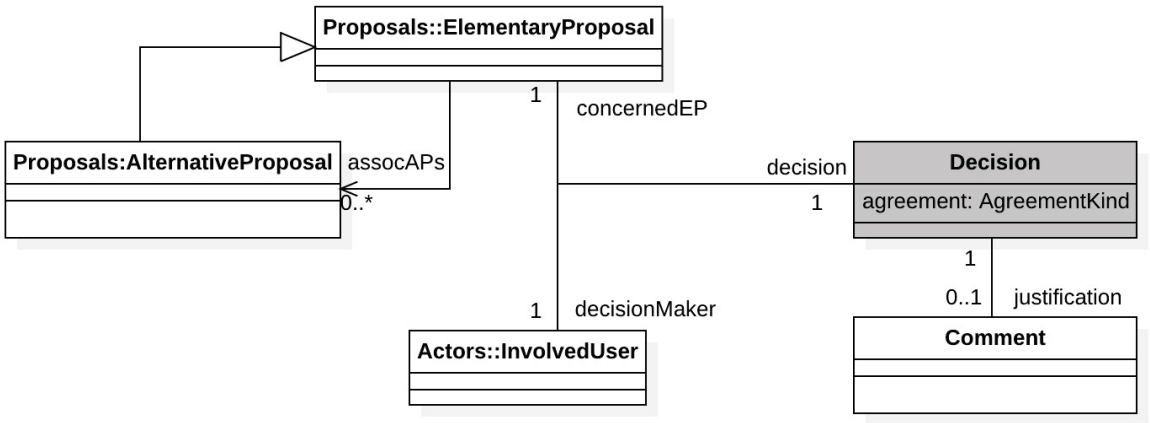


FIGURE 4.16 – Méta-classe *Decision* et ses relations dans MMCollab.

Description

Decision (Figure 4.16) est une méta-classe qui représente l'évaluation individuelle d'un acteur pour une proposition élémentaire.

Classe(s) parente(s)

Néant.

Attributs

agreement : l'évaluation individuelle d'un acteur pour une proposition donnée. Elle prend une valeur parmi celles fixées dans l'énumération *AgreementKind*, à savoir : *approval*, *reject* ou *refinement*.

Relations au sein du paquetage

La décision peut faire l'objet d'un commentaire *Comment*. Le commentaire est obligatoire dans le cas d'une décision de rejet d'une proposition (valeur *reject* de l'attribut *agreement*).

Une *Decision* avec la valeur *refinement* dans son attribut *agreement* pour une proposition élémentaire (EP) signifie que le décideur considère que l'EP doit être raffinée. Ainsi, il doit proposer une proposition alternative (*AlternativeProposal*) qui affine cette EP.

Relations externes

une *Decision* est associée à chaque couple *concernedEP* et *decisionMaker*.

Règles de bonne formation

[WF9] Si un des décideurs d'une proposition élémentaire a positionné l'attribut *agreement* de la *Decision* à *refinement*, il doit y avoir au moins une proposition alternative associée à cette proposition élémentaire, avec comme initiateur ce décideur.

```
1  context Decision
2  inv APRefinesEP:
3  Decision.allInstances()->forAll(d| d.agreement = agreementKind::
   refinement implies d.concernedEP.assocAPs->size()>=1 and d.concernedEP.
   assocAPs.initiator=d.decisionMaker);
```

Listing 4.9 – Contrainte OCL WF9

Sémantique

L'agrégation des évaluations individuelles est faite en se basant sur le *GDMPattern* adopté. Par la suite, on la nomme *acceptation*. Ainsi acceptation(A) est l'agrégation des évaluations individuelles pour la proposition A, en considérant qu'une approbation est traduite par 1, un rejet est traduit par 0, et un raffinement est traduit par 0 si les propositions élémentaire et alternative sont en conflit et par 1 sinon. De cette manière les évaluations individuelles peuvent être agrégées en calculant leur moyenne par exemple.

4.3.6.2 Comment

Description

Comment est une méta-classe qui représente la justification associée à une décision.

Classe(s) parente(s)

Néant.

Attributs

body : le corps du commentaire. C'est une chaîne de caractères qui peut être structurée pour extraire ses éléments grâce à une expression régulière, ou un format de données de type XML.

Relations au sein du paquetage

Un *Comment* est associé à une décision. Le commentaire est obligatoire dans le cas d'une décision de rejet d'une proposition (valeur *reject* de l'attribut *agreement*).

Règles de bonne formation

[WF10] Si un décideur d'une proposition élémentaire a positionné l'attribut *agreement* de la *Decision* à *reject*, il doit fournir un commentaire pour justifier sa décision.

```
1  context Decision
2  inv commentByReject:
3  Decision.allInstances()->forAll(d| d.agreement = agreementKind::reject
   implies d.justification->notEmpty());
```

Listing 4.10 – Contrainte OCL WF10

4.3.7 Syntaxe concrète du méta-modèle MMCollab

Les syntaxes concrètes d'un langage fournissent des formalismes pour manipuler les concepts de la syntaxe abstraite et ainsi représenter des instances du méta-modèle. Nous avons élaboré une syntaxe concrète graphique pour MMCollab inspirée de celles proposées pour les méta-modèles SPEM et CMSPEM.
Le tableau 4.1 ci-dessous présente les représentations graphiques définies pour les concepts de MMCollab.

TABLEAU 4.1 – Représentations graphiques des concepts de MMCollab.

Paquetage CoreConcepts	Collaboration	Proposal	CollaborativeWorkProduct
Paquetage Actors	Acteur	Initiateur	Modérateur
		Décideur	
Autres paquetages	Évaluation individuelle	Décision collective	Politique de décision
		Conflit de propositions	

4.4 Instanciation de MMCollab

Collaboration - concept central de MMCollab - est une activité qui consiste en l'élaboration et l'évaluation d'un ensemble de propositions afin d'aboutir à des décisions collectives en respectant un patron de prise de décision.
Les situations de collaboration varient selon leur contexte et objectif. Dans cette section, nous commençons par présenter un exemple illustratif simple de situation de collaboration afin d'instancier les méta-classes *Collaboration*, *Proposal*, *Decision* et *GDMPattern*. L'objectif de la section est de définir et illustrer les instances de la méta-classe *GDMPattern* que nous appelons « *Politiques de décision* ».
Une *politique de décision* décrit une pratique récurrente pour la prise de décision en groupe. Nous décrivons tout au long de cette section des politiques de décision représentatives et les appliquons à la situation de collaboration introduite dans la section 4.4.1. Un exemple plus détaillé est présenté dans le chapitre 5, dans lequel nous nous intéressons aux situations de collaboration lors du processus d'alignement de modèles.

4.4.1 Exemple de situation de collaboration : Définir des Règles de Gestion (RGs)

Considérons une *Collaboration* qui vise à définir les règles de gestion à mettre dans le cahier des charges d'un projet informatique. Le responsable du projet Bob demande à trois concepteurs Alice, Claire et Jack de définir les règles fonctionnelles à respecter. Nous exploitons MMCollab pour gérer cette *Collaboration*, Bob jouant le rôle de *modérateur*.
La Figure 4.17 illustre l'instanciation de certains concepts de MMCollab pour cette collaboration : Les *propositions* sont l'ensemble des règles de gestion (RGs). Les acteurs impliqués sont Bob, Alice, Claire et Jack. Les *produits de la collaboration* sont les règles de gestion validées (RGV). La collaboration adopte une *politique de décision* (DP).
La Figure 4.17 montre synthétiquement que trois règles de gestion ont été proposées : RGa_1, RGa_2 et RGj_1. Notons que RGa_1 et RGa_2 sont proposées par Alice ; et RGj_1 par Jack. la Figure ne précise pas l'initiateur de chaque proposition ; cette information sera accessible lors de l'évaluation des propositions.

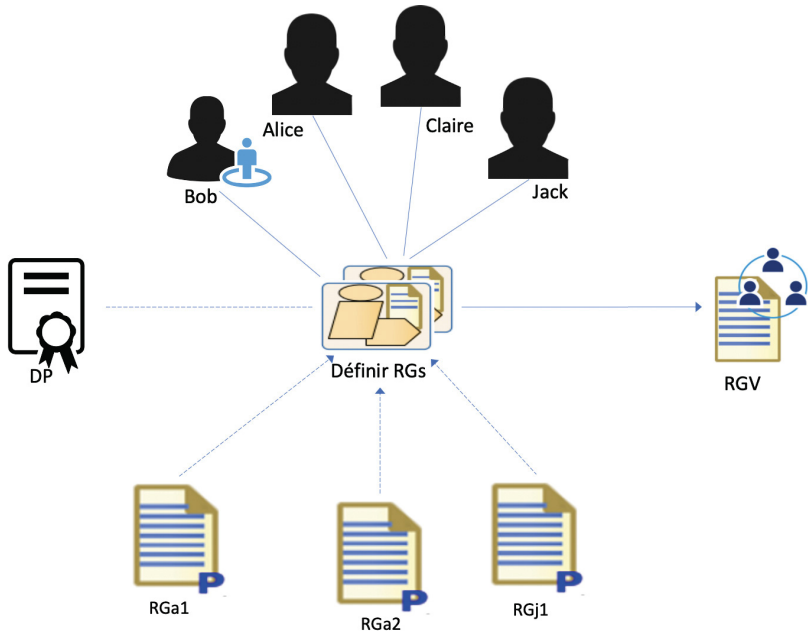


FIGURE 4.17 – Situation de collaboration « Définir des Règles de Gestion » avant le choix de la DP.

Bob étant le modérateur de la collaboration, c'est à lui de choisir la politique de décision à adopter pour décider des règles de gestion à garder.
Dans la section suivante, nous définissons sept politiques de décision - instances de *GDMPattern* - et comparons leur application dans cette situation de collaboration. Ces politiques de décision pouvant être utilisées dans d'autres contextes et situations de collaboration, nous les présentons dans des sections dédiées.

4.4.2 DecisionPolicy : Instance de CollectiveDecision : :GDMPattern

GDMPattern, tel que présenté dans la section 4.3.5.2, est un type particulier de patron qui permet de formaliser la prise de décision collective.
Un *GDMPattern* est caractérisé par une *méthode de co-décision* et un *type de participation des acteurs* à la collaboration. Les configurations, en terme de combinaison de valeurs, de ces deux concepts et des concepts qui leur sont liés nous ont permis de diviser les instances de GDMPattern - que nous nommons politique de Décision (*DecisionPolicy*) - en politiques démocratiques (*DemocraticDP*) versus restrictives (*RestrictiveDP*) (selon leur type de participation) et en politiques à un seul tour (*SingleElectionDP*) versus à plusieurs tours (*IterativeDP*) (selon le nombre de tours requis pour aboutir à la décision collective).

La Figure 4.18 présente une hiérarchisation des sept politiques de décision définies avec deux politiques de décision abstraites : Taking Advice et Delegating. Les sept politiques de décision sont les suivantes :

— *Restricted Taking Advice*;

— *Democratic Taking Advice*;

— *Majority deciding*;

— *Consenting together*;

— *Negotiating together*;

— *Delegated Negotiating*;

— *Delegated Voting.*

TakingAdvice est une politique de décision à un seul tour : le décideur consulte des personnes et c'est à lui seul de décider. Elle est réalisée par *Democratic Taking Advice* et *Restricted Taking Advice*. Il s'agit de *Democratic Taking Advice* quand le décideur consulte tous les acteurs impliqués dans la collaboration sans distinction et de *Restricted Taking Advice* dans le cas contraire.
MajorityDeciding (décision à la majorité) est une politique de décision démocratique. Elle réalise l'interface *SingleElectionDP* puisque sa mise en oeuvre se déroule en un seul tour. Cela signifie que si les acteurs concernés n'aboutissent pas au seuil d'agrément fixé, soit ils se mettent d'accord avec le modérateur pour ajuster ce seuil, soit ils ré-évaluent les propositions.
ConsentingTogether et *NegotiatingTogether* sont des politiques de décision itératives (i.e. à plusieurs tours). *ConsentingTogether* requiert un seuil d'agrément *strict* fixé à 100% tandis que *NegotiatingTogether* fonctionne avec des valeurs de seuil moyen ou élevé (inférieures à 100%).
Delegating est une politique de décision restrictive, ce qui signifie que le critère de sélection des acteurs doit être précisé. *Delegating* est réalisée par *Delegated Voting* quand elle se déroule en un seul tour et par *Delegated Negotiating* dans le cas contraire.

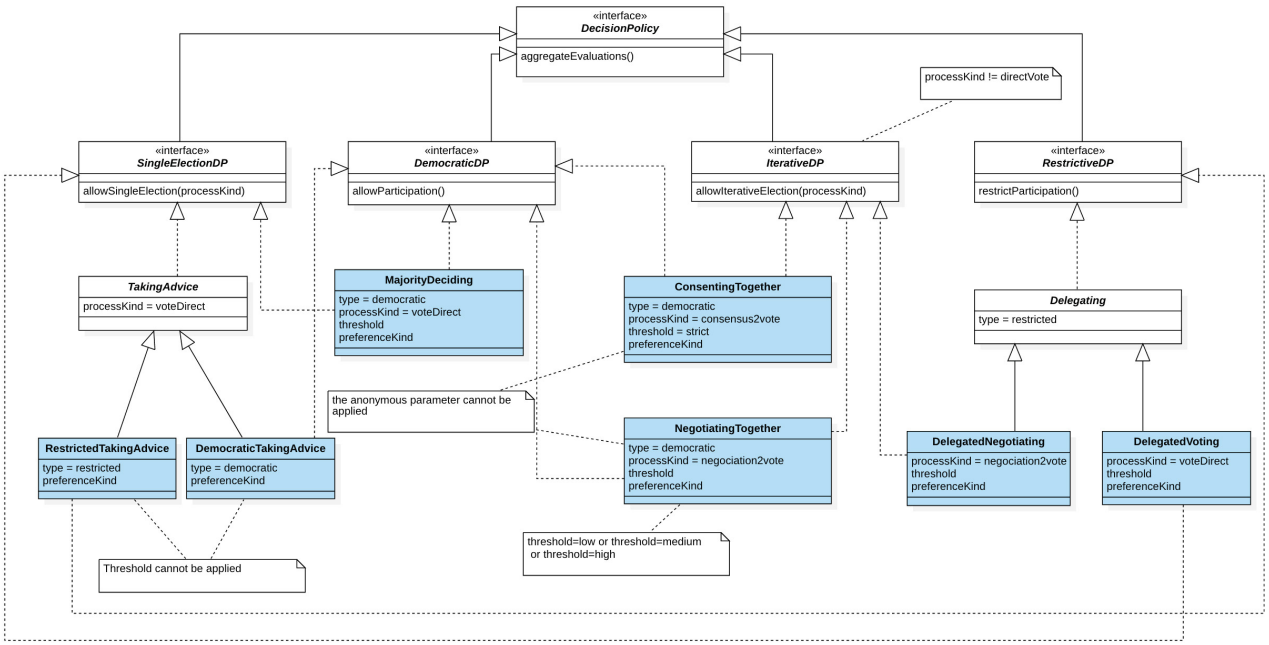


FIGURE 4.18 – Structuration des politiques de décision (les politiques concrètes sont en bleu).

Ces sept politiques de décision ne sont pas figées et peuvent être complétées par d'autres selon les exigences spécifiques des contextes d'application, en explorant les combinaisons possibles des éléments qui les définissent. En effet, le choix de la politique de décision revient à faire quatre choix concernant :

— L'implication (*involvement*) des acteurs dans la prise de décision (i.e, politiques démocratiques versus restrictives).

— La possibilité de recourir à plusieurs tours lors de la mise en oeuvre de la politique.

— La considération équitable des participants (pondération ou non de leurs préférences).

— L'obligation d'avoir une approbation unanime.

Dans les sections suivantes, nous présentons ces politiques de décision en utilisant un formalisme classique pour décrire les patrons [GAMMA, 1995], qui correspond aux caractéristiques de la méta-classe *CollectiveDecision : :Pattern* (i.e. nom (*name*), objectif (*intent*), applications (*applications*), solution (*solution*), utilisations connues (*knownuses*), politiques de décision connexes (*related*)). Notons que pour *Restricted Taking Advice* et *Democratic Taking Advice* d'un côté et *Delegated Voting* et *Delegated Negotiating* de l'autre côté, nous présentons leurs politiques parentes *Taking Advice* et *Delegating* puisque les mises en oeuvre se ressemblent. Ainsi nous présentons cinq politiques de décision, à savoir : *Taking Advice*, *Majority Deciding*, *Consenting Together*, *Negotiating Together* et *Delegating*. Pour chaque politique de décision, nous donnons ci-dessous sa description et son application à l'exemple introduit ci-dessus.

4.4.3 Politique de décision « Taking Advice »

4.4.3.1 Description de « Taking Advice »

Nom *Taking advice*. Cette politique possède deux spécialisations concrètes *Restricted Taking Advice* et *Democratic Taking Advice*.

La politique *Restricted Taking Advice* implique uniquement certains membres du groupe, tandis que *Democratic Taking Advice* implique tout le monde.

Objectif Le décideur demande l'avis de personnes compétentes tout en restant seul décisionnaire.

Applications Cette politique de décision est à utiliser dans les cas suivants :

— Manque d'expertise ou de connaissances techniques de certains acteurs impliqués dans la collaboration ;

— Manque de vue globale des acteurs impliqués dans la collaboration et risque de ne pas cerner toutes les alternatives possibles ;

— Membres du groupe réticents à la prise de responsabilité au sein d'une collaboration (le fait d'être seulement consulté allège la responsabilité).

Solution Dans la mise en place de cette politique de décision, deux rôles sont nécessaires : un décideur, le *decisionMaker*, qui prend seul la décision finale et joue le rôle de modérateur, et un ensemble de personnes à consulter, *consultedActors*, désignées par C1, C2, .., Cn dans le diagramme d'activités de la Figure 4.19. Les activités associées à chacun de ces rôles sont décrites dans ce diagramme.

Tout d'abord le *decisionMaker* définit la situation de collaboration en fixant le sujet et l'objectif de la prise de décision (qui correspond à *intent* de *Pattern*), le degré d'implication des personnes à consulter (*involvement degree*).

Dans le cas d'une implication faible (*low involvement*), les personnes consultées participent uniquement à l'élaboration des propositions, alors que dans le cas d'une implication forte (*high involvement*), non seulement elles participent à l'élaboration des propositions (ensemble des *Proposal*) mais elles expriment aussi leurs préférences par rapport à l'ensemble des propositions SelectionCriterion (ensemble des *Decision*).

Le *decisionMaker* fixe aussi la durée de la consultation (*duration*) qui est égal à la valeur de l'attribut *collectionDuration* de la méta-classe *Collaboration* dans le cas d'un *low involvement* et à la somme de *collectionDuration* et *evaluationDuration* dans le cas d'un *high involvement*. La collaboration échoue si la durée fixée arrive à échéance sans que les décideurs aient formulé leurs évaluations individuelles.

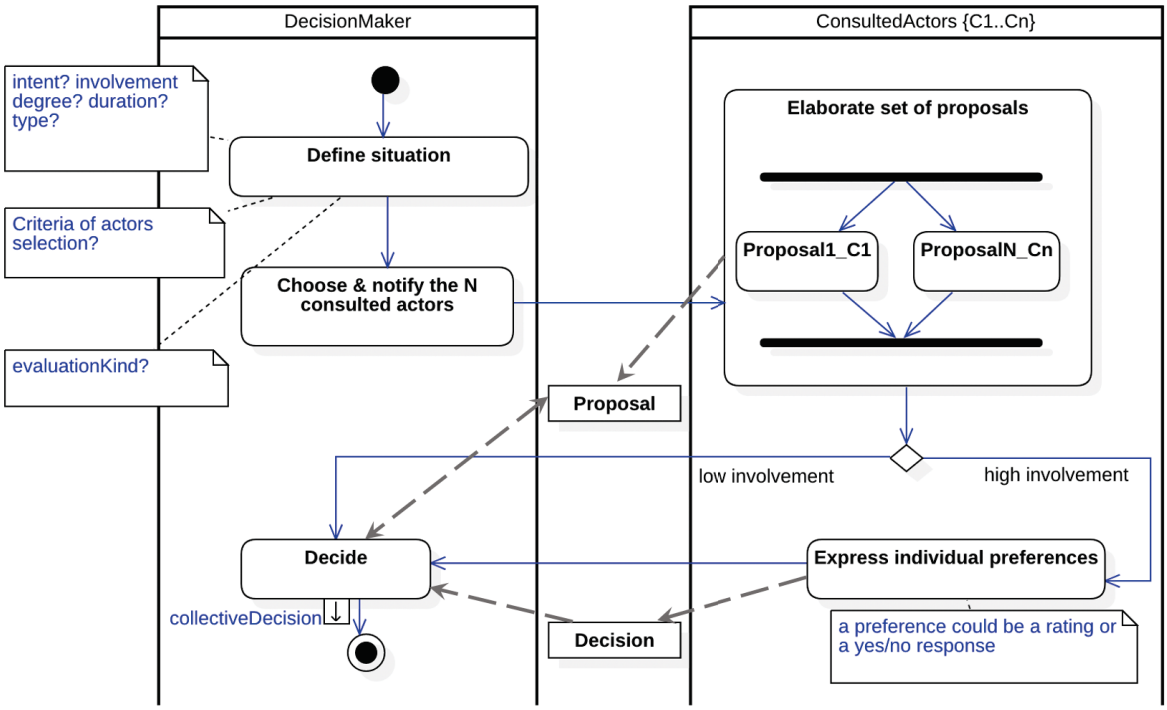


FIGURE 4.19 – Diagramme d'activités de la politique « *Taking Advice* ».

Taking Advice étant une politique de décision à un seul tour, dans le cas d'un *high involvement*, elle suit un processus de vote direct dans lequel les personnes à consulter indiquent leurs préférences. Le *decisionMaker* est alors amené à fixer les autres caractéristiques de la *DecisionPolicy* qui ne sont pas déjà figées, à savoir le type de participation (*type*) et, pour la méthode de co-décision, la méthode d'expression des préférences individuelles (*evaluationKind : rating ou yesNo*).

En cas de participation restrictive (*Restricted Taking Advice*), le décideur doit préciser les critères de sélection des acteurs à consulter lors de la configuration de la politique de décision (*criteria of actors selection* : *availability*?, *expertiseDomain*?, *involvement*?).

Le seuil d'agrément n'est pas utilisé dans cette politique de décision puisqu'il y a un seul décideur.

Utilisations connues

— Cette politique de décision est utilisée par exemple dans les Judge-Advisor Systems (JAS) [SNIEZEK et BUCKLEY, 1995]. Les deux rôles dans un JAS sont ceux de juge et de conseiller. Le juge est le décideur qui évalue les informations concernant une décision particulière et rend le jugement final sur le résultat de cette décision. Le conseiller est une personne qui fournit des conseils, des informations ou des suggestions au juge. Le juge et le conseiller peuvent avoir une relation d'ordre hiérarchique ou bien être au même niveau (une personne recevant les recommandations d'un collègue).

— Dans l'approche AHM [EL HAMLAOUI et al., 2014, 2018] (voir section 2.3.2), l'expert peut être considéré comme seul décideur. Il consulte les concepteurs métier pour qu'ils lui proposent les correspondances de niveau méta-modèle utiles pour le système étudié.

4.4.3.2 Application de « Taking Advice » à l'exemple « Définir des Règles de Gestion »

Dans la situation de collaboration présentée dans la section 4.4.1, Bob est le modérateur, il choisit d'adopter la politique de décision ***Restricted Taking Advice*** en impliquant uniquement Claire. Ainsi, cette dernière joue le rôle de *consultedActors* et Bob est le seul décideur (*decisionMaker*).
Les propositions de règles de gestion étant déjà établies, Claire évalue RGa_1, RGa_2 et RGj_1.
Les décisions de Claire n'influencent pas la décision collective par proposition puisque dans le cas d'une politique *Taking Advice*, les évaluations individuelles des personnes consultées ne sont pas agrégées mais constituent uniquement des avis pour que le décideur prenne une décision par rapport à chaque proposition.
La Figure 4.20 illustre une partie de la collaboration. Claire n'est pas représentée comme un décideur puisque ses évaluations ne comptent pas dans les décisions collectives. Le décideur (Bob) n'est pas obligé de suivre les recommandations de Claire ; cette dernière a rejeté, par exemple, la proposition RGa_2 mais au final Bob l'a approuvée. On peut supposer que le commentaire que Claire a associé à sa décision pour RGa_2 n'a pas convaincu Bob. Pour la proposition RGj_1, Bob a suivi la recommandation de Claire et a rejeté la proposition.

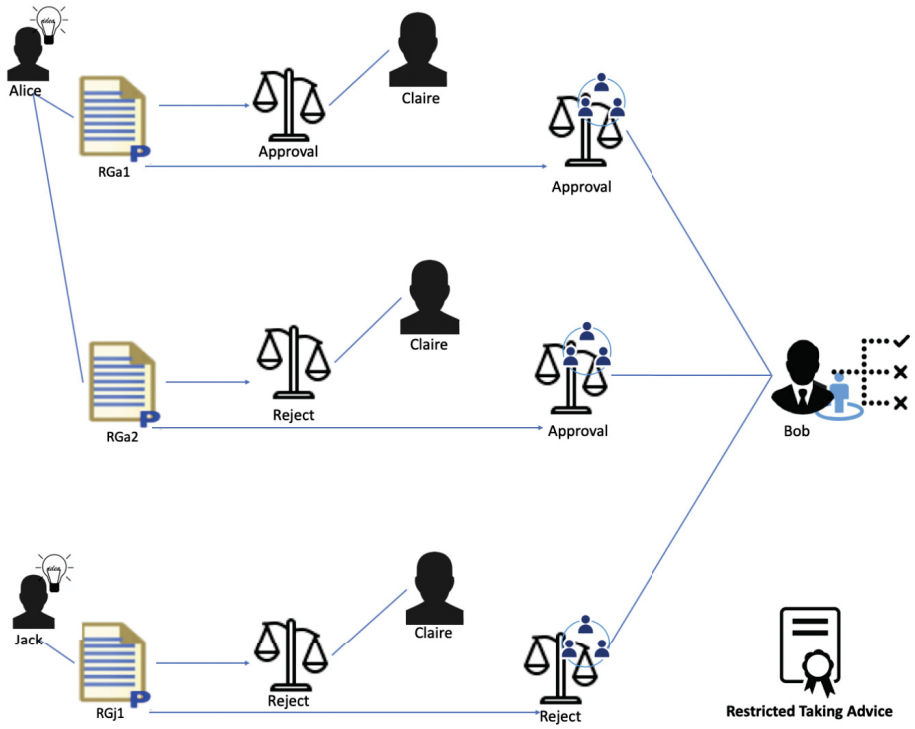


FIGURE 4.20 – Application de la politique *Restricted Taking Advice* à l'exemple Définir RGs.

4.4.4 Politique de décision « Majority Deciding »

4.4.4.1 Description de « Majority Deciding »

Nom *Majority deciding*

Objectif Obtenir une décision démocratique qui prenne en compte les avis de tous les collaborateurs : La (les) proposition(s) approuvée(s) par la majorité du groupe est (sont) adoptée(s).

Applications Cette politique de décision est à utiliser dans les cas de :

- Répartition homogène des compétences et du poids décisionnel entre les membres du groupe.
- Contraintes de temps : par rapport aux méthodes à plusieurs tours, la politique *Majority deciding* nécessite généralement moins de temps. En effet, elle est mise en oeuvre par

un vote à un seul tour et permet donc d'aboutir - théoriquement - plus rapidement à une décision.

Solution La mise en place de cette politique de décision passe par cinq activités comme décrit par le diagramme d'activités suivant (Figure 4.21).

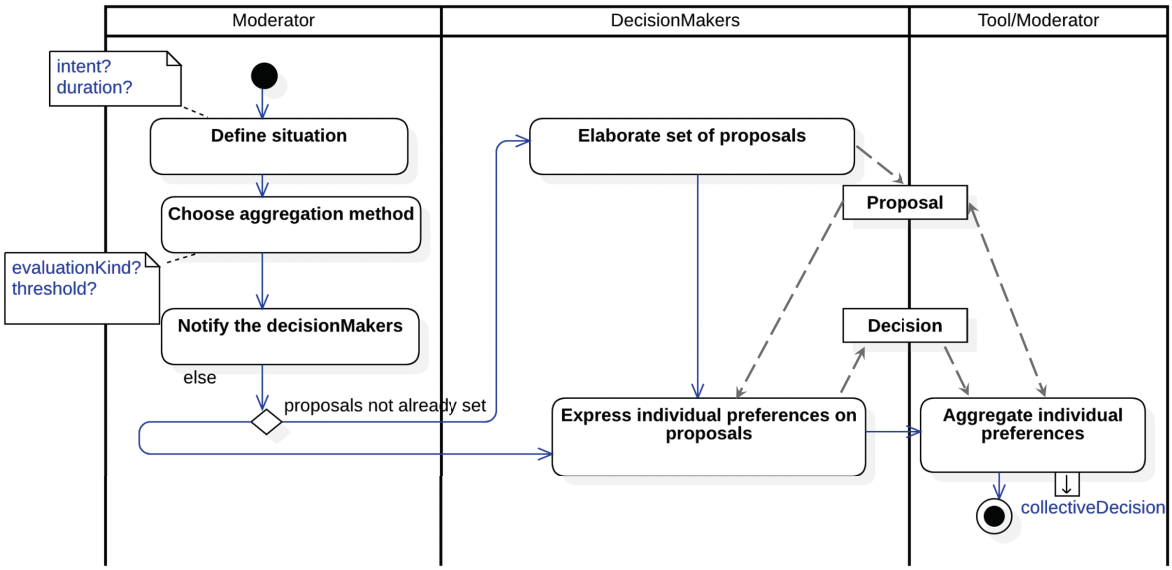


FIGURE 4.21 – Diagramme d'activités de la politique « *Majority deciding* ».

D'abord, le modérateur définit la situation de collaboration (objectif (*intent*), durée (*duration*)). Ensuite, il fixe les caractéristiques *threshold* et *evaluationKind* de la méthode de co-décision (le *processKind* étant fixé à *directVote*). Après, il notifie les acteurs à qui il attribue le rôle de décideur (*decisionMakers*). Si les propositions ne sont pas déjà établies, les décideurs commencent par élaborer la liste des propositions. Ensuite, ils expriment leurs évaluations individuelles pour chaque proposition. A la fin de cette étape, un outil (ou éventuellement le modérateur) agrège les évaluations individuelles. Les propositions dont les acceptations [3] dépassent le seuil d'agrément prédéfini sont approuvées et constituent la décision collective dans la mesure où elles ne sont pas conflictuelles.

L'activité « *Aggregate individual preferences* » peut réussir ou échouer selon les acceptations des propositions. Nous donnons ci-dessous deux exemples de scénario de cette activité. Soient A,B et C, trois propositions.

cas 1 : Deux propositions au moins sont conflictuelles. Dans cet exemple on considère que seules A et B sont conflictuelles.

cas 1.1 : Les acceptations de A, B et C dépassent le seuil d'agrément prédéfini, avec : Acceptation(A) > Acceptation(B) :

- ➜ A est approuvée par le groupe.
- ➜ B est rejetée par le groupe.
- ➜ C est approuvée par le groupe.

cas 2 : Propositions non conflictuelles.

cas 2.1 : Les acceptations des propositions ne dépassent pas le seuil d'agrément prédéfini. :

- ➜ Échec. Sauf si le groupe se met d'accord sur le fait de réajuster le seuil pour conclure la collaboration sans devoir recommencer le processus d'évaluation.

3. Rappel : acceptation(A) est l'agrégation des évaluations individuelles de la proposition A ; une évaluation de type approbation est traduite par 1, une évaluation de type refus est traduite par 0.

Utilisations connues

— Les élections à un seul tour, qu'elles se déroulent en présentiel ou par vote électronique.

— Dans l'approche d'alignement collaboratif de modèles (CAHM) que nous présentons dans le chapitre 5, l'élaboration des relations sémantiques de type DSR peut être effectuée selon cette politique de décision.

Politiques de décision connexes *Delegated Voting*. *Majority deciding* et *Delegated Voting* diffèrent dans le type de participation. *Delegated Voting* fait un choix préalable des personnes à impliquer dans la prise de décision, alors que *Majority deciding* fait participer tout le monde.

4.4.4.2 Application de « Majority Deciding » à l'exemple « Définir des Règles de Gestion »

Bob étant le modérateur, il choisit d'adopter la politique de décision ***Majority deciding***, avec un seuil posé à *high* (i.e., agrément entre 67% et 99%). Ainsi, chaque décideur évalue les propositions des autres. Les évaluations individuelles - instances du concept *Decision* de MMCollab - consistent à faire un *approval*, un *reject* ou un *refinement*.
La Figure 4.22 illustre une partie de la collaboration qui correspond aux décisions (évaluations individuelles) et à l'obtention de la décision collective sur la proposition RGa_1. Cette proposition a été initiée par Alice. Ainsi elle est évaluée par le reste des acteurs à savoir Bob, Jack et Claire. La décision collective pour cette proposition dépend des décisions individuelles de tous les décideurs. Deux des trois décideurs ayant rejeté la proposition, elle est rejetée par le groupe.

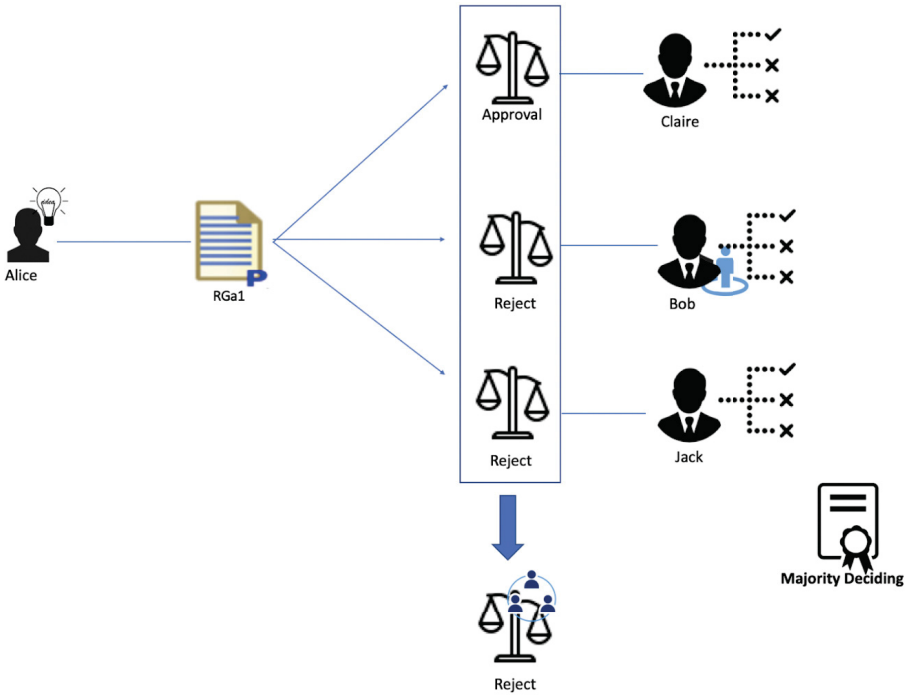


FIGURE 4.22 – Évaluation de la proposition RGa_1 par 3 décideurs selon la politique *Majority Deciding*.

4.4.5 Politique de décision « Consenting Together »

4.4.5.1 Description de « Consenting Together »

Nom *Consenting Together*.

Objectif Obtenir une solution qui satisfasse tous les membres du groupe.

Applications Cette politique de décision est à utiliser dans les cas de :

- Groupe de taille réduite avec, au sein du groupe, des intérêts convergents (ou du moins non opposés).
- Répartition homogène des compétences décisionnelles au sein du groupe.
- Pas de contraintes de temps puisque le processus risque de durer longtemps.
- Membres du groupe non réticents à être impliqués.

Solution La solution exposée ici permet de simuler le déroulement d'un processus consensuel en présentiel ou en distribué.

Le déroulement de cette politique de décision requiert la présence de trois rôles : le modérateur, l'ensemble des décideurs et l'agrégateur (outil ou modérateur) comme résumé sur la Figure 4.23 ci-dessous.

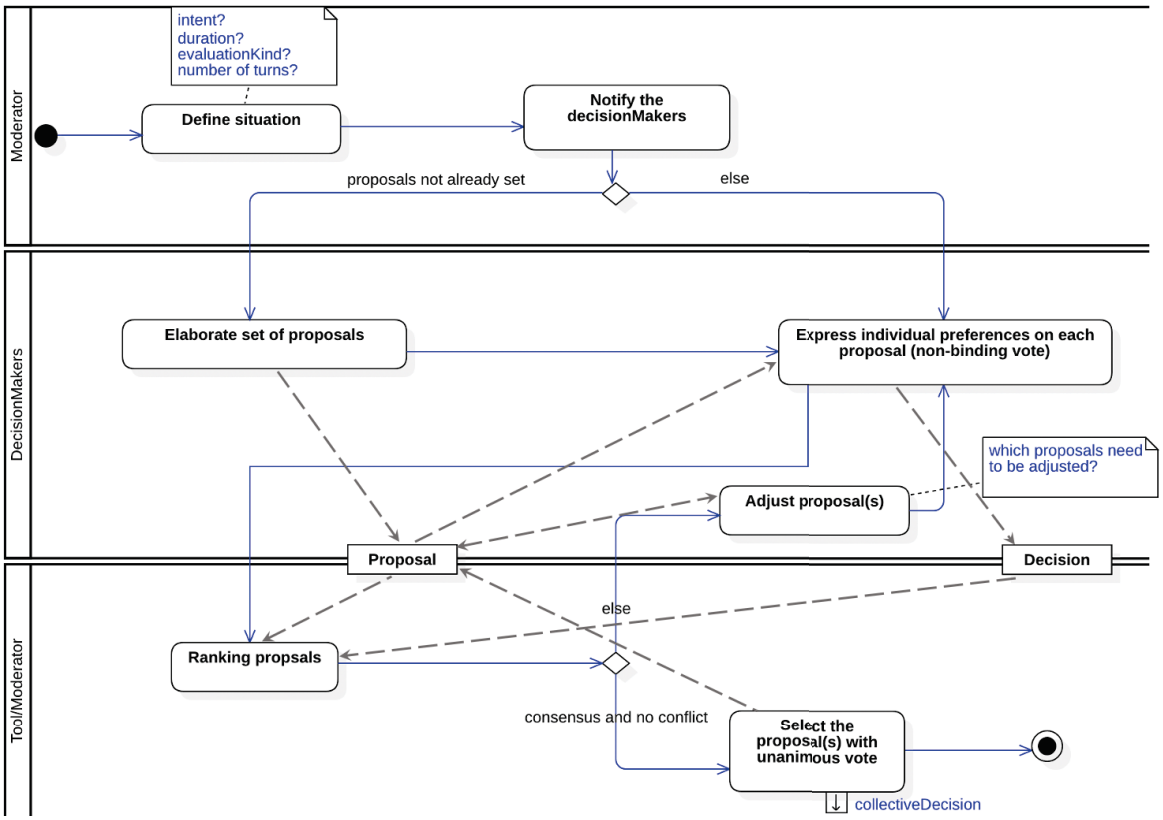


FIGURE 4.23 – Diagramme d'activités de la politique « *Consenting Together* ».

D'abord, le modérateur définit la situation de collaboration (objectif (*intent*), durée (*duration*) qui correspond à la somme des attributs *collectionDuration* et *evaluationDuration*), l'*evaluationKind* de la méthode de co-décision et le nombre de tours autorisés avant de clore la collaboration ; le *processKind* étant fixé à *consensus2vote* et le *threshold* fixé à *strict*. Ensuite, il notifie les personnes à qui il attribue le rôle de décideur (*decisionMakers*).

Les *decisionMakers* doivent commencer par produire la liste des propositions si elles ne sont pas déjà établies, puis exprimer leurs choix individuels sur ces propositions par un vote indicatif.

Un outil (*tool*) (ou éventuellement le modérateur) trie par la suite ces votes. Si des propositions sont approuvées à l'unanimité et sont non conflictuelles, elles sont acceptées, sinon les membres du groupe revoient certaines propositions en les affinant jusqu'à ce qu'elles satisfassent l'ensemble des membres.

Le processus réussit si l'unanimité est atteinte pour une ou plusieurs propositions après un ou plusieurs tours sans dépasser le nombre de tours autorisé et la durée maximale fixée pour la collaboration. Sinon la collaboration échoue et le modérateur choisit soit de changer de politique, soit de réinitialiser la session de collaboration.

Utilisations connues

— Les réunions en présentiel impliquant un ensemble homogène de décisionnaires. On peut citer par exemple la réunion d'une commission d'attribution de bourses d'étudiants, sans vote, procédant par recherche de consensus.

— Dans l'approche d'alignement collaboratif de modèles (CAHM) présenté dans le chapitre 5, l'élaboration des méta-correspondances peut exiger une politique *Consenting Together* pour s'assurer que les méta-correspondances sont approuvées par l'ensemble des collaborateurs et réduire ainsi les risques de retours en arrière après la génération des correspondances au niveau modèle.

Politiques de décision connexes *Negotiating together*. *Consenting Together* et *Negotiating Together* sont des politiques de décision itératives (i.e. à plusieurs tours). *ConsentingTogether* requiert un seuil d'agrément strict de 100% tandis que *NegotiatingTogether* fonctionne avec des valeurs de seuil moyennes ou élevées.

4.4.5.2 Application de « Consenting Together » à l'exemple « Définir des Règles de Gestion »

Sur la Figure 4.24, nous considérons la proposition RGa_1. Lors du premier tour, elle a été évaluée de la même façon que par la politique *Majority Deciding*, i.e., une approbation de Claire et des rejets de Bob et Jack.
Étant donné que *Consenting Together* est itérative, que le consensus n'a pas été atteint et que le nombre de tours autorisé par le modérateur a été fixé à trois, les acteurs sont amenés à affiner la proposition. Sur la Figure 4.24 (deuxième tour), nous considérons que Claire initie une RGa_1c_1, proposition alternative à RGa_1 et en conflit avec cette dernière. RGa_1c_1 doit prendre en compte théoriquement les commentaires faits par les décideurs qui ont rejeté RGa_1 (pour espérer un changement des évaluations individuelles).
RGa_1c_1 doit aussi être évaluée par les décideurs, qui sont, dans ce cas, Alice, Bob et Jack (puisque Claire est l'initiateur de RGa_1c_1). Les évaluations sont concluantes pour RGa_1c_1 (unanimité), l'application de la politique est réussie et la proposition acceptée.

4.4.6 Politique de décision « Negotiating together »

4.4.6.1 Description de « Negotiating together »

Nom *Negotiating together*.

Objectif Obtenir une solution qui satisfasse la plupart des membres du groupe. Ceci demande que tous les membres du groupe affinent le cas échéant leur décision jusqu'à ce qu'un seuil d'agrément prédéfini soit atteint.

Applications Cette politique de décision est à utiliser dans les cas de :

— Répartition homogène du poids décisionnel et des compétences au sein du groupe.

— Intérêts convergents au sein du groupe.

— Possibilité d'acceptation d'une décision sans consensus (décision à impact réversible sans grand coût).

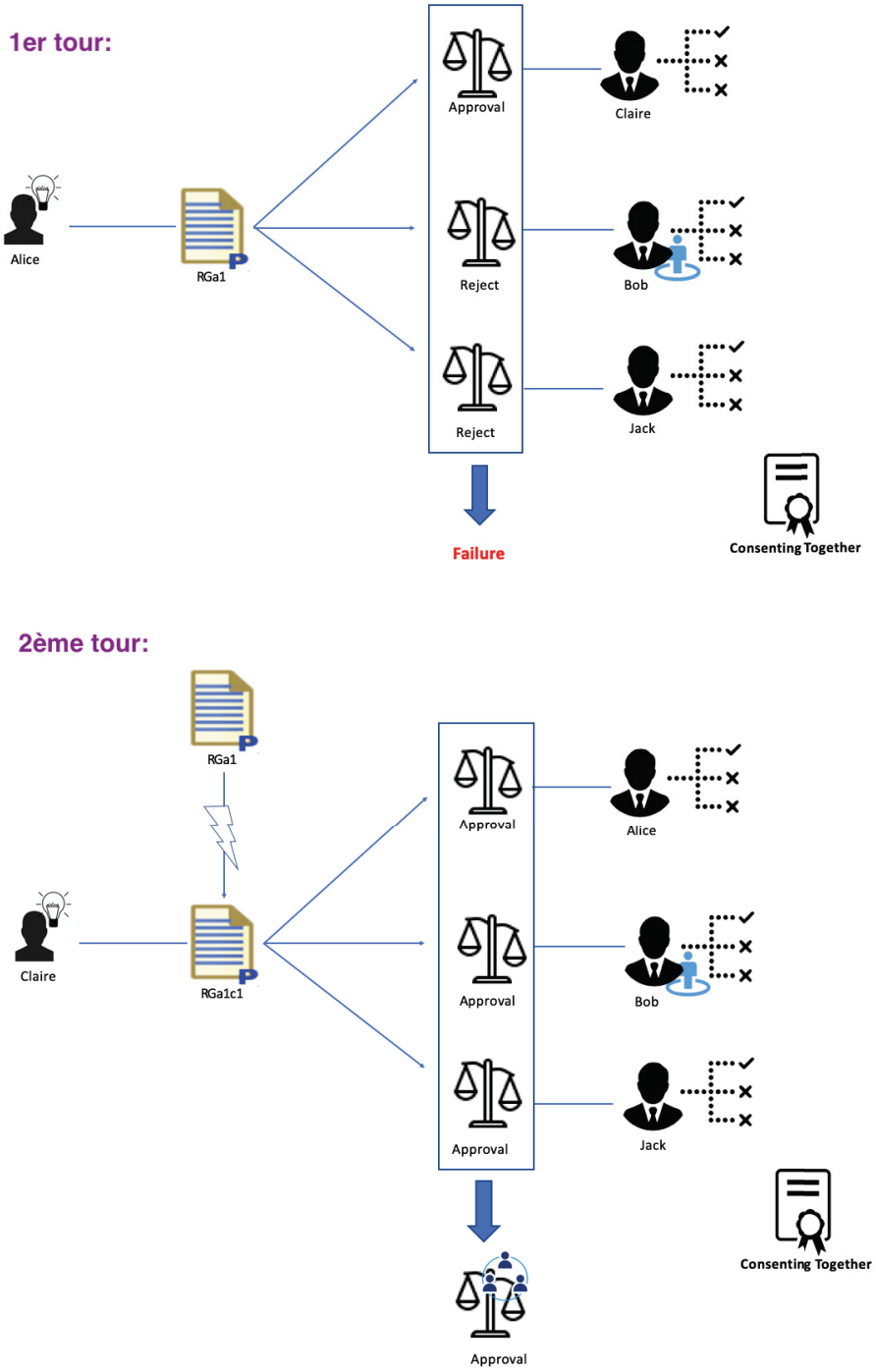


FIGURE 4.24 – Application de la politique *Consenting Together* à l'exemple Définir RGs.

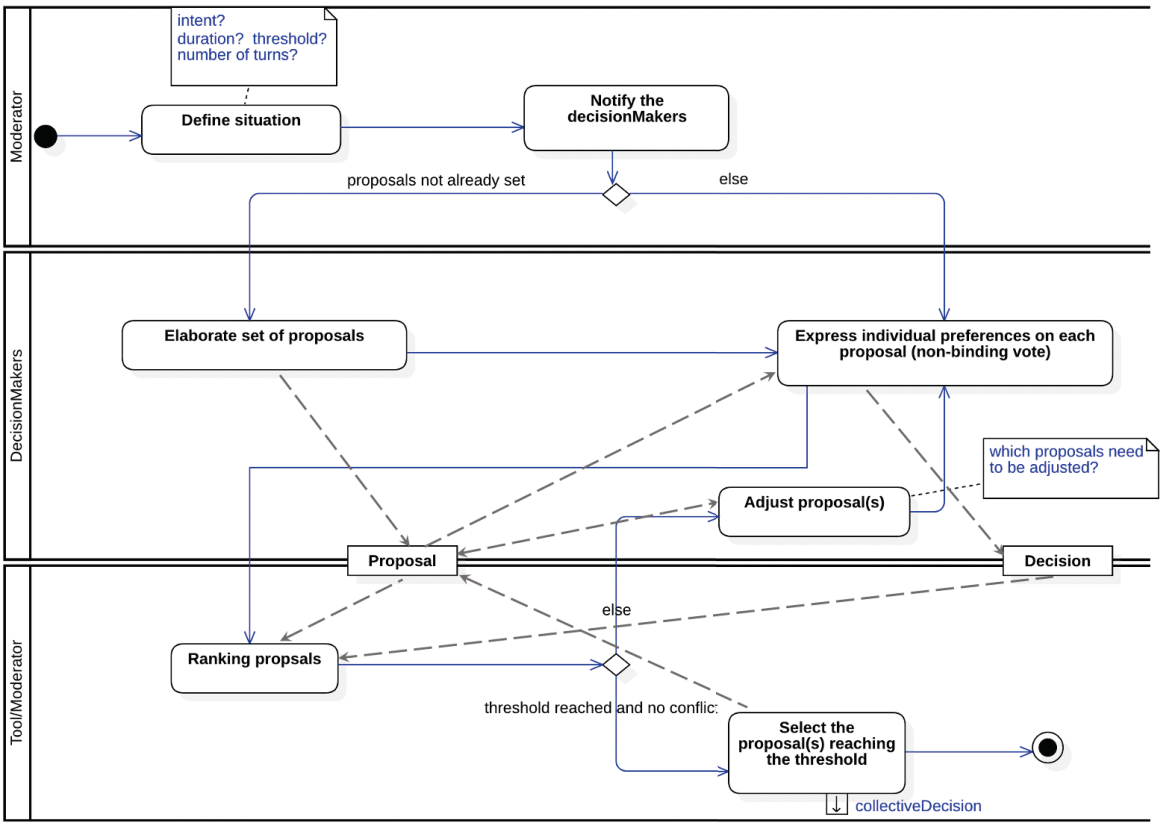


FIGURE 4.25 – Diagramme d'activités de la politique « *Negotiating together* ».

Solution La Figure 4.25 résume le déroulement de cette politique. Trois rôles sont requis :

1. Un modérateur (*moderator*) qui définit la situation en fixant l'objectif de la prise de décision (*intent*), le seuil d'agrément visé (*threshold*), la durée maximale du processus (*duration*), le nombre de tours autorisé (*number of turns*) et notifie les personnes concernées.
2. Des décideurs (*decisionMakers*) qui doivent d'abord produire la liste des propositions si elles ne sont pas déjà établies, puis renseigner leurs évaluations individuelles sur ces propositions par un vote indicatif.
3. Un outil (*tool*) qui trie par la suite ces votes (*ranking proposals*).

Si des propositions non conflictuelles dépassent le seuil d'agrément prédéfini, elles sont sélectionnées, sinon les membres du groupe affinent les propositions (toutes les propositions ou seulement celles qu'ils trouvent intéressantes).

Le processus réussit si le seuil d'agrément choisi est atteint pour au moins une proposition. Sinon la collaboration échoue et le modérateur choisit soit de changer de politique de décision, soit de réinitialiser la session de collaboration.

Utilisations connues

— Les réunions en présentiel impliquant un ensemble homogène de décisionnaires et ne nécessitant pas une approbation consensuelle. Les négociations gouvernement-syndicats par exemple.

Politiques de décision connexes *Consenting Together* et *Delegated Negotiating*.

Negotiating Together, *Consenting Together* et *Delegated Negotiating* sont des politiques de décision itératives (i.e. à plusieurs tours). La différence est que d'une part *Consenting Together* suit un processus de consensus suivi de vote où une décision unanime est exigée, alors que *Delegated Negotiating* et *Negotiating Together* suivent un processus de négociation suivie de

vote (agrément<100%). D'autre part, *Negotiating Together* est une politique démocratique alors que *Delegated Negotiating* est restrictive.

4.4.6.2 Application de « Negotiating together » à l'exemple « Définir des Règles de Gestion »

L'application de cette politique de décision à la situation de collaboration définie dans la section 4.4.1. consiste en ce que les décideurs fassent des votes indicatifs sur toutes les propositions afin de voir lesquelles de ces dernières dépassent le seuil posé. Nous considérons dans cet exemple que ce seuil a été fixé à *medium* (i.e., acceptation entre 50% et 66%).
Sur la Figure 4.26, nous considérons la proposition RGj_1 qui a reçu une approbation de Bob et Claire, et un rejet d'Alice. Alice a alors proposé RGj_1a_1 qui n'est pas en conflict avec RGj_1.
En appliquant la politique *Negotiating together*, la décision collective est un *approval* de la proposition RGj_1 dès le premier tour même si elle a reçu un rejet de Alice, car son acceptation dépasse le seuil posé et que RGj_1 et RGj_1a_1 ne sont pas en conflit. RGj_1a_1 est aussi approuvée puisqu'elle a reçu trois *approval* une fois évaluée par Jack, Claire et Bob dans le deuxième tour.

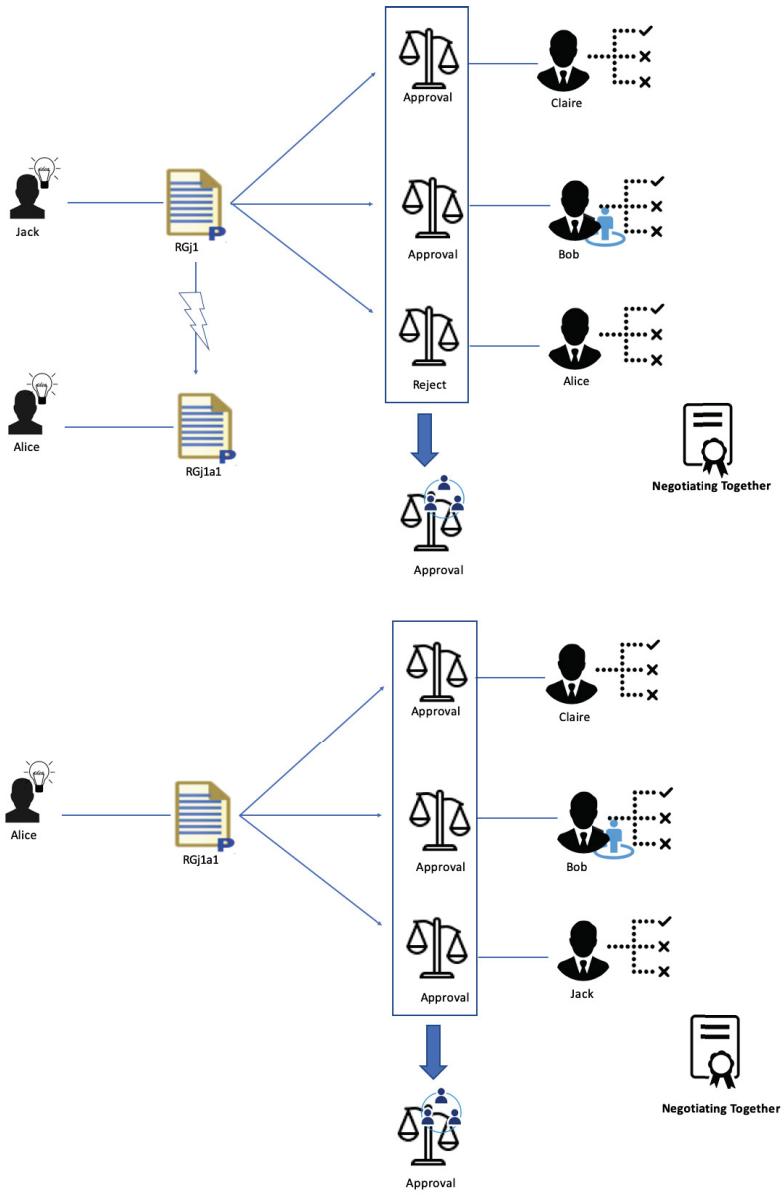


FIGURE 4.26 – Application de la politique *Negotiating together* à l'exemple Définir RGs.

4.4.7 Politique de décision « Delegating »

4.4.7.1 Description de « Delegating »

Nom *Delegating*. *Delegating* possède deux spécialisations concrètes *Delegated Voting* et *Delegated Negotiating*.

Objectif Impliquer un sous-ensemble des membres du groupe satisfaisant certains critères (disponibilité, niveau d'expertise, implication). La politique *Delegated Voting* suit un processus de vote direct, tandis que *Delegated Negotiating* suit un processus de négociation suivie de vote.

Applications Cette politique de décision est pertinente lorsque le temps disponible pour la prise de décision est critique (*Delegated Voting*) ou quand un sous-ensemble seulement des membres du groupe dispose de la connaissance pour prendre cette décision (*Delegated Negotiating*). Dans ces cas, le modérateur de la collaboration réduit la taille du groupe concerné pour fluidifier le processus de prise de décision. Il délègue ainsi la prise de décision aux acteurs sélectionnés.

Solution La Figure 4.27 résume la mise en oeuvre de cette politique de décision.

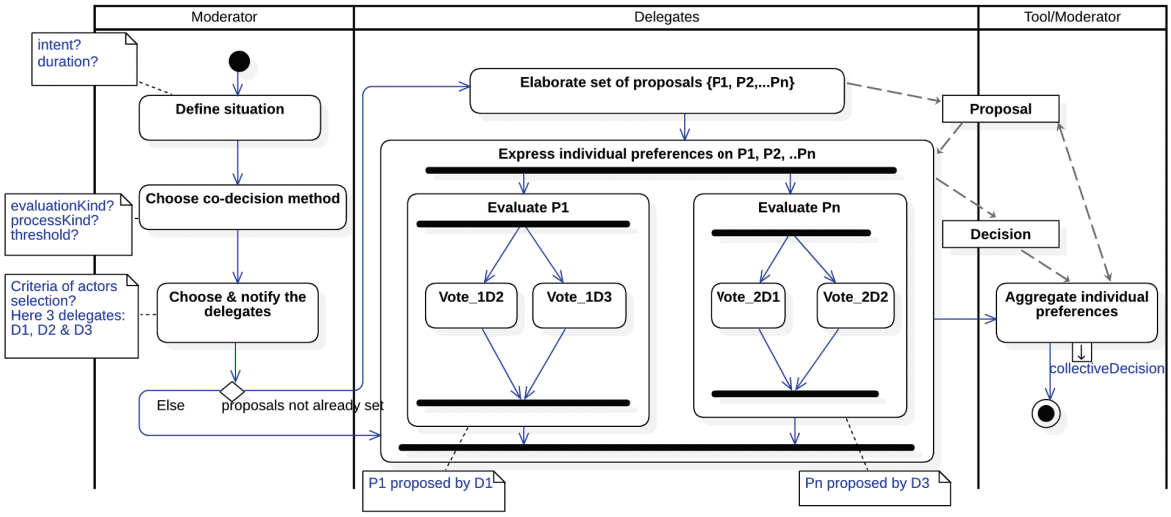


FIGURE 4.27 – Diagramme d'activités de la politique « *Delegated Voting* ».

Trois rôles sont impliqués :

1. Un modérateur (*Moderator*) définit la situation de la collaboration : le sujet, l'objectif de la prise de décision (*intent*) et la durée (*duration*). Le type de participation pour *Delegating* étant figé (*restricted*), le modérateur doit préciser les critères de sélection des acteurs de la prise de décision (*criteria of actors selection*).

   Concernant la méthode de co-décision, il doit choisir le moyen d'expression des préférences individuelles et le processus adopté pour leur agrégation (*processKind*) qui est : *negotiation2vote* pour *Delegated Negotiating* et *directVote* pour *Delegated Voting*. Le modérateur fixe aussi le seuil d'agrément visé (*threshold*).

2. Les délégués (*Delegates*) élaborent d'abord la liste des propositions le cas échéant, puis évaluent les propositions des autres. Ainsi, en considérant 3 délégués D1, D2 et D3 et une politique *Delegated Voting* sur la Figure 4.27, la proposition P1 proposée par le délégué D1 est évaluée par les délégués D2 et D3 par les votes $Vote1_{D2}$ et $Vote1_{D3}$ (où $Votei_{Dj}$ est le vote du délégué Dj à propos de la proposition i).

3. Un agrégateur (l'outil ou le modérateur) agrège les préférences individuelles selon la méthode d'agrégation choisie par le modérateur. La proposition (en cas d'alternatives conflictuelles) ou bien les propositions (en cas de possibilités non contradictoires) dépassant le seuil d'agrément prédéfini (*threshold*) sont sélectionnées et constituent

le *CollaborativeWorkProduct* résultant de la collaboration. Quand le seuil d'agrément prédéfini n'est pas atteint, la prise de décision échoue sauf si le modérateur revoit ce seuil ; l'agrégation des évaluations individuelles se base sur le nouveau seuil le cas échéant.

Dans toutes les politiques de décision que nous venons de voir, la présence de l'outil n'est pas indispensable pour l'agrégation ; en effet, le modérateur peut collecter et agréger manuellement les préférences individuelles, mais ceci risque d'être une tâche fastidieuse.

Utilisations connues

— Dans le module *Oracle PeopleSoft HCM : Talent Acquisition Manager*[4], une personne (*delegator*) peut désigner une autre personne (*proxy*) et lui déléguer un ensemble de tâches. La délégation a une durée de validité (*delegation period*). Une fois celle-ci dépassée, les tâches non réalisées par le *proxy* sont remises dans la pile des tâches du *delegator*. Le *delegator* peut aussi retirer (*revoke*) les droits de délégation (Delegated Authority) du *proxy* avant la fin de la d*elegation period*.

— Dans l'alignement des modèles que nous traitons dans notre approche (chapitre 5), une session collaborative d'alignement fait intervenir un ensemble d'utilisateurs (concepteurs des points de vue métiers) qui sont concernés par la production des correspondances. Cependant, seuls les utilisateurs ayant des éléments de modèles impliqués dans une correspondance peuvent l'évaluer.

Politiques de décision connexes

— *Majority Deciding* : Quand *Delegating* est effectuée en un seul tour (i.e., *Delegated Voting*), elle ressemble à la politique *Majority Deciding*, avec comme seule différence le type de participation : *restricted* pour *Delegated Voting* et *democratic* pour *Majority Deciding*.

— *Negotiating Together* : Quand *Delegating* est effectué en plusieurs tours (i.e., *Delegated Negotiating*), elle ressemble à la politique *Negotiating Together*, avec comme différence le type de participation : *restricted* pour *Delegated Negotiating* et *democratic* pour *Negotiating Together*).

4.4.7.2 Application de « Delegating » à l'exemple « Définir des Règles de Gestion »

Le modérateur Bob choisit la politique ***Delegated Voting***, avec un seuil d'agrément fixé à *high* (i.e., entre 67% et 99%) pour mettre en oeuvre la situation de collaboration « Définir des Règles de Gestion ».
La prise de décision est déléguée à Claire et Jack pour RGa_1 et RGa_2 et uniquement à Claire pour RGj_1. Claire approuve RGa_1 et RGa_2 tandis que Jack rejette RGa_1 et approuve RGa_2. La décision collective est un rejet de RGa_1, et une approbation de RGa_2 étant donné que l'agrément a été fixé à *high*. Pour RGj_1, l'évaluation de Claire compte comme décision collective puisqu'elle est la seule décisionnaire.

4. Oracle Corporation, "PeopleSoft Enterprise Human Capital Management", 2006, http ://www.oracle.com/us/products/ applications/peoplesoft-enterprise/hcm/index.html

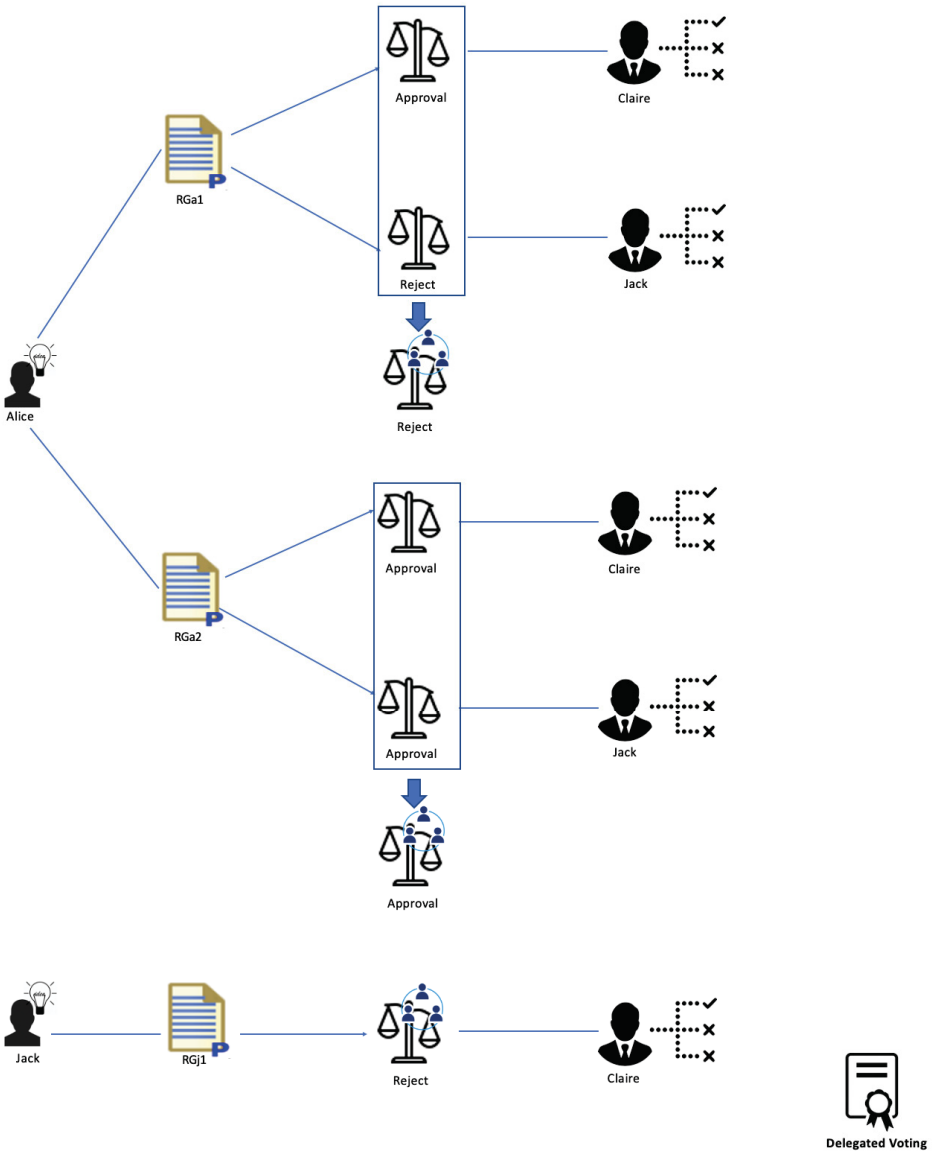


FIGURE 4.28 – Application de la politique *Delegated Voting* à l'exemple Définir RGs.

4.5 Conclusion

Dans ce chapitre nous avons présenté le méta-modèle MMCollab dédié à la modélisation et à la mise en œuvre de processus de prise de décision en groupe.
MMCollab importe certains concepts de SPEM et CMSPEM dans le but d'utiliser un standard de l'OMG et d'éviter de redéfinir des concepts existants dans d'autres langages. Nous avons défini les concepts nécessaires à la modélisation d'un processus de prise de décision en groupe. En particulier, nous pouvons représenter les propositions, leurs évaluations et les méthodes suivies pour l'élaboration des décisions collectives.
Nous avons également défini un ensemble de politiques de décision représentant les pratiques récurrentes d'élaboration de la décision collective lors d'une collaboration. Nous promouvons la réutilisation de ces politiques de décision en permettant leur configuration.
Le tableau 4.2 suivant complète la synthèse des approches de modélisation du GDM de l'état de l'art en incluant notre méta-modèle MMCollab. Il montre en quoi MMCollab répond à la plupart des limitations des approches mises en évidence dans l'état de l'art (chapitre 3).
MMCollab et MADISE se distinguent des autres approches concernant les critères *évolution des propositions* et *adaptabilité des méthodes de GDM*, tandis que MMCollab est le seul à intégrer le critère *sélection des participants*. Pour le critère *évolution des propositions*, MMCollab inclut les concepts *AlternativeProposal* et *Refinement* pour faire évoluer les propositions durant une session de collaboration, en fonction des réflexions et évaluations des membres du groupe. Pour le critère *adaptabilité des méthodes de GDM*, l'instanciation du concept *GDMPattern* permet de définir un ensemble configurable et évolutif de politiques de décision en groupe, adaptables selon les caractéristiques des groupes. Pour le critère *sélection des participants*, MMCollab permet de distinguer les parties prenantes de la prise de décision en incluant des mécanismes (expertise, disponibilité et implication) pour la sélection de sous-ensembles d'acteurs et aussi pour la pondération des préférences des acteurs selon leur expertise.

Dans le chapitre suivant, nous allons utiliser MMCollab pour décrire le processus d'alignement collaboratif de modèles hétérogènes.

	DSO	MADISE	OntoGDSS	Appr. Malavolta	Collaboro	MMCollab
Élaboration des propositions						
Identification des propositions	●	●	●	●	●	●
Evolution des propositions	-	●	-	◐	●	●
Dépendances entre propositions						
Relation de conflit	●	●	●	●	●	●
Relation de spécialisation	-	-	-	●	-	●
Priorisation des participants						
Sélection de participants	-	-	-	-	-	●
Pondération de participants	-	●	-	●	-	●
Expression des préférences						
Critères mesurables	●	●	-	-	-	-
Critères pondérables	●	●	-	◐	-	◐
Méthode décision du groupe						
Adaptabilité	-	●	-	-	◐	●
Outil support						
Complétude	-	●	-	◐	●	◐

● : Critère au centre de l'approche, ◐ : Partiellement considéré, - : Non considéré

TABLEAU 4.2 – Synthèse des caractéristiques supportées par les approches de modélisation de GDM incluant MMCollab.

Chapitre 5

Alignement collaboratif de modèles hétérogènes : CAHM

Sommaire

5.1 Introduction 94
5.2 Exemple fil rouge : Conference Management System (CMS) 94
5.2.1 Contexte 94
5.2.2 Méta-modèles des points de vue du CMS 95
5.2.2.1 Méta-modèle de conception logicielle (méta-modèle SD) 95
5.2.2.2 Méta-modèle de persistance (méta-modèle PS) 95
5.2.2.3 Méta-modèle de processus métier (méta-modèle BP) 96
5.2.3 Modèles des points de vue du CMS 97
5.2.3.1 Modèle de conception logicielle (modèle SD) 97
5.2.3.2 Modèle de persistance (modèle PS) 98
5.2.3.3 Modèle de processus métier (modèle BP) 98
5.3 Présentation globale de l'approche CAHM 99
5.3.1 Vue d'ensemble de CAHM 99
5.3.2 Noyau du Méta-Modèle de Correspondances (MMC) dans CAHM 100
5.3.3 Principe de CAHM - phase 1 101
5.3.4 Principe de CAHM - phase 2 102
5.4 CAHM - phase 1 : Mise en correspondance collaborative 104
5.4.1 Vue d'ensemble du processus de mise en correspondance collaborative . 104
5.4.2 Activité 2 : Étendre MMC 105
5.4.2.1 Description de « Étendre MMC » 105
5.4.2.2 Mise en oeuvre de « Étendre MMC » sur l'exemple fil conducteur CMS 105
5.4.3 Activité 3 : Définir les méta-correspondances 107
5.4.3.1 Description de « Définir les méta-Correspondances » 107
5.4.3.2 Mise en oeuvre de « Définir les méta-Correspondances » sur l'exemple fil conducteur CMS 107
5.4.4 Activité 4 : Générer le modèle de correspondances 110
5.4.4.1 Description de « Générer le modèle de correspondances » 110
5.4.4.2 Mise en oeuvre de « Générer le modèle de correspondances » sur l'exemple fil conducteur CMS 110
5.5 CAHM - phase 2 : Maintien de la cohérence lors des évolutions 111
5.5.1 Évolutions supportées 111
5.5.2 MMC : version étendue pour supporter l'évolution 112
5.5.3 Vue d'ensemble du processus de traitement des évolutions de (méta-)modèles 113
5.5.4 Activité 1 : Détecter les changements 115

5.5.4.1 Description de « Détecter les changements » 115
5.5.4.2 Mise en oeuvre de « Détecter les changements » sur le CMS 115
5.5.5 Activité 2 : Calculer l'impact et l'ordre de traitement des changements . . 115
5.5.5.1 Description de « Calculer l'impact et l'ordre de traitement des changements » 115
5.5.5.2 Mise en oeuvre de « Calculer l'impact et l'ordre de traitement des changements » sur le CMS 116
5.5.6 Activité 3 : Traiter les incohérences 116
5.5.6.1 Description de « Traiter les incohérences » 116
5.5.6.2 Mise en oeuvre de « Traiter les incohérences » sur le CMS 118
5.6 Conclusion 120

5.1 Introduction

Dans le chapitre 2, nous avons présenté une revue de la littérature des approches d'alignement de modèles hétérogènes, tandis que le chapitre 4 a présenté le méta-modèle MMCollab permettant de décrire la prise de décision en groupe.
Dans ce chapitre, nous présentons une approche support à l'alignement collaboratif de modèles de conception hétérogènes appelée Collaborative Alignment of Heterogeneous Models (CAHM). Elle part des principales limitations des approches analysées dans le chapitre 2 et s'appuie sur le méta-modèle MMCollab pour gérer les prises de décision collectives.
L'approche CAHM proposée relève d'une démarche fédérée collaborative où les modèles source sont conservés et reliés par un modèle de correspondances établi par les concepteurs métiers. Ces derniers interagissent et s'appuient sur le Méta-Modèle de Correspondances (MMC), défini dans l'approche AHM [El Hamlaoui, 2015] (voir section 2.3.2), et le méta-modèle MMCollab pour définir les liens inter-modèles et pour les maintenir en cas d'évolution.
La section 5.2 de ce chapitre présente l'exemple fil rouge que nous utilisons pour illustrer et mettre en oeuvre l'approche. La section 5.3 donne un aperçu du processus global de l'approche CAHM qui est composé de deux phases. La section 5.4 décrit la première phase de l'approche, qui vise à établir le modèle de correspondances entre les modèles source et met en oeuvre cette phase sur l'exemple fil rouge. La section 5.5 décrit la deuxième phase de l'approche en l'appliquant également à l'exemple fil rouge ; cette phase permet de maintenir la cohérence inter-modèles et de gérer les impacts des évolutions apportées aux modèles source ou aux points de vue métier. La section 5.6 conclut le chapitre et fait un résumé des contributions de l'approche proposée. Il faut noter que pour ne pas rendre ce chapitre trop long, nous fournissons certains compléments sous la forme de trois sections dans l'annexe B.

5.2 Exemple fil rouge : Conference Management System (CMS)

5.2.1 Contexte

Nous présentons l'exemple fil rouge sur lequel nous allons appliquer et illustrer notre approche. Il s'agit d'un système de gestion de conférence (CMS) conçu pour supporter des fonctions telles que : la gestion scientifique des publications (appel à communications, soumission d'articles, relecture des articles, notification des résultats), les inscriptions (paramétrage, paiement), l'établissement et la mise à jour du planning des communications, la gestion logistique, etc.
Ce type de système étant familier à tous les chercheurs, il ne nécessite pas d'être minutieusement décrit. Par souci de simplification, le CMS est représenté ici selon trois points de vue métier, décrits par trois modèles élaborés séparément par les trois acteurs suivants :

— Architecte logiciel : responsable de la conception logicielle du CMS, il produit un modèle de conception logicielle (Software Design (SD)) exprimé à l'aide d'un méta-modèle spécifique de conception logicielle ;

— Architecte de base de données : responsable de la modélisation et de la sauvegarde des données, il crée un modèle de persistance (PersiStence (PS)) exprimé à l'aide d'un méta-modèle de persistance ;

— Concepteur de procédés : responsable de la conception des activités (tâches) permettant de mettre en oeuvre un CMS, il crée un modèle de processus (Business Process (BP)) exprimé à l'aide d'un méta-modèle de processus métier.

5.2.2 Méta-modèles des points de vue du CMS

Considérés dans un cadre IDM, les modèles représentant les points de vue sur le système étudié sont conformes à des méta-modèles, conformes eux-mêmes au méta-méta-modèle MOF [1]. Nous proposons trois méta-modèles distincts relativement simples mais suffisants pour illustrer l'application de notre approche sur l'exemple CMS.

5.2.2.1 Méta-modèle de conception logicielle (méta-modèle SD)

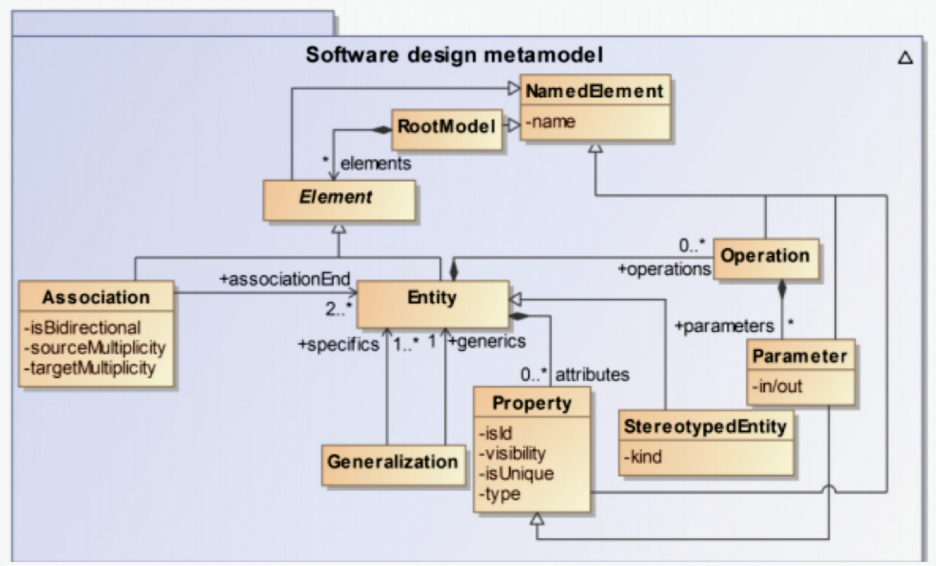


FIGURE 5.1 – Méta-modèle du point de vue conception logicielle (méta-modèle SD).

Le méta-modèle SD (Figure 5.1) est une version simplifiée d'UML [OBJECT MANAGEMENT GROUP (OMG), 2017] dédiée à la description de la conception logicielle. Il définit des entités (*Entity*), avec leurs propriétés (*Property*) et opérations (*Operation*), ainsi que les associations (*Association*) qui existent entre elles.

5.2.2.2 Méta-modèle de persistance (méta-modèle PS)

Le méta-modèle PS (Figure 5.2) regroupe les méta-classes nécessaires pour décrire une base de données relationnelle et les relations entre ces méta-classes. Le schéma de la base de données regroupe des tables (*Table*) et des vues (*View*). Ces dernières contiennent des colonnes (*Column*) qui peuvent être des clés primaires (*PrimaryKey*) ou des clés étrangères (*ForeignKey*).

1. https://www.omg.org/mof/

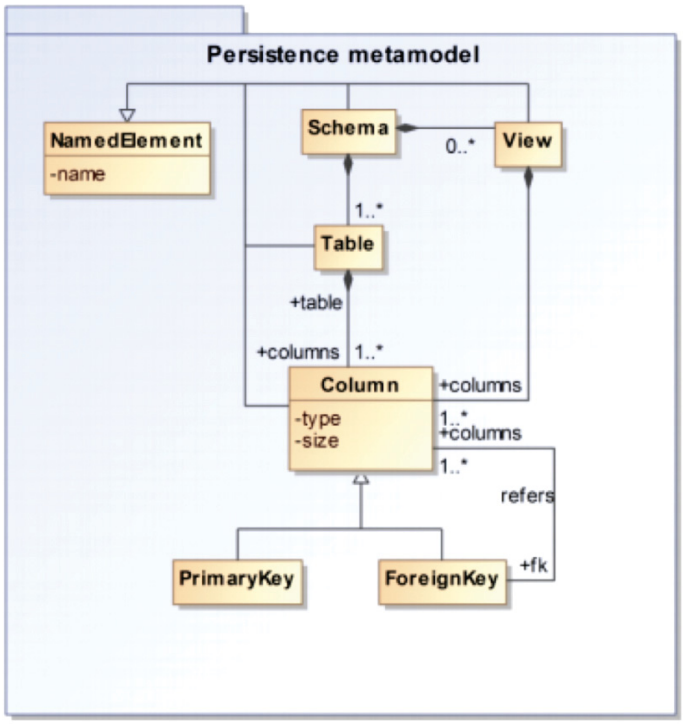


FIGURE 5.2 – Méta-modèle du point de vue persistance (méta-modèle PS).

5.2.2.3 Méta-modèle de processus métier (méta-modèle BP)

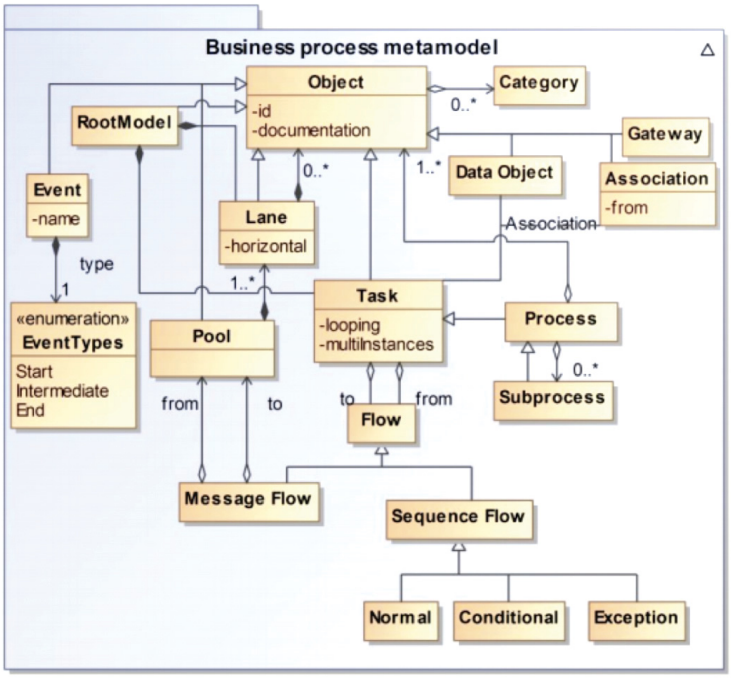


FIGURE 5.3 – Méta-modèle du point de vue processus métier (méta-modèle BP).

Le méta-modèle BP (Figure 5.3) est inspiré de BPMN [2]. Il permet la modélisation et la description de processus (*Process*). Il comprend entre autres les concepts *Pool*, *Lane*, et *Task*. *Pool* décrit la représentation graphique d'un participant au processus. *Lane* est une sous-partition dans un processus (souvent dans un *Pool*). Une tâche (*Task*) est une activité du processus.

2. https://www.omg.org/spec/BPMN/2.0/PDF

5.2.3 Modèles des points de vue du CMS

Dans cette section, nous présentons les trois modèles issus des méta-modèles décrits ci-dessus. Dans ces modèles, pour ne pas alourdir l'exemple, l'accent est mis essentiellement sur la modélisation de la gestion scientifique d'une conférence et plus précisément sur la relecture des articles.

5.2.3.1 Modèle de conception logicielle (modèle SD)

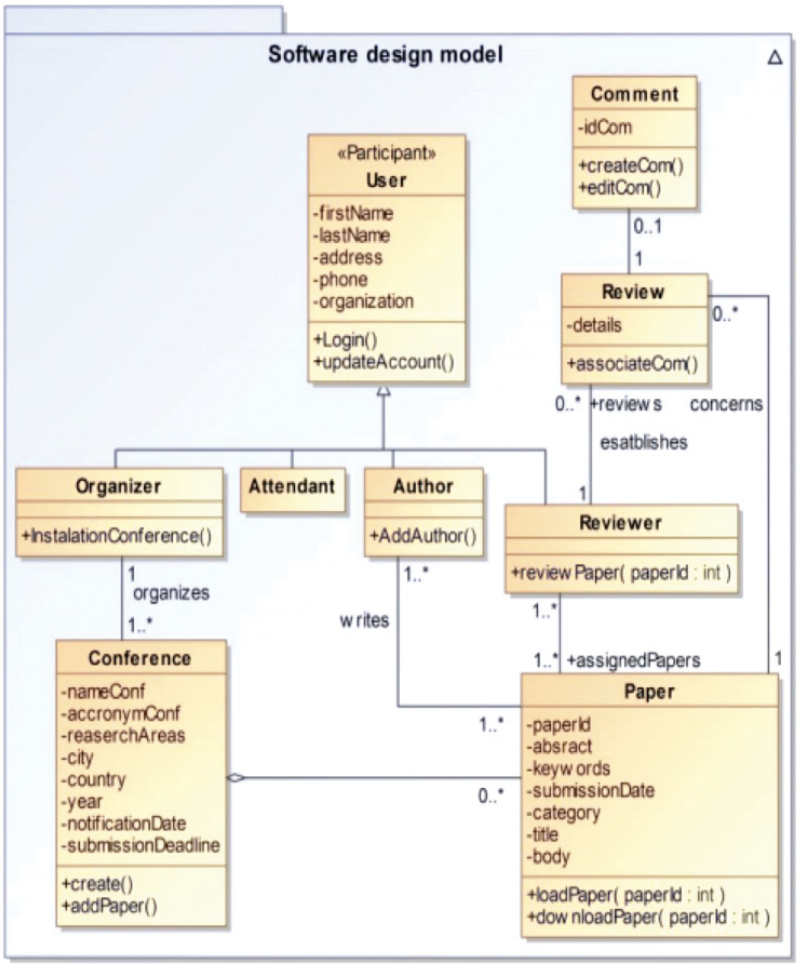


FIGURE 5.4 – Modèle du point de vue conception logicielle (modèle SD) du CMS.

Le modèle SD (Figure 5.4) présente les caractéristiques principales d'un CMS : *Conference*, *Paper*, *Review* et les types d'acteurs impliqués *Organizer*, *Author*, *Reviewer*, et *Attendant*.
La conférence à gérer est décrite par l'intermédiaire de l'entité *Conference*, qui contient différentes propriétés dont le domaine de recherche, la date limite de soumission, la date de notification, le lieu de la conférence, etc.
Au sein d'une conférence sont gérés un ensemble de papiers soumis. Un papier, représenté via l'entité *Paper*, est caractérisé par plusieurs propriétés parmi lesquelles : titre, résumé, mots-clé, contenu, et un identificateur attribué une fois que le papier a été soumis. Un papier peut être rédigé par plusieurs auteurs (d'où l'association *writes*) et il est assigné à plusieurs relecteurs pour évaluation (*assignedPapers*).

5.2.3.2 Modèle de persistance (modèle PS)

Le modèle PS (Figure 5.5) contient les tables permettant d'enregistrer les informations propres à un acteur (*AuthorTable*) telles que son identificateur, le nom de l'organisation à laquelle il est affilié. La table *ArticleReviewsTable* reflète les données concernant l'évaluation d'un article à savoir une colonne *reviews* pour les remarques et critiques, une colonne *decision* pour renseigner l'acceptation ou bien le refus. Dans la même table, une colonne de type clé primaire est utilisée pour énumérer de façon unique les différentes évaluations et une colonne de type clé étrangère est utilisée pour identifier le relecteur.

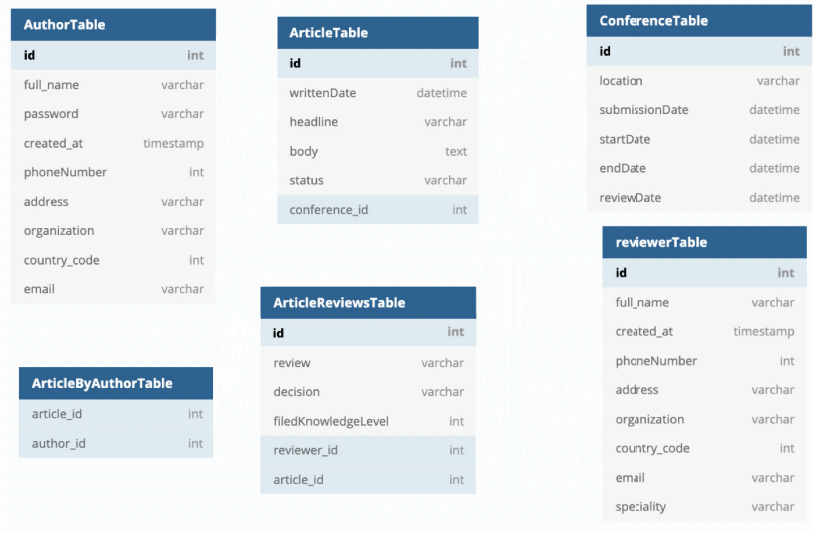


FIGURE 5.5 – Modèle du point de vue persistance (modèle PS) du CMS.

5.2.3.3 Modèle de processus métier (modèle BP)

Pour illustrer le modèle BP (Figure 5.6), nous avons choisi de décrire le processus de relecture. Quand la date de fin de soumission est atteinte, le président de la conférence invite les relecteurs (*reviewer*) à indiquer via le système les articles qu'ils souhaitent relire et leurs éventuels conflits d'intérêt. Une fois le processus de relecture déclenché, le relecteur choisit un papier parmi ceux qu'il a précédemment sélectionnés (*choose paper to review*). À partir de ce moment, il peut soit faire lui-même l'évaluation (*edit review*), soit la sous-traiter à une tierce personne (*outsource paper*). L'évaluation est saisie ensuite dans un formulaire (*review*) dans lequel il peut ajouter des commentaires (*enter comment*).

Notons qu'en réalité les points de vue (modèles et méta-modèles) sont gérés par des équipes, mais nous considérons pour simplifier qu'un seul acteur par point de vue est impliqué dans le processus ; par la suite, nous désignons cet acteur par le terme de coordinateur local (Local Coordinator (LC)). Ainsi, le système CMS implique trois coordinateurs locaux : SD_{LC}, PS_{LC} et BP_{LC} pour respectivement les points de vue logiciel (Software Design), persistance (PersiStence) et processus métier (Business Process).
Le déroulement de cet exemple fil rouge a été réalisé par trois doctorants du laboratoire ADMIR[3]. Chaque doctorant a joué le rôle d'un coordinateur local d'un point de vue métier.

3. http://ensias.um5.ac.ma/article/composition-du-rabat-it-center

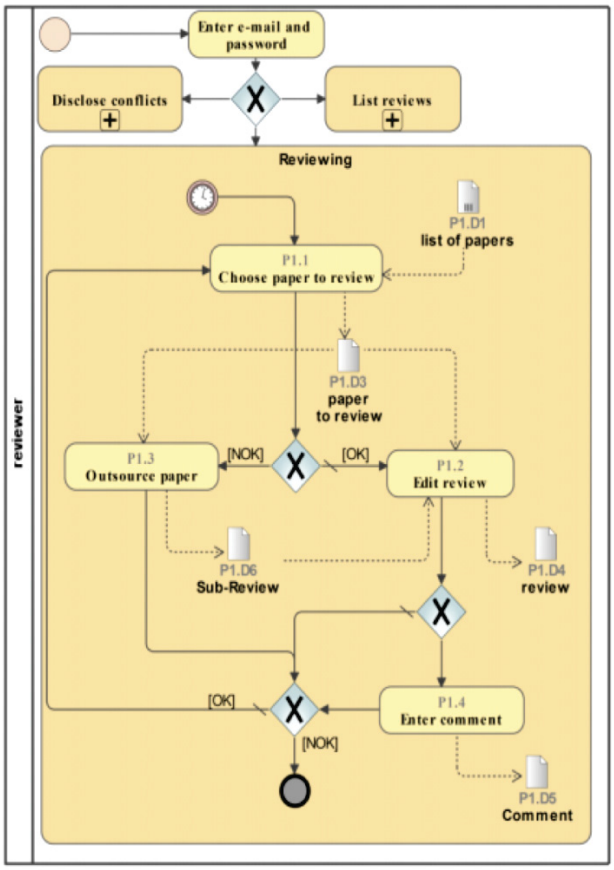


FIGURE 5.6 – Modèle du point de vue processus métier (modèle BP) du CMS.

5.3 Présentation globale de l'approche CAHM

5.3.1 Vue d'ensemble de CAHM

Le but de l'approche CAHM est d'élaborer une vision globale sur les modèles des points de vue métier représentant un système complexe. L'approche propose un processus collaboratif décomposé en deux sous-processus consécutifs :

— Un sous-processus de mise en correspondance dont l'objectif est d'établir des liens (correspondances) entre les modèles des points de vue (par la suite, nous faisons référence à ce sous-processus par le terme CAHM - phase 1) ;

— Un sous-processus pour le maintien de la cohérence en cas d'évolution (changements dans les (méta-)modèles source) dont l'objectif est de maintenir les liens établis en phase 1 (CAHM - phase 2).

La Figure 5.7 présente l'enchaînement global de l'approche sous forme d'un diagramme inspiré de SADT [4] [MARCA et MCGOWAN, 1987]. Le diagramme montre les artefacts en entrée et sortie ainsi que les mécanismes utilisés par l'approche pour aboutir à la vue globale, et les contraintes ou les éléments déclencheurs pour chaque phase.
Les deux phases de l'approche CAHM prennent en entrée à la fois les modèles et les méta-modèles des domaines métier. L'objectif de l'approche est d'élaborer un modèle de correspondances qui regroupe les correspondances inter-modèles.
Pour aboutir à ce modèle de correspondances (artefact en sortie du processus global de l'approche et/ou de CAHM - phase 1 s'il n'y a pas d'évolution), CAHM exploite à la fois les connaissances des acteurs métier et un outil support.

4. SADT : Structured Analysis and Design Techniques

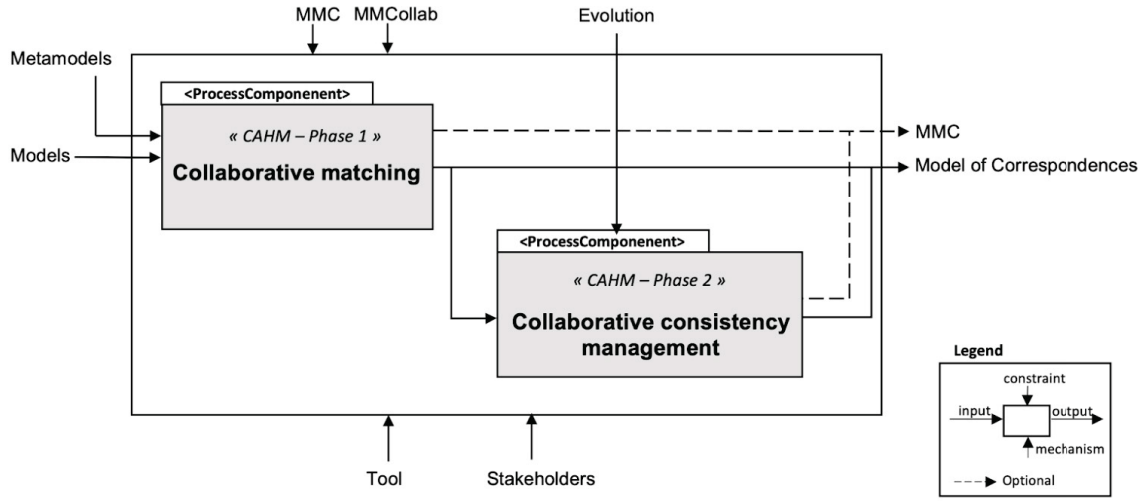


FIGURE 5.7 – Diagramme SADT décrivant le processus global d'alignement de modèles de CAHM.

Les connaissances des acteurs métier et leurs choix sont exploités dans les deux phases pour réaliser les définitions manuelles et les activités de prise de décision. L'outil réalise les tâches automatisables qui ne requièrent pas d'intervention humaine à savoir la génération du modèle des correspondances, la détection des changements des (méta-)modèles et le traitement (en partie) des incohérences.
L'approche s'appuie sur les deux méta-modèles MMCollab (cf. chapitre 4) et MMC, présenté en deux temps : le noyau dans la section 5.3.2 et la version étendue pour supporter l'évolution en 5.5.2. MMCollab est exploité pour dérouler les prises de décision que ce soit au niveau de la phase de mise en correspondance collaborative, ou de celle de maintien de la cohérence. Dans la première phase, MMCollab est instancié pour permettre la prise de décision en groupe à partir des correspondances proposées. Dans la deuxième phase, MMCollab est instancié pour supporter les décisions collectives concernant le maintien de la cohérence suite aux évolutions des modèles source.
MMC est exploité pour définir les correspondances inter-modèles et les maintenir en cas d'évolution.

5.3.2 Noyau du Méta-Modèle de Correspondances (MMC) dans CAHM

MMC définit la structure du Modèle de Correspondances (M1C) qui permet de mettre en correspondance plusieurs modèles source en élaborant des relations typées entre leurs éléments.
La Figure 5.8 présente le noyau de MMC. Un *CorrespondenceModel* est composé d'un ensemble de correspondances. Il est caractérisé par un niveau (*CorrespondenceLevel*) qui correspond au niveau des correspondances qui le composent : modèle pour des correspondances (i.e., des liens au niveau modèle) et méta-modèle pour des méta-correspondances [5].
Chaque correspondance comprend au moins deux extrémités (*RefElement*) issues de modèles différents (*RefModel*) liées par une relation.
Le noyau de MMC comprend quatre relations génériques, indépendantes du domaine d'application (Domain Independant Relationship, DIR) : similarité (*Similarity*), agrégation (*Aggregation*), dépendance (*Dependency*) et généralisation (*Generalization*). Il peut être étendu pour ajouter de nouvelles relations spécifiques à un domaine d'application donné (Domain Specific Relationship, DSR).
Une relation est caractérisée par un nom (*name*) et un attribut de navigabilité de type booléen (*bidirectional*) qui précise si la relation est symétrique ou non. Elle est associée à au moins deux *RefElement* (source *SourceL* et cible *TargetL*).

5. Par abus de langage, nous appelons méta-correspondance, une correspondance intervenant au niveau des méta-modèles.

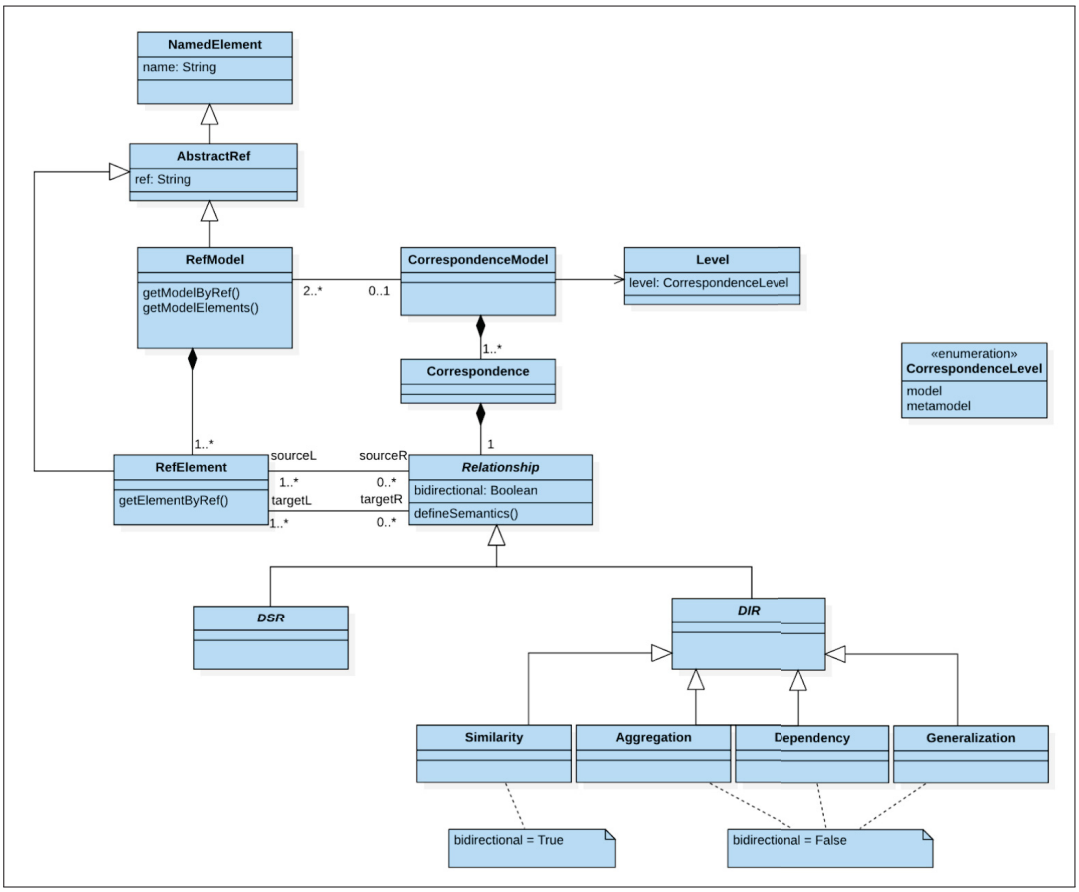


FIGURE 5.8 – Noyau du méta-modèle de correspondances dans CAHM.

RefModel et *RefElement* héritent de *AbstractRef* et contiennent des références qui permettent d'accéder respectivement à un modèle ou à un élément de modèle.
MMC nécessite d'associer une sémantique à chaque relation pour permettre de vérifier sa validité sur un ensemble de (méta-)éléments. En réalité, la sémantique d'une relation précise la logique et la contrainte sous-jacentes. Par exemple, une correspondance utilisant une similarité entre l'élément « a » du modèle « 1 » et l'élément « α » du modèle « 2 » n'est correcte que si la sémantique associée à la relation de similarité est satisfaite entre « a » et « α ». Les sémantiques des relations DIR et DSR utilisées dans ce manuscrit sont décrites dans l'annexe C.

5.3.3 Principe de CAHM - phase 1

La phase de mise en correspondance collaborative (CAHM - phase 1, détaillée dans la section 5.4) consiste en un processus semi-automatique, se déroulant à deux niveaux. Au niveau méta-modélisation, il se base sur la participation des acteurs des domaines métier alors qu'au niveau modélisation, il s'appuie sur un outil implémentant un mécanisme de propagation pour automatiser la production des correspondances.
La Figure 5.9 schématise le principe sur lequel repose cette phase dans les deux niveaux, pour un système représenté par trois points de vue métier (*viewpoints* 1, 2 et 3) et qui sont gérés par trois acteurs différents.
Les coordinateurs locaux participent dans l'étape 1, i.e., la définition des méta-correspondances, alors que l'assistance fournie concerne l'étape 2 où les méta-correspondances sont automatiquement propagées pour générer les correspondances.
La Figure 5.10 illustre un exemple de méta-correspondance (a) et d'une correspondance générée pour le système CMS (b). Nous adoptons la notation *Metamodel :MetaElement* pour désigner un concept de niveau méta-modèle ; par exemple, *SD :Entity* fait référence au concept *Entity* du

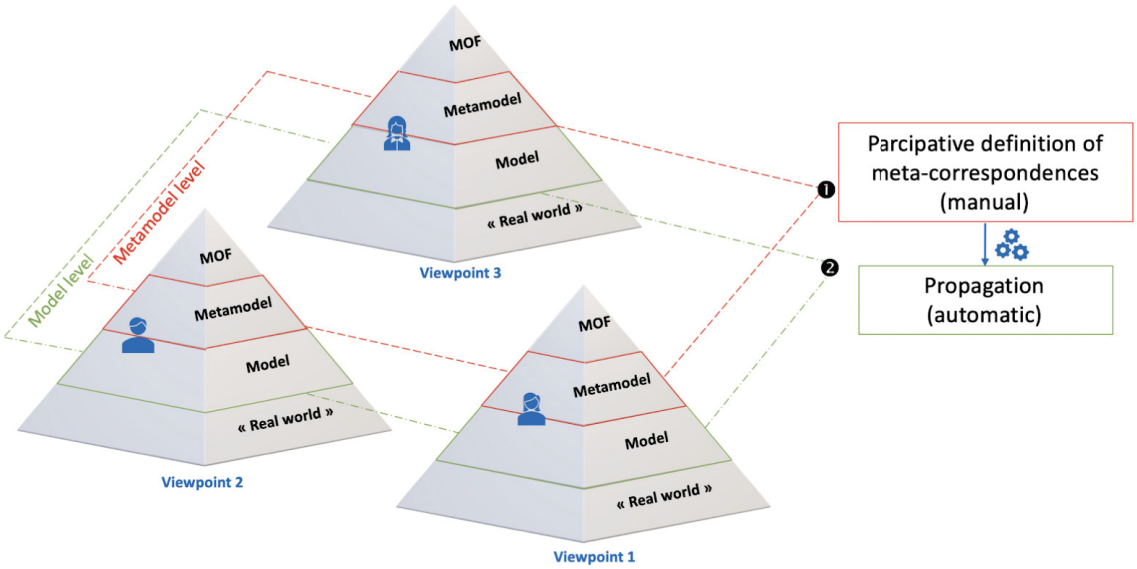
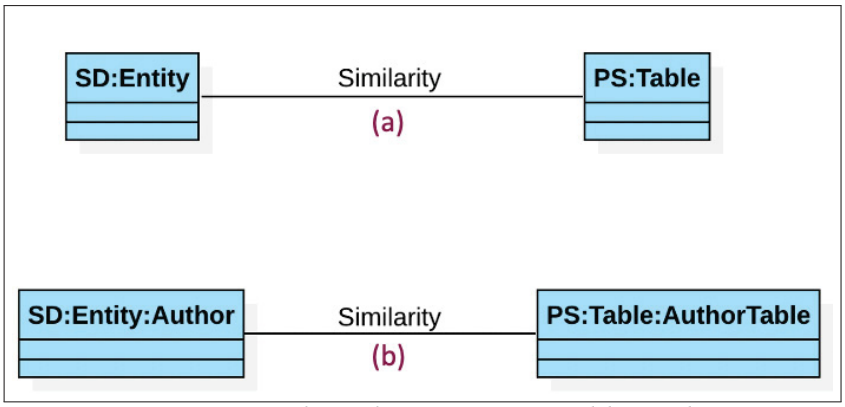


FIGURE 5.9 – Principe de mise en correspondance.

méta-modèle *Software Design* (SD). La méta-correspondance présentée sur la Figure 5.10 définit ainsi une similarité entre le concept *Entity* du méta-modèle *Software Design (SD)* et le concept *Table* du méta-modèle *PersiStence (PS)*. La correspondance définissant une similarité entre la *Table AuthorTable* et l'*Entity Author* est un exemple de propagation de cette méta-correspondance au niveau modèle. Pour identifier un élément appartenant à une correspondance, nous utilisons la notation suivante *Metamodel :MetaElement :Element* (e.g., *SD :Entity :Author*).



(a) : Méta-correspondance, désignée par *Metamodel :MetaElement.*

(b) : Correspondance, désignée par *Metamodel :MetaElement :Element.*

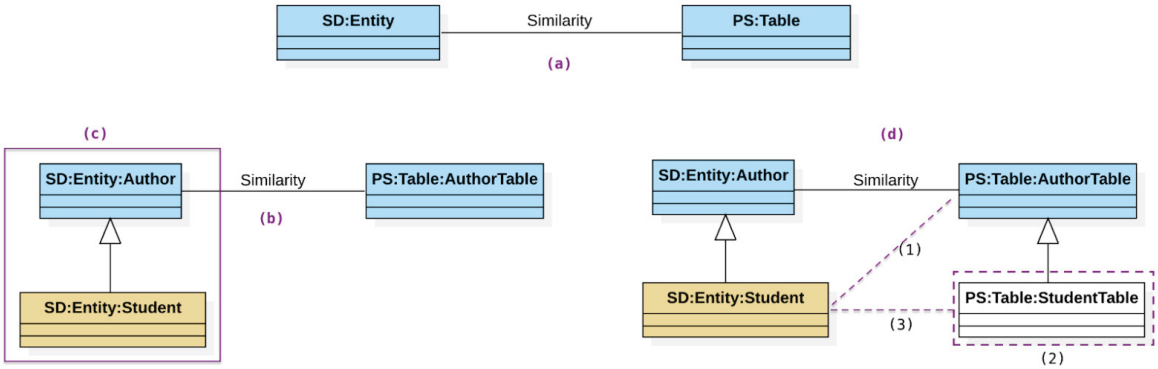
FIGURE 5.10 – Exemple de méta-correspondance et d'une correspondance générée.

5.3.4 Principe de CAHM - phase 2

Les modèles de points de vue ne sont évidemment pas figés et sont modifiés en fonction de différents facteurs : besoins fonctionnels, nouvelles réglementations, contraintes techniques, etc. Ces évolutions de modèles doivent être prises en considération et reflétées sur le modèle de correspondances pour maintenir la cohérence inter-modèles établie dans la phase 1.
La Figure 5.11 présente un exemple d'évolution du système CMS et les répercussions qu'elle pourrait avoir. Une nouvelle *Entity* est ajoutée dans le modèle SD pour refléter un type spécial d'auteur, en l'occurrence *Student* (c). Cette évolution peut avoir des répercussions (d) sur le modèle PS, puisqu'une correspondance à base de *Similarity* relie *SD :Entity :Author* et *PS :Table :AuthorTable* (b).

C'est un cas typique d'évolution qui doit être gérée. Nous souhaitons que la gestion de l'impact des changements soit la plus automatique possible. Néanmoins, certains changements requièrent une prise de décision collaborative. Dans l'exemple de la Figure 5.11, la gestion de l'évolution nécessite l'accord des deux coordinateurs locaux concernés : SD_{LC} et PS_{LC}. Ils devront donc collaborer pour prendre les décisions qui s'imposent, notamment au sujet du modèle PS :

1. Une *Similarity* doit-elle être définie entre *SD :Entity :Student* et *PS :Table :AuthorTable*?
2. Faut-il ajouter une *Table* dérivée de *AuthorTable*?
3. Une *Similarity* doit-elle être définie entre *SD :Entity :Student* et la table dérivée de *AuthorTable* (*PS :Table :StudentTable*) ?



(a) : Méta-correspondance, **(b)** : Correspondance, **(c)** : Évolution du modèle SD, **(d)** : Répercussions possibles

FIGURE 5.11 – Exemple d'évolution de modèle du CMS et ses possibles répercussions sur le modèle de correspondances.

Ainsi le principe de cette phase s'articule autour de trois étapes comme toute autre approche de co-évolution de modèles : (i) Détection des changements, (ii) Classification des changements et calcul de leur impact et (iii) Traitement des changements et de leurs impacts [GRUSCHKO et al., 2007; PAIGE et al., 2016; CICCHETTI et al., 2008b].
Les deux premières étapes sont automatisables. Tandis que l'étape (iii) peut nécessiter l'implication des acteurs métier, et donner lieu à des sessions collaboratives pour traiter les impacts d'un changement. Cette collaboration est nécessaire étant donné que certains changements concernent des éléments impliqués dans le modèle de correspondances, et leur modification risque donc de rompre la cohérence globale du système étudié.

5.4 CAHM - phase 1 : Mise en correspondance collaborative

5.4.1 Vue d'ensemble du processus de mise en correspondance collaborative

Ce (sous-)processus est présenté sur la Figure 5.12 en adoptant une syntaxe concrète associée au standard SPEM [OBJECT MANAGEMENT GROUP (OMG), 2008] et en incluant des éléments de la syntaxe concrète de MMCollab (présentée dans la section 4.3.7).

Acteurs impliqués :

— Coordinateurs locaux : Un coordinateur local est un concepteur appartenant à l'équipe de conception d'un modèle source. On suppose que chaque point de vue métier (équipe de conception) est représenté par un seul coordinateur local et on a donc autant de coordinateurs locaux que de points de vue métier représentant le système.

— Manager : Le manager est la personne qui lance le processus de mise en correspondance collaborative et qui spécifie par conséquent la politique de décision qui sera adoptée lors des collaborations. Il peut être l'un des coordinateurs locaux des équipes métier comme il peut aussi être une personne à part qui a une vue transverse sur le système et qui pilote/coordonne les différentes équipes (chef de projet par exemple).

— Expert en sémantique : L'expert en sémantique a pour rôle d'associer des expressions sémantiques aux types de relations apparaissant dans le méta-modèle MMC.

— HMCS-Collab : C'est l'outil qui étend la suite HMCS [EL HAMLAOUI et al., 2014, 2019], le prototype support de l'approche AHM. HMCS-Collab supporte la collaboration lors de la mise en correspondance et inclut aussi des fonctionnalités pour l'alignement et le maintien de la cohérence lors des évolutions.

Enchaînement du processus

Le manager et chaque coordinateur local analysent - individuellement - le noyau de MMC pour voir s'il est suffisant pour décrire les correspondances du système étudié (*activité 1 : Vérifier l'adéquation du MMC*). Suite à cette analyse, le manager peut lancer la première collaboration qui correspond à *l'activité 2 : Étendre MMC* (section 5.4.2) qui permet principalement aux coordinateurs locaux de proposer - s'ils le souhaitent - de nouveaux types de relations (DSR).
Que le noyau de MMC soit étendu ou pas, le coeur du processus est *l'activité 3 : Définir les méta-correspondances* (section 5.4.3) qui permet aux coordinateurs locaux de collaborer pour proposer et valider les correspondances au niveau méta-modèle (i.e., les méta-correspondances).
La dernière activité du processus est *l'activité 4 : Générer le modèle de correspondances* (section 5.4.4) ; elle se base sur la sémantique des relations implémentées afin de produire les correspondances issues des méta-correspondances validées et qui satisfont la sémantique des relations qu'elles utilisent.

Artefacts en entrée

— Les méta-modèles représentant les points de vue métier : entrées des activités 1 et 3 ;

— Les modèles représentant les points de vue métier : entrées de l'activité 4 ;

— Le Méta-Modèle de Correspondances (MMC) : entrée des activités 1 et 4.

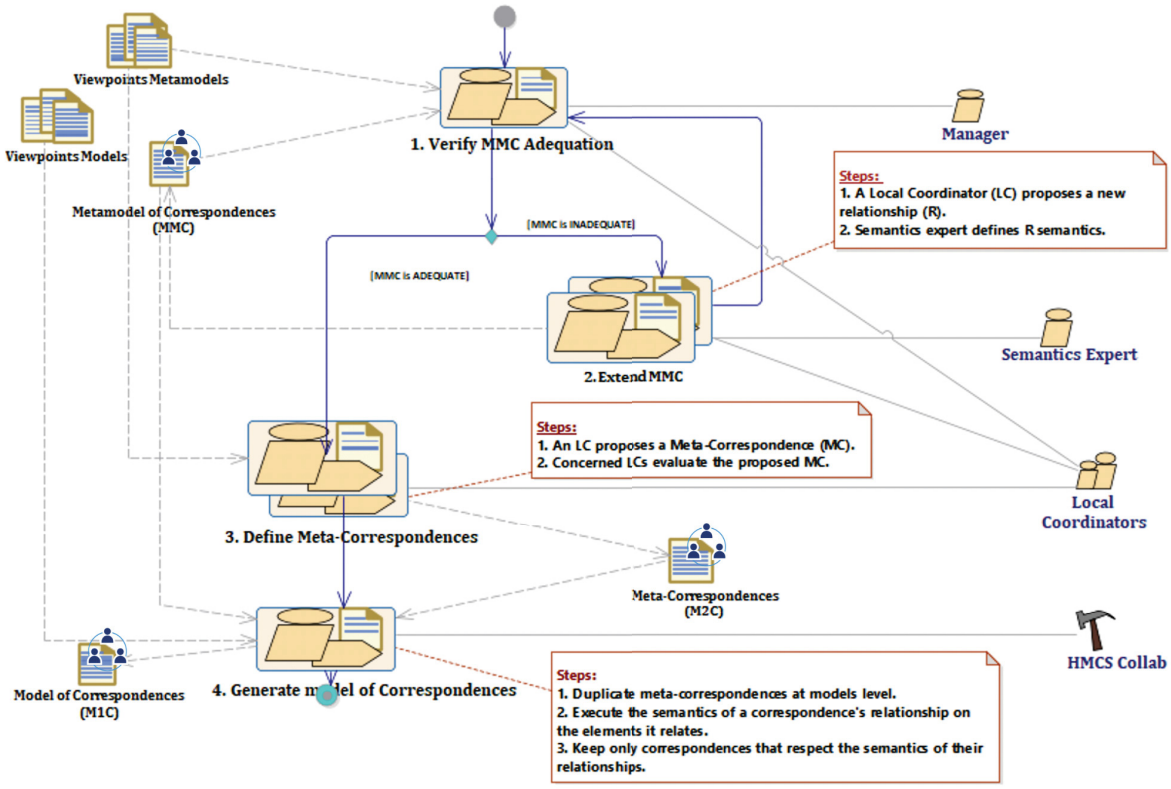


FIGURE 5.12 – Vue d'ensemble du processus de mise en correspondance collaborative.

Artefacts en sortie

— Le Modèle de Correspondances (M1C) : sortie de l'activité 4 ;

— Les méta-correspondances (M2C) : sortie de l'activité 3.

Dans la suite de la section 5.4, nous détaillons les activités principales de ce (sous-)processus.

5.4.2 Activité 2 : Étendre MMC (Extend MMC)

5.4.2.1 Description de « Étendre MMC »

Lorsque MMC a été jugé incomplet par l'activité 1, il doit être étendu. Ainsi, les coordinateurs locaux peuvent proposer des relations spécifiques (DSR) au système étudié, qui seront utilisées dans la définition des (méta-)correspondances. Le rôle de l'expert en sémantique est de fournir les définitions formelles des nouvelles relations et de les mettre en œuvre dans l'outil HMCS-Collab. **Extend MMC** est une *Collaboration* au sens de MMCollab. Les *Proposals* sont les nouvelles DSR à inclure dans le MMC et le *CollaborativeWorkProduct* résultant est le MMC étendu.

5.4.2.2 Mise en oeuvre de « Étendre MMC » sur l'exemple fil conducteur CMS

Choix de la politique de décision pour le CMS

Avant d'exécuter chaque collaboration, le modérateur (MOD), rôle joué par l'auteure du manuscrit, choisit la politique de décision à adopter. La Figure 5.13 résume le choix de la politique de décision. Lors de la collaboration *Extend MMC*, MOD choisit la politique de décision ***Restricted Taking Advice*** dans laquelle le coordinateur local SD_{LC} est impliqué pour apporter son savoir puisqu'il a des connaissances en IDM et web sémantique. La prise de décision est restreinte à MOD.

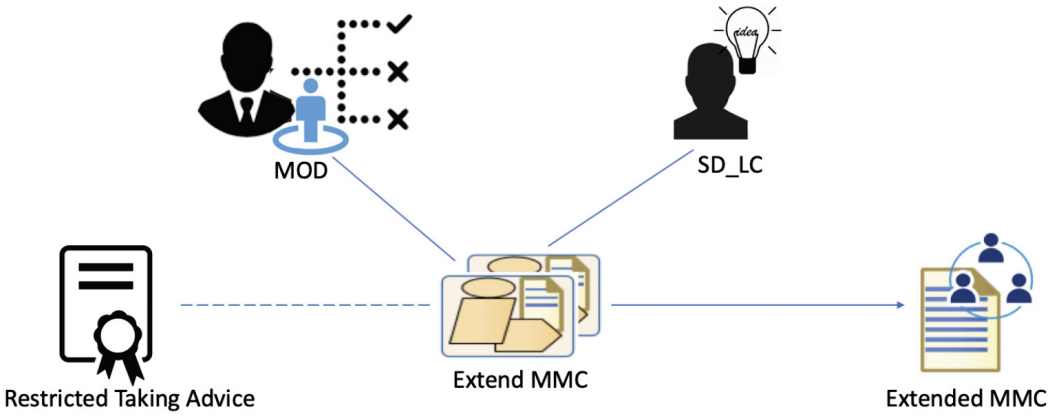


FIGURE 5.13 – Choix de la politique de décision pour l'activité *Extend MMC*.

Déroulement de la collaboration sur le CMS

La Figure 5.14 montre le déroulement de la collaboration *Extend MMC*. SD_{LC} fait ses propositions qui sont deux nouvelles relations : *induction* et *déduction*. MOD évalue ensuite ces propositions. Dans cet exemple, elle a validé les deux propositions. A la fin de la collaboration, le produit de la collaboration est le « MMC étendu » (*Extended MMC*).

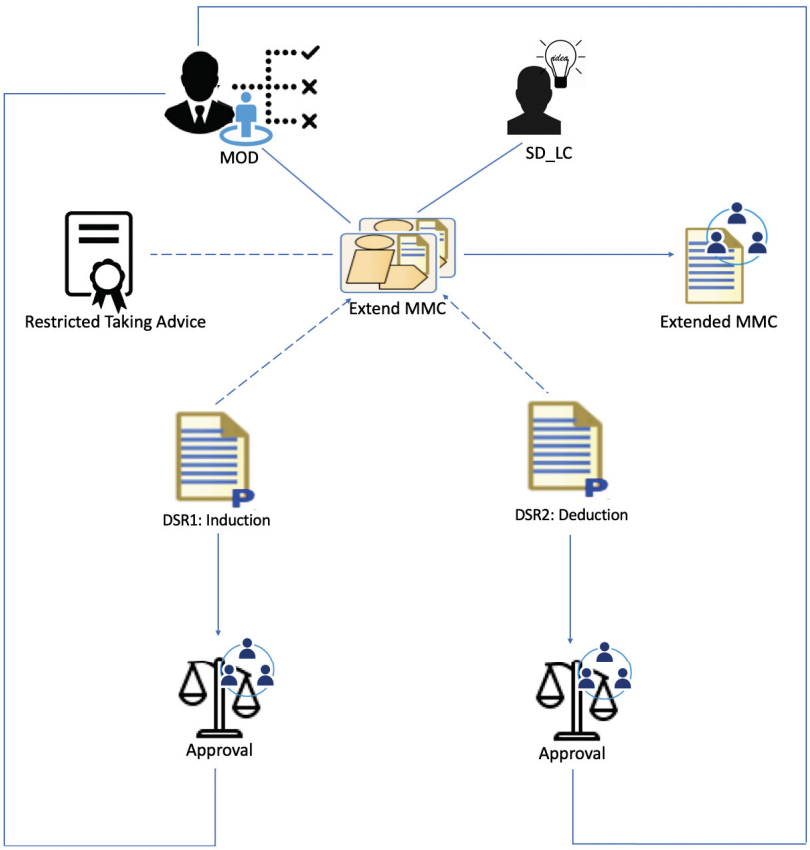


FIGURE 5.14 – Déroulement de la collaboration *Extend MMC*.

Une fois les relations spécifiques proposées et évaluées, l'expert en sémantique doit leur associer des expressions sémantiques et les intégrer dans HMCS-Collab. Dans ce cas, l'expert en sémantique s'est basé (1) sur les descriptions fournies par la personne qui les a proposées (SD_{LC}) et (2) sur les

Rapport-gratuit.com
LE NUMERO 1 MONDIAL DU MÉMOIRES

relations internes des bases de connaissances WordNet [6] et ConceptNet [7] comme résumé dans l'annexe C.

5.4.3 Activité 3 : Définir les méta-correspondances (Define Meta-Correspondences)

5.4.3.1 Description de « Définir les méta-Correspondances »

Une fois les relations du MMC fixées, les coordinateurs locaux définissent les méta-correspondances. En réalité, chaque coordinateur local propose des méta-correspondances impliquant des méta-éléments de son méta-modèle.
En s'appuyant sur le MMC et les méta-modèles des points de vue, il spécifie les méta-éléments impliqués dans la méta-correspondance (c'est-à-dire les méta-éléments de son méta-modèle et ceux issus des autres méta-modèles) et les relations qui les unissent. Après cela, les méta-correspondances proposées sont évaluées collaborativement.
L'activité **Define Meta-Correspondences** est la deuxième *Collaboration* du processus de mise en correspondance. Les *proposals* sont les méta-correspondances proposées et le *CollaborativeWorkProduct* contient les méta-correspondances approuvées.

5.4.3.2 Mise en oeuvre de « Définir les méta-Correspondances » sur l'exemple fil conducteur CMS

Choix de la politique de décision pour le CMS
La Figure 5.15 présente un extrait de l'instanciation de MMCollab pour cette collaboration, représentant le choix de la politique de décision (DP).

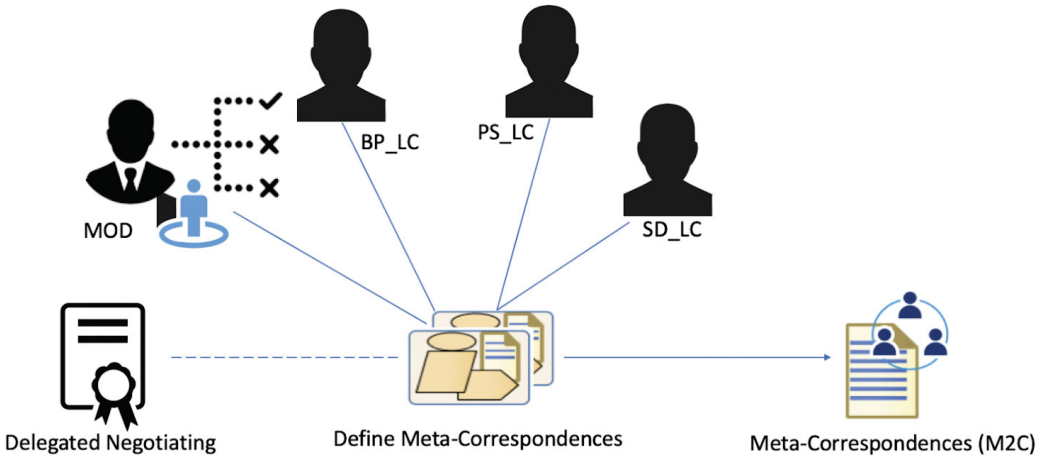


FIGURE 5.15 – Choix de la politique de décision pour l'activité *Define Meta-Correspondences*.

MOD choisit la politique de décision ***Delegated Negotiating*** qui est une politique de décision restrictive adoptant un processus de négociation suivie de vote. Le seuil d'agrément fixé pour cette collaboration est *medium*, ainsi, les propositions sont validées si la valeur du seuil est entre 50% et 66%.
MOD choisit ensuite les acteurs à impliquer dans cette collaboration. La liste *eligibleDMs* est constituée au départ des trois concepteurs locaux (SD_{LC}, PS_{LC} et BP_{LC}), en plus du MOD. Cependant, comme la politique est *Delegated Negotiating* et que le critère de sélection des décideurs considéré est leur implication dans une correspondance, l'ensemble des décideurs pour chaque proposition est restreint en excluant son initiateur et les acteurs n'ayant aucun méta-élément de leur méta-modèle impliqué dans la méta-correspondance proposée.

6. https://wordnet.princeton.edu
7. http://conceptnet.io

Déroulement de la collaboration sur le CMS

Les coordinateurs locaux font leurs propositions en termes de méta-correspondances pour le système CMS. La Table 5.1 résume les propositions faites (numérotées), et précise le corps (*body*) de chaque méta-correspondance proposée et les acteurs concernés (initiateur (*initiator*) et décideurs (*decisionMaker*)). Huit méta-correspondances binaires (impliquant deux méta-modèles) et une ternaire (impliquant trois méta-modèles : MC7) ont été proposées.

A titre d'exemples, MC1 définit une relation de similarité entre une *PS :Column* et une *SD :Property*; MC2 relie ces mêmes méta-éléments par une relation de déduction, tandis que MC3 indique que plusieurs *SD :Property* peuvent être agrégées en une *PS :Column*.

Le symbole → est utilisé pour exprimer qu'une relation est asymétrique, tandis que ↔ est utilisé pour les relations symétriques (*bidirectional* = *True*).

TABLEAU 5.1 – Méta-correspondances proposées pour le CMS

N° MC	Proposal body	initiator	decisionMakers
MC1	Similarity[PS :Column ↔ SD :Property]	SD_{LC}	PS_{LC}
MC2	Deduction[PS :Column ↔ SD :Property]	SD_{LC}	PS_{LC}
MC3	Agregation[SD :Property → PS :Column]	PS_{LC}	SD_{LC}
MC4	Similarity[PS :Table ↔ SD :Entity]	SD_{LC}	PS_{LC}
MC5	Similarity[BP :DataObject ↔ SD :Entity]	BP_{LC}	SD_{LC}
MC6	Dependency[BP :Task → SD :Operation]	BP_{LC}	SD_{LC}
MC7	Induction[BP :Task, SD :Operation → PS :Column]	BP_{LC}	PS_{LC}, SD_{LC}
MC8	Similarity[PS :Table ↔ SD :StereotypedEntity]	PS_{LC}	SD_{LC}
MC9	Deduction[PS :Table → SD :Entity]	PS_{LC}	SD_{LC}

MC1, MC2, MC4 ont été proposées par le coordinateur local du point de vue conception logicielle (SD_{LC}) ; MC3, MC8 et MC9 par PS_{LC} ; MC5, MC6 et MC7 par BP_{LC}.

Comme MC1, MC2, MC3, MC4, MC5, MC6, MC8 et MC9 sont binaires, il y a un seul décideur pour chacune d'elles : c'est le coordinateur qui est concerné par la méta-correspondance (à part l'initiateur). MC7 implique deux décideurs puisqu'elle est ternaire. Sur la Figure 5.16, seules MC6 et MC7 sont présentées ; ces deux méta-correspondances ont été choisies pour leur particularité :

— L'évaluation de MC6 a donné lieu à une instance d'*AlternativeProposal*. En effet, le décideur pour cette méta-correspondance, à savoir SD_{LC}, a proposé de raffiner cette proposition en définissant une relation d'induction entre *BP :Task* et *SD :Operation* (MC6A) à la place d'une relation de dépendance (MC6). Il a justifié cette décision par le fait que l'induction est plus précise que la dépendance. C'est alors à BP_{LC} d'évaluer MC6A qui a pour initiateur SD_{LC}. MC6 et MC6A ne peuvent pas être approuvées simultanément puisque SD_{LC} a positionné l'attribut *isConflictualWithEP* de MC6A à vrai.

— MC7 est une méta-correspondance ternaire qui implique les deux décideurs SD_{LC} et PS_{LC}. Les deux l'ont approuvée.

La Table 5.2 montre les résultats d'évaluation des méta-correspondances proposées. Seule MC9 a fait l'objet d'un rejet direct. En effet, SD_{LC} a associé le commentaire suivant à son évaluation individuelle « une *Entity* ne peut pas être déduite d'une *Table*. C'est plutôt une relation de *Similarity* qui devrait relier ces deux concepts et cette relation est déjà exprimée par MC1 ». Le raffinement de MC6 est considéré comme un rejet puisqu'elle est en conflit avec MC6A et que cette dernière a été approuvée.

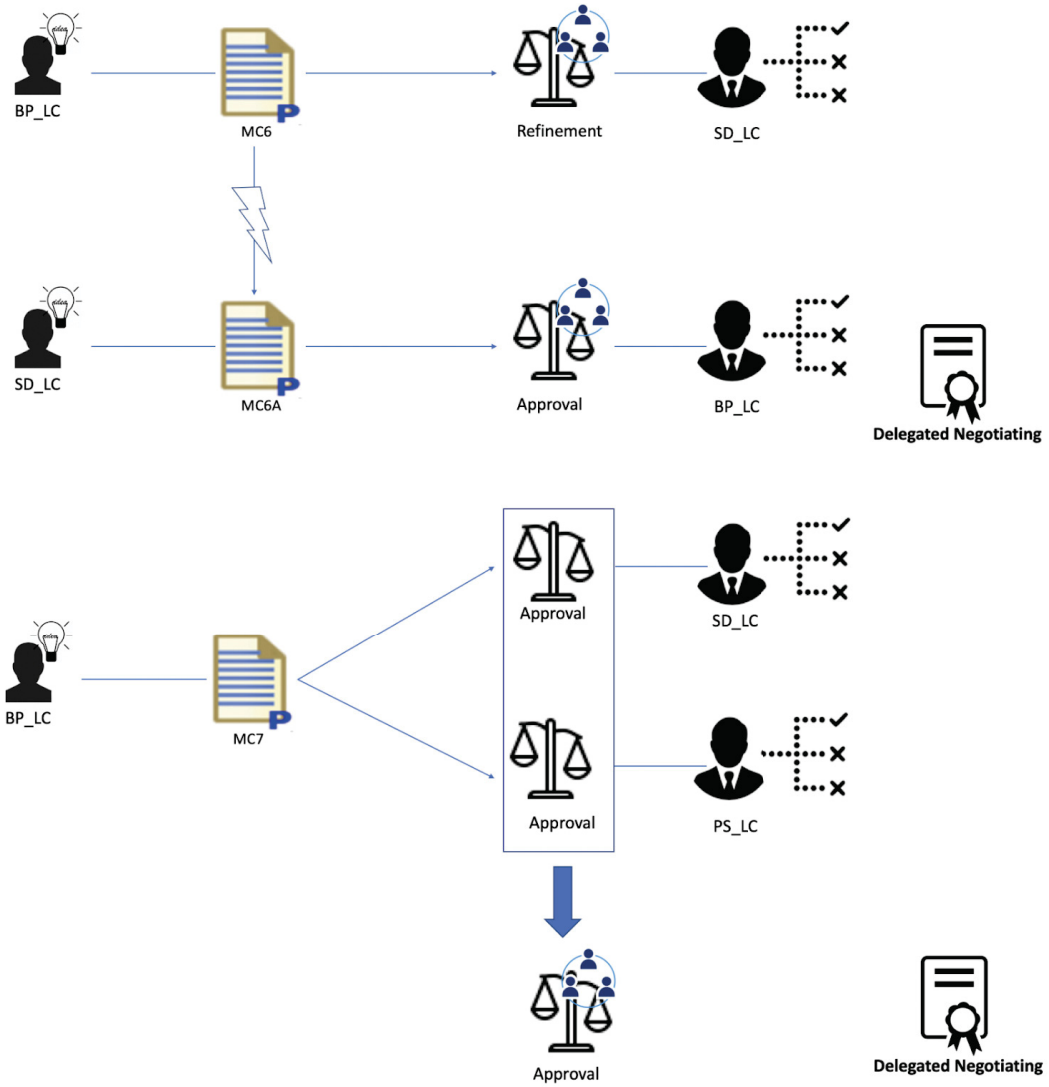


FIGURE 5.16 – Extrait de la collaboration *Define Meta-Correspondences* (méta-correspondances MC6 et MC7).

TABLEAU 5.2 – Décisions collectives pour les méta-correspondances proposées pour le CMS

N° MC	Proposal body	initiator	decisionMakers	decision
MC1	Similarity[PS :Column ↔ SD :Property]	SD_{LC}	PS_{LC}	approval
MC2	Deduction[PS :Column ↔ SD :Property]	SD_{LC}	PS_{LC}	approval
MC3	Agregation[SD :Property → PS :Column]	PS_{LC}	SD_{LC}	approval
MC4	Similarity[PS :Table ↔ SD :Entity]	SD_{LC}	PS_{LC}	approval
MC5	Similarity[BP :DataObject ↔ SD :Entity]	BP_{LC}	SD_{LC}	approval
MC6	Dependency[BP :Task → SD :Operation]	BP_{LC}	SD_{LC}	reject
MC6A	Induction[BP :Task → SD :Operation]	SD_{LC}	BP_{LC}	approval
MC7	Induction[BP :Task, SD :Operation → PS :Column]	BP_{LC}	PS_{LC}, SD_{LC}	approval
MC8	Similarity[PS :Table ↔ SD :StereotypedEntity]	PS_{LC}	SD_{LC}	approval
MC9	Deduction[PS :Table → SD :Entity]	PS_{LC}	SD_{LC}	reject

5.4.4 Activité 4 : Générer le modèle de correspondances (Generate Model of Correspondences)

5.4.4.1 Description de « Générer le modèle de correspondances »

L'outil HMCS-Collab génère les correspondances entre modèles en utilisant comme entrées les modèles par points de vue, MMC et l'ensemble des méta-correspondances approuvées.
Ces correspondances sont générées automatiquement par HMCS-Collab, et stockées dans le modèle de correspondances (M1C) qui est conforme au MMC. La génération exploite la sémantique des relations par le biais d'un processus de propagation composé des étapes suivantes [El Hamlaoui, 2015] :

— Une ***duplication***, i.e., génération du produit cartésien des instances des méta-correspondances approuvées. Cette étape produit comme résultat une liste de correspondances conformes aux méta-correspondances à partir desquelles elles ont été générées. Toutes ces correspondances ne sont pas forcément valides : c'est dans l'étape suivante qu'elles font l'objet d'un filtrage.

— Un ***filtrage*** des correspondances, i.e., l'élimination des correspondances qui ne respectent pas la sémantique des relations qu'elles utilisent. Pour rappel, l'expert en sémantique associe, lors de l'activité 2, des expressions sémantiques à chaque relation nouvellement créée, puis ces expressions sémantiques sont intégrées dans l'outil HMCS-Collab.

5.4.4.2 Mise en oeuvre de « Générer le modèle de correspondances » sur l'exemple fil conducteur CMS

Si on prend l'exemple de MC6A qui relit une *BP :Task* à une *SD :Operation* par une relation d'*induction*, et qu'on prend l'élément *Task :Edit review*, la duplication de MC6A au niveau modèle génère 12 correspondances (puisque le modèle SD comporte 12 *SD :Operation*), comme détaillé sur la Figure 5.17.

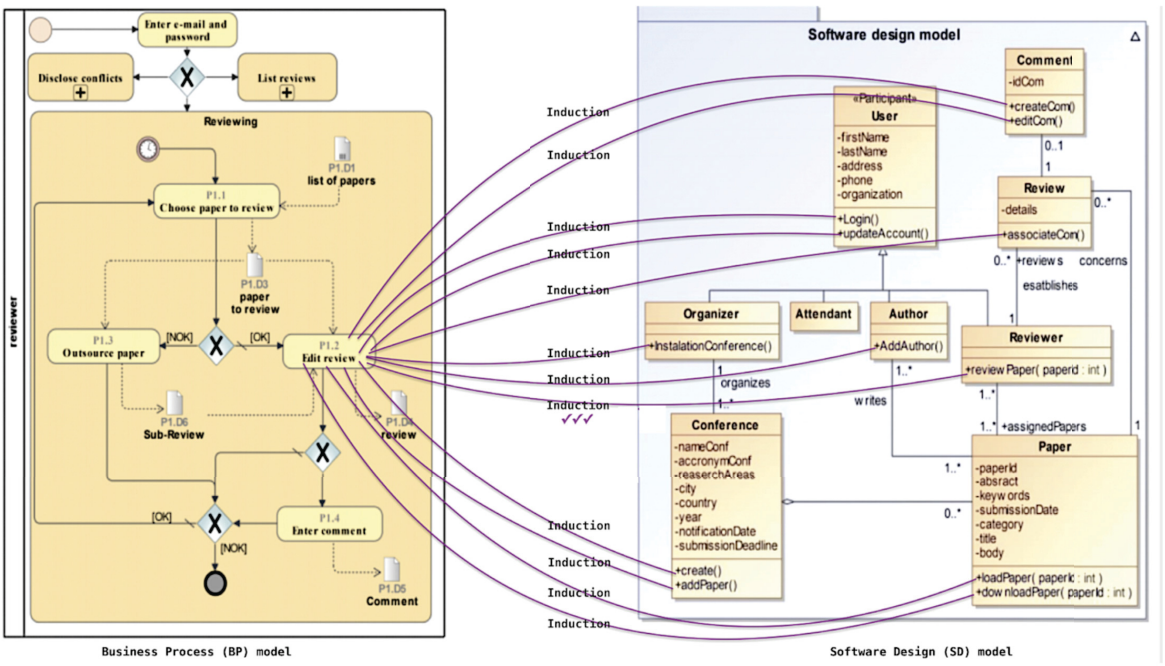


FIGURE 5.17 – Propagation de la méta-correspondance MC6A.

Après le filtrage de ces 12 correspondances basé sur la sémantique de la relation d'induction, seule la correspondance marquée par ✓✓✓ est gardée.
La Figure 5.18 montre les correspondances qui respectent la sémantique de leurs relations et qui ont été par conséquent conservées dans le modèle de correspondances. On y trouve onze

correspondances à base de similarité, cinq correspondances à base d'induction et une à base d'agrégation.

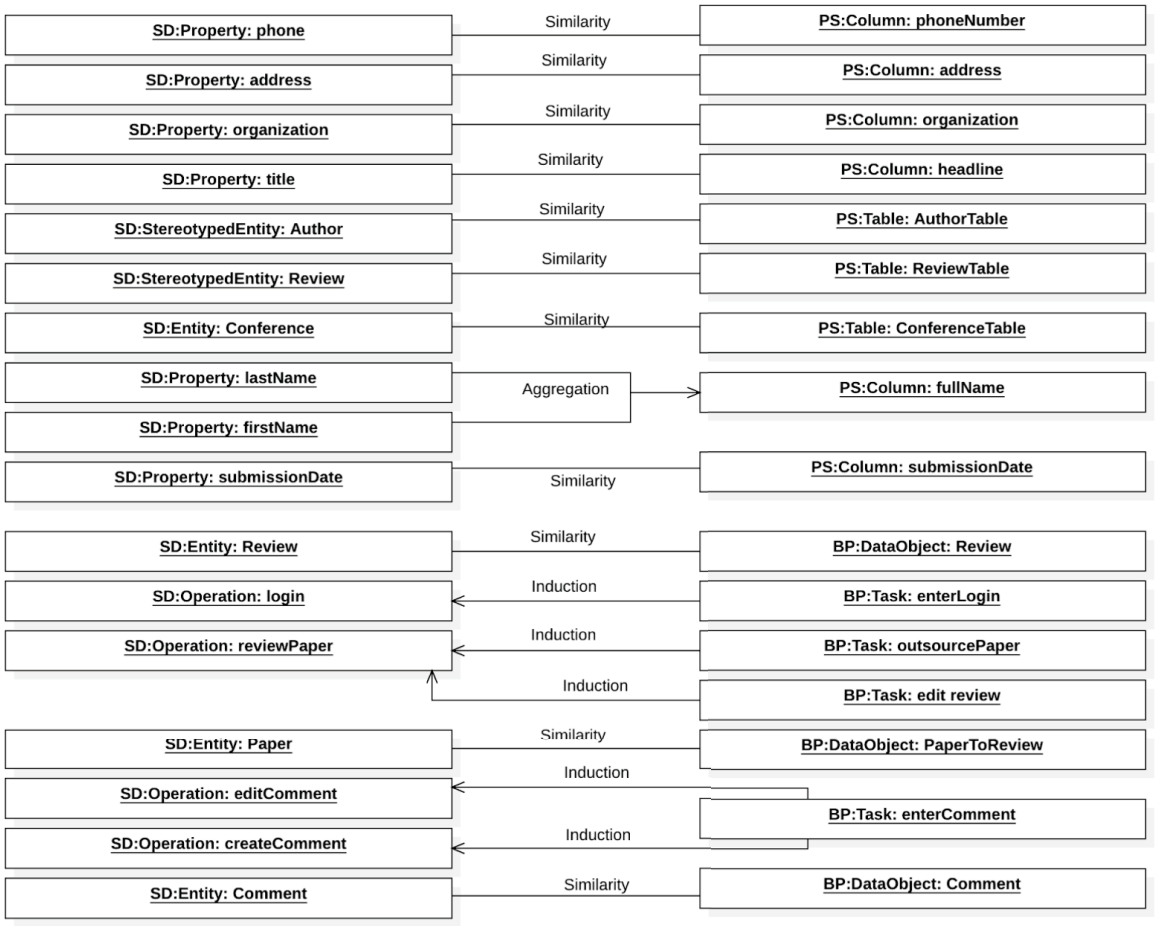


FIGURE 5.18 – Modèle de correspondances M1C du CMS après filtrage.

5.5 CAHM - phase 2 : Maintien de la cohérence lors des évolutions

5.5.1 Évolutions supportées

En IDM, de nombreux types d'évolution peuvent se produire. Les changements peuvent concerner des méta-modèles ou des modèles. Nous proposons de classer les changements en globaux et partiels.
Un **changement global** se produit lorsqu'un (méta-)modèle est créé ou supprimé, tandis qu'un **changement partiel** se produit lorsqu'un élément d'un (méta-)modèle est ajouté, supprimé ou modifié. Ainsi, les changements possibles dans un contexte d'alignement de modèles sont les suivants :

— Ajout d'un point de vue (un méta-modèle et les modèles qui lui sont conformes) ;
— Suppression d'un méta-modèle ;
— Modification d'un méta-modèle ;
— Ajout d'un modèle conforme à un méta-modèle existant ;
— Suppression d'un modèle ;
— Modification d'un modèle (i.e., ajout, suppression ou modification d'un élément du modèle).

Nous limitons l'ensemble des évolutions possibles en fonction des hypothèses suivantes :

(H1) : Nous supposons que la cohérence interne d'un méta-modèle sort du cadre de notre étude. Ainsi, les changements partiels au niveau méta-modèle ne sont pas supportés. Si un méta-modèle doit subir des changements partiels, on se contente de le traiter, dans ce travail, comme une suppression de ce méta-modèle suivie de l'ajout d'un nouveau méta-modèle pour le même point de vue.

(H2) : Nous considérons qu'il y a un modèle par point de vue et supposons que les modèles source sont hétérogènes. Autrement dit, si pour un système donné, plusieurs modèles source étaient conformes au même méta-modèle, il faudrait prévoir une pré-phase d'unification de ces modèles pour produire le modèle source du point de vue.

Par conséquent, les changements supportés par notre processus de gestion de l'évolution sont les suivants :

— Ajout d'un point de vue (un modèle et son méta-modèle) ;

— Suppression d'un point de vue ;

— Ajout d'un modèle (conforme à un méta-modèle existant) ;

— Suppression d'un modèle ;

— Modification d'un modèle.

5.5.2 MMC : version étendue pour supporter l'évolution

La prise en compte de ces changements a été intégrée dans le méta-modèle MMC. La Figure 5.19 présente une vue complète du MMC intégrant en plus de son noyau (partie droite), l'extension apportée dans le cadre de cette thèse pour gérer l'aspect évolutif des (méta-)modèles (partie gauche).
Un *Repository* conserve l'historique des évolutions des (méta-)modèles source. Il est composé d'un ensemble de versions caractérisant les changements opérés sur les modèles.
Une *ChangesVersion* consiste en un ensemble de modifications et possède un état. Un état en attente (*pending*) signifie que l'impact des modifications apportées à un (méta-)modèle sur les autres (méta-)modèles n'a pas encore été analysé et considéré, un état validé (*validated*) signifie que les impacts des modifications apportées à un (méta-)modèle sur les autres (méta-)modèles ont été analysés et traités.
Un changement global concerne un (méta-)modèle (*concernedModel*). Les changements globaux (i.e., création ou suppression d'un (méta-)modèle) sont répertoriés dans l'énumération *ChangeGlobalKind*.
Un *PartialChange* concerne un élément de modèle (*concernedElt*). En nous inspirant des travaux de HEBIG et al. [2015] et de KHELLADI et al. [2017], nous répertorions dans l'énumération *ChangePartialKind* les changements partiels suivants :

— *Create* : ajout d'une propriété ou d'une classe à un modèle,

— *Delete* : suppression d'une propriété ou d'une classe d'un modèle,

— *Move* : déplacement d'une propriété d'une classe à une autre. Il s'agit d'un *pull up* si la propriété est déplacée vers une classe parente et d'un *push down* si la propriété est déplacée vers une classe descendante,

— *Rename* : renommage d'un élément de modèle,

— *Extract class* : ce changement peut être nécessaire quand une classe a beaucoup de méthodes et que son objectif n'est pas clair. Extract class permet la création d'une nouvelle classe et le déplacement de méthodes et/ou de données vers cette nouvelle classe,

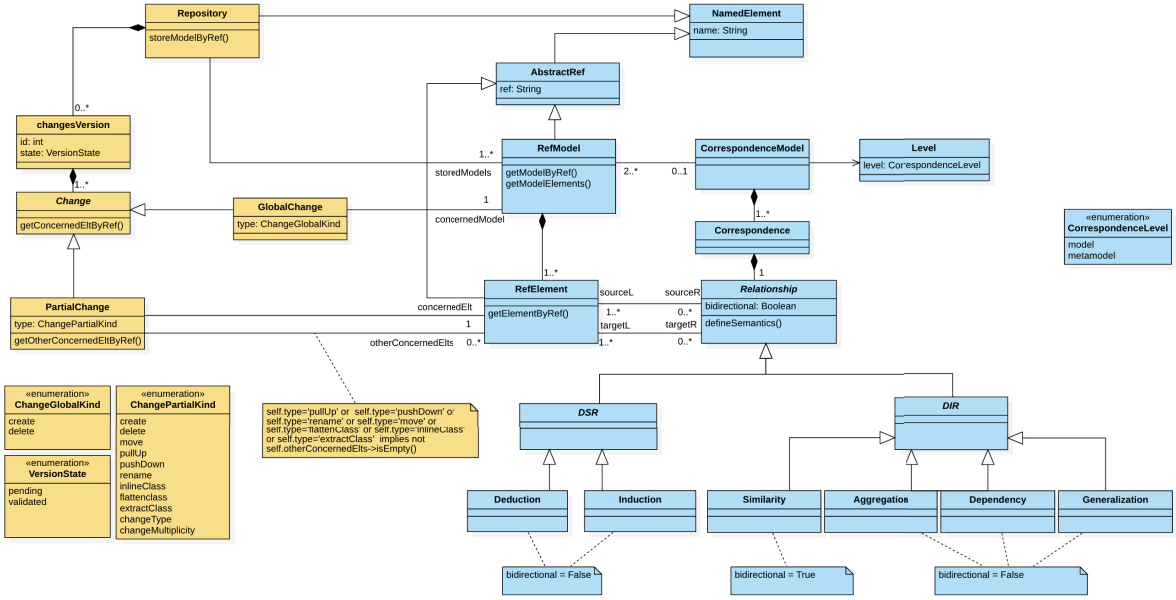
— *Inline class* : la suppression d'une classe après avoir déplacé ses propriétés dans une autre classe,

— *Flatten class* : la suppression d'un ensemble de propriétés p1, ..., pn d'une classe-mère A, et leur ajout dans les sous-classes B1, ... Bn. Ensuite, la classe-mère A est supprimée,

— *Change multiplicity* : le changement de la multiplicité d'une association,

— *Change type* : le changement du type d'une propriété.

Ces changements partiels peuvent être classés en atomiques (*Create, Delete, Rename, Change type, Change multiplicity*) ou composites (*Move, Pull up, Push down, Extract class, Inline class, Flatten class*). Dans CAHM, nous traitons tout changement composite comme une suite de changements atomiques.
Move, Pull up, Push down, Extract class, Inline class, Flatten class et *Rename* impliquent plus d'un élément. Par exemple, le *Pull up* de l'élément « i » de la classe « A » vers la classe « B » est un changement partiel qui a « i » comme *concernedElt* et « A » et « B » comme *otherConcernedElts*. Le *Rename* d'un élément « e » par « f », a comme *concernedElt* « e » et « f » comme *otherConcernedElts*. Ainsi si on renomme l'*Entity Author* par *Writer* dans le modèle SD du CMS, le *concernedElt* est *Author* et l'*otherConcernedElts* est *Writer*.



En bleu : MMC [EL HAMLAOUI, 2015], **En jaune** : concepts ajoutés dans le cadre de cette thèse

FIGURE 5.19 – Vue complète du méta-modèle de correspondances.

5.5.3 Vue d'ensemble du processus de traitement des évolutions de (méta-)modèles

Une vue d'ensemble du processus est donnée sur la Figure 5.20. Ce processus traite les évolutions effectuées sur les points de vue métier.

Acteurs impliqués

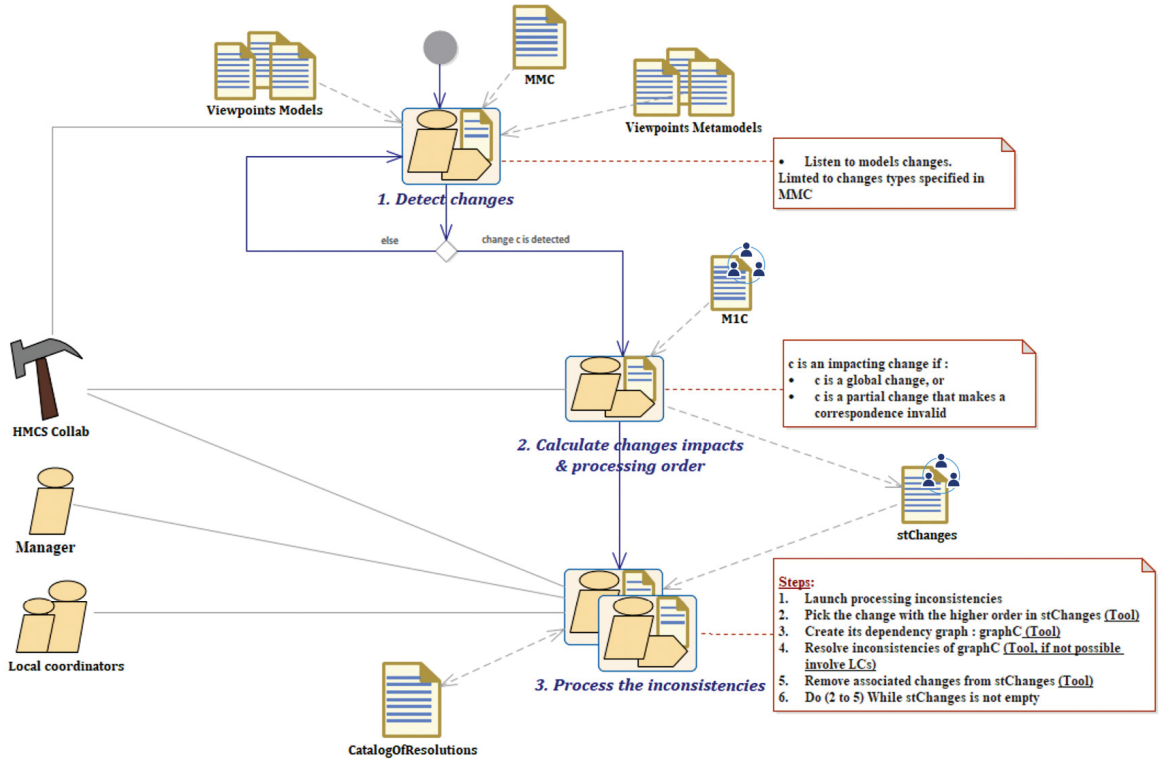
— Coordinateurs locaux ;

— Manager de la collaboration ;

— Outil HMCS-Collab.

Enchaînement du processus

D'abord, l'outil HMCS-Collab observe les (méta-)modèles (activité 1) pour capturer les changements (section 5.5.4). Les modifications détectées sont stockées dans une pile de changements dans l'ordre croissant de leurs impacts par l'intermédiaire de l'activité 2 (section 5.5.5). Ensuite, vient l'activité de traitement des incohérences provoquées par les changements (activité 3) qui est une activité collaborative elle-même composée de sous-activités (voir section 5.5.6). Cette activité peut nécessiter l'implication des coordinateurs locaux s'il n'y a pas de résolution automatique unique applicable pour le traitement de l'incohérence identifiée.



N.B : Nous considérons que les artefacts peuvent être exploités dans toute activité postérieure à celle de leur première utilisation.

FIGURE 5.20 – Vue d'ensemble du processus de traitement des évolutions de (méta-)modèles.

Artefacts en entrée

- Les méta-modèles représentant les points de vue métier (entrée de l'activité 1) ;
- Les modèles représentant les points de vue métier(entrée de l'activité 1) ;
- Le méta-modèle de correspondances - MMC (entrée de l'activité 1) ;
- Le modèle de correspondances (entrée de l'activité 2) ;

Artefacts en sortie

- Le modèle de correspondances ;
- Les méta-modèles et les modèles ;
- La pile des changements opérés (*stChanges*).

5.5.4 Activité 1 : Détecter les changements (Detect changes)

5.5.4.1 Description de « Détecter les changements »

Cette activité repose sur le méta-modèle de correspondances (MMC). En effet, l'outil HMCS-Collab observe[8] le répertoire (concept *Repository* du MMC, qui regroupe les changements possibles) et capture automatiquement les types de changement supportés par MMC, en se basant sur les UUID[9].

5.5.4.2 Mise en oeuvre de « Détecter les changements » sur le CMS

Supposons que les modèles du CMS subissent les changements partiels suivants :

— Ajout de *Pool :Chairman* au modèle Business Process (BP) ;

— Suppression de *Entity :Comment* du modèle Software Design (SD) ;

— Suppression de Task :Outsource paper du modèle BP ;

— Renommage de *Task* :*Enter comment* du modèle BP, son nouveau nom étant *Add comment*.

Le tableau 5.3 décrit ces évolutions : la colonne *Version* désigne la version du répertoire dans laquelle ce changement a eu lieu. La colonne *Change* spécifie le type de changement conformément à l'énumération *ChangePartialKind* présentée dans la section 5.5.1. *RefModel* désigne le modèle auquel appartient l'élément changé alors que *ConcernedElt* identifie l'élément changé. *OtherConcernedElts* est applicable en cas de changements composites.

TABLEAU 5.3 – Exemples d'évolution des modèles du CMS

Version	Change	RefModel	ConcernedElt	OtherConcernedElts
2	Create	BP	Pool :Chairman	–
3	Delete	SD	Entity :Comment	–
4	Delete	BP	Task :OutsourcePaper	–
4	Rename	BP	Task :EnterComment	Task :addComment

5.5.5 Activité 2 : Calculer l'impact et l'ordre de traitement des changements (Calculate changes impacts & processing order)

5.5.5.1 Description de « Calculer l'impact et l'ordre de traitement des changements »

Une fois qu'un changement est détecté, HMCS-Collab vérifie s'il est impactant.
Un changement est impactant s'il est :

— Un changement global ou

— Un changement partiel de type ajout ou

— Un changement partiel d'un élément utilisé dans au moins une correspondance **ET** qu'une correspondance devient invalide à cause de ce changement.

Pour les changements impactants, HMCS-Collab calcule l'ordre de traitement du changement. Cet ordre dépend du type de changement :

— Un changement **global** est traité en priorité ; il est **inséré au sommet de la pile *stChanges*** étant donné qu'il change la structure du point de vue métier.

— Un changement **partiel** :

8. L'outil écoute les changements en se basant sur le patron de conception *observer* qui compare les UUID des éléments pour voir s'ils ont changé.

9. Universally Unique IDentifier, identifiant unique de chaque concept (i.e., chaque (méta-)modèle ou élément de (méta-)modèle). Deux éléments sont considérés comme identiques si et seulement s'ils ont le même UUID.

— de type **ajout** (i.e., *Create*) est traité en dernier ; il est **inséré au bas de la pile** étant donné qu'il n'a aucune incidence sur le modèle de correspondances déjà établi.

— de type **modification** (i.e., *Move, Pull up, Push down, Inline class, Extract class, Flatten class*) ou **suppression** (i.e., *Delete*) est traité selon son **ordre d'importance**, qui correspond au nombre de correspondances qu'il rend invalides.

L'algorithme de l'annexe B.1 décrit le calcul de l'impact d'un changement selon ces considérations.

Pour les changements non impactants, HMCS-Collab valide l'état de *ChangesVersion* (i.e. état positionné à validé, *state* = *validated*) et reste à l'écoute de changements impactants.

5.5.5.2 Mise en oeuvre de « Calculer l'impact et l'ordre de traitement des changements » sur le CMS

L'outil HMCS-Collab classe les changements effectués (cf. Table 5.3) selon qu'ils sont impactants ou non :

— Changements non impactants :

— *Delete Entity :Comment* est un changement non impactant, car *Entity :Comment* n'est pas utilisé dans le modèle de correspondances (cf. Figure 5.18) ;

— *Rename Task :EnterComment* est un changement non impactant. En effet, ce changement concerne deux correspondances. Cependant, en appliquant la sémantique de leurs relations (i.e., la relation d'induction), on constate qu'elles sont toujours valides. Le changement ne sera donc pas inséré dans la pile des changements *stChanges*.

— Changements impactants :

— *Create Pool :Chairman* est un changement impactant car c'est un changement partiel de type ajout ;

— *Delete Task :OutsourcePaper* est un changement impactant, car c'est un changement partiel impliqué dans une correspondance qui devient invalide (*induction[Task :OutsourcePaper → Operation :reviewPaper]*).

L'ordre du traitement de ces deux changements est le suivant : *Delete Task :OutsourcePaper* est traité en premier, suivi de *Create Pool :Chairman*, car les changements de type ajout sont traités en dernier.

5.5.6 Activité 3 : Traiter les incohérences (Process the inconsistencies)

5.5.6.1 Description de « Traiter les incohérences »

Les modèles source sont verrouillés durant cette activité pour que les coordinateurs locaux n'apportent pas d'autres changements.
La Figure 5.21 illustre la décomposition de cette activité en six sous-activités. Elle prend en entrée la pile des changements *stChanges* et un catalogue répertoriant les résolutions applicables par type de changement.
Le traitement des incohérences est lancé **automatiquement** en cas de **changement global** (niveau méta-modèle ou modèle).
Dans le cas contraire (**changement partiel**), le traitement des incohérences ne se fait pas à la volée ; c'est au **manager de le lancer manuellement** (activité 1) suite à une requête d'un des coordinateurs locaux qui estime qu'il vient d'apporter des changements partiels importants au modèle de son point de vue.
Une fois le traitement lancé, HMCS-Collab récupère (via la pile *stChanges*) le changement ayant le plus grand impact. Ensuite, il établit un **graphe de dépendances de ce changement** (activité 2). Ce graphe a pour objectif de tracer les éléments directement ou indirectement reliés à l'élément changé (qu'il soit un modèle, un méta-modèle ou un élément de modèle). Les éléments constituant

le graphe dépendent du type du changement. Dans l'annexe B.2, nous détaillons comment le graphe est construit dans les deux cas : (i) cas d'un changement partiel, (ii) cas d'un changement global.
Une fois les graphes de tous les changements créés, le traitement des changements démarre en commençant par le changement le plus impactant jusqu'à ce que la pile *stChanges* devienne vide. HMCS-Collab repère les (méta-)correspondances contenues dans le graphe et qui ne sont plus valides. Ainsi, le manager de la collaboration et les coordinateurs locaux ont connaissance des incohérences à résoudre. Ils pourront donc s'appuyer sur ce graphe pour décider des résolutions nécessaires, afin d'obtenir un graphe correct. Ils ont pour cela un catalogue de résolutions à disposition. Ce catalogue contient une liste extensible de résolutions classées en fonction du type de changement ; il est inspiré des catalogues de résolution et de propagation des changements de la littérature [KHELLADI et al., 2017; OUSSALAH, 2018] et est détaillé dans l'annexe B.3 ; des exemples de résolution issus de ce catalogue sont donnés tout au long de cette section.
HMCS-Collab recherche les résolutions appropriées dans le catalogue de résolutions prédéfini (activité 3), et les propose en support à la prise de décision. Néanmoins, certaines résolutions présentes dans ce catalogue permettent une mise en oeuvre automatique.

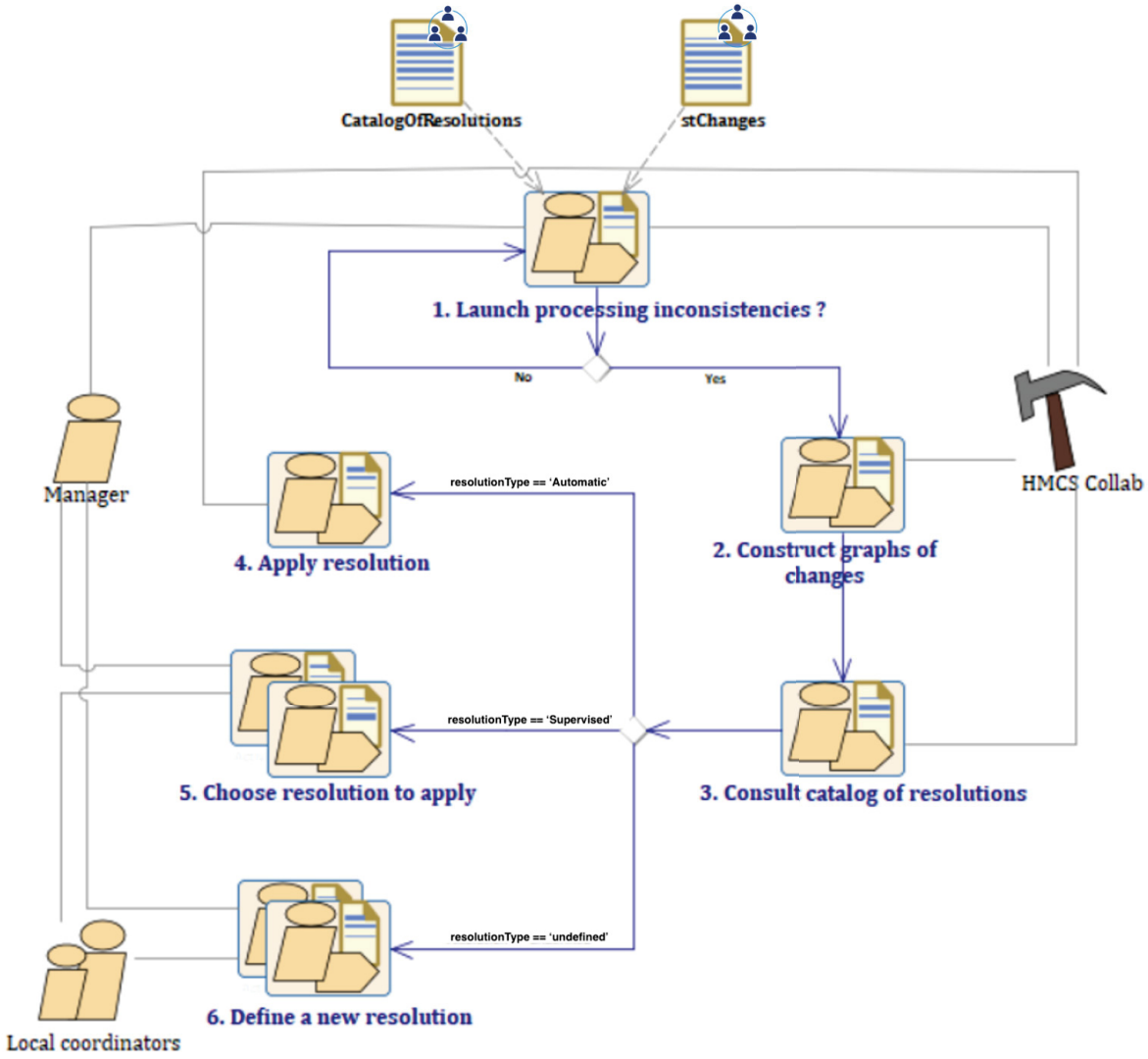


FIGURE 5.21 – Diagramme d'activités de Traiter les incohérences.

Le tableau 5.4 suivant présente un extrait du catalogue pour le cas d'un changement au niveau méta-modèle, de type global.
La colonne *Resolution* résume les résolutions possibles, tandis que la colonne *Resolution Type* précise la nature de chaque résolution (i.e., *automatique* versus *supervisée*).

Les résolutions automatiques sont appliquées par l'outil HMCS-Collab. Quand une résolution automatique est la seule envisagée pour un changement, elle est appliquée sans intervention humaine (ceci correspond à l'activité 4 : ***Apply resolution*** de la Figure 5.21).
Les résolutions supervisées sont des résolutions qui impliquent les coordinateurs locaux, soit parce que :

(i) Plusieurs résolutions dans le catalogue de résolutions sont appropriées au changement, i.e., existence de l'opérateur « or » entre au moins deux résolutions (dont certaines peuvent être automatiques), ce qui correspond à l'activité 5 : ***Choose resolution to apply*** de la Figure 5.21.

(ii) Le changement n'a pas de résolution adaptée (ceci correspond à l'activité 6 : ***Define a new resolution*** de la Figure 5.21).

TABLEAU 5.4 – Extrait du catalogue de résolutions d'un changement global

Change			Resolution	Resolution
Level	**Type**	**Subtype**		**Type**
Metamodel	Global	Delete MM1	**(RM1** or **RM2)** and **R1**, with : **RM1** : Remove meta-correspondences (MM1) **R1** : Remove correspondences(M1) **RM2** : Adjust meta-correspondances (MM1)	 Automatic Automatic Supervised
		Add MM1	**RM3** : define meta-correspondences (MM1)	Supervised
		Delete + add MM1	**RM1** or **RM2** or **RM3**	Supervised

Les deux activités **Define a new resolution** et **Choose resolution to apply** sont des collaborations sur la Figure 5.21. Il s'agit de **Define a new resolution** quand la collaboration consiste à fournir de nouvelles résolutions qui vont alimenter le catalogue de résolutions. Sinon, il s'agit de choisir la résolution à appliquer quand plusieurs résolutions sont possibles pour un changement (i.e. **Choose resolution to apply**).

5.5.6.2 Mise en oeuvre de « Traiter les incohérences » sur le CMS

Le système CMS n'ayant pas subi de changement global dans cet exemple, nous détaillons ici le traitement des changements partiels impactants qui sont pour rappel et par ordre d'impact :

— *Delete Task :OutsourcePaper*.

— *Create Pool :chairman*;

L'extrait du tableau 5.5 suivant reprend les résolutions contenues dans le catalogue des résolutions pour les changements partiels de type *create element* et *delete element*.

TABLEAU 5.5 – Extrait du catalogue de résolutions d'un changement partiel

Change			Resolution	Resolution
Level	**Type**	**Subtype**		**Type**
Model	Partial	Create element e	**RM4.1** : Propagate meta-correspondences (MM, me)	Automatic
		Delete element e	**R2** or **R3** or **R4** or **R5**, with : **R2** : Remove correspondences(e) if their cardinality = 2 **R3** : Remove correspondence's extremity if it still holds **R4** : Adjust the other models **R5** : Restore e	Supervised

Selon cet extrait, pour traiter un changement partiel de type suppression, quatre résolutions supervisées sont possibles (voir *R2, R3, R4, et R5* dans l'extrait du catalogue de résolutions ci-dessus). Ainsi, pour traiter le changement *Delete Task :OutsourcePaper*, les coordinateurs locaux doivent choisir entre R2, R3, R4 et R5.

Étant donné que l'élément *Task :OutsourcePaper* impacte uniquement la correspondance *Induction[BP :Task :OutsourcePaper → SD :Operation :reviewPaper]*, le graphe de ce changement est tel qu'illustré sur la Figure 5.22.

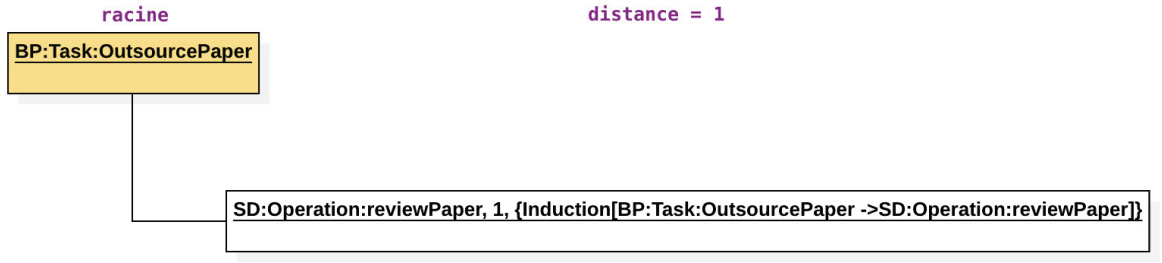


FIGURE 5.22 – Graphe de dépendances de *Task : OutsourcePaper*.

Lors de la collaboration *Choose resolution to apply* pour le changement *Delete Task :OutsourcePaper*, le modérateur MOD choisit la politique de décision ***Consenting Together***. La décision du groupe consiste donc à choisir la résolution à appliquer parmi celles envisagées dans le cas d'un *delete element* (i.e., choisir entre R2, R3, R4 et R5) et les acteurs concernés sont SD_{LC} et BP_{LC} comme résumé sur la Figure 5.23.

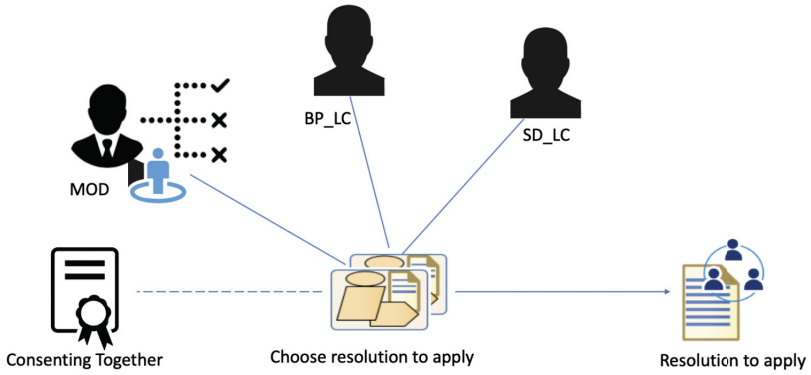


FIGURE 5.23 – Choix de la politique de décision pour l'activité *Choose resolution to apply*.

Puisque cette collaboration consiste à choisir entre des résolutions existantes et donc conflictuelles, il est judicieux de laisser les acteurs évaluer les propositions à tour de rôle pour qu'ils prennent en compte les commentaires et justifications des autres (un tour donc par acteur).
Le modérateur MOD donne la main d'abord à SD_{LC} pour évaluer les résolutions applicables pour le changement *Delete Task :OutsourcePaper* car c'est BP_{LC} qui a apporté ce changement.
Ce déroulement est explicité sur la Figure 5.24. SD_{LC} approuve R5 (*Restore deleted element*) et rejette les autres. SD_{LC} a choisi R5 étant donné que l'opération *reviewPaper()* du modèle SD se base sur *Task :OutsourcePaper*.
BP_{LC} à son tour doit évaluer les résolutions. Il informe à SD_{LC} que dans son processus métier, les articles ne sont plus externalisés pour sous-traitance. Ceci consiste donc à supprimer l'élément *OutsourcePaper* et les correspondances dans lesquelles il est impliqué, ce qui revient à choisir R2 comme résolution (*Remove correspondences(e) if their cardinality = 2*). Une fois que SD_{LC} a pris connaissance de la justification de BP_{LC}, il exprime son accord pour l'application de R2.
Pour les changements partiels de type ajout, le traitement consiste à propager les méta-correspondances impliquant le méta-élément de cet élément. Cette résolution est automatiquement réalisée par l'outil. Ainsi le changement *Create Pool :Chairman* (cf. Table 5.3) est traité de façon automatique, en propageant les méta-correspondances impliquant le méta-élément Pool du modèle BP. Or, ce méta-élément Pool n'intervient dans aucune méta-correspondance, donc le traitement de ce changement se termine sans aucun impact sur le modèle de correspondances (M1C).

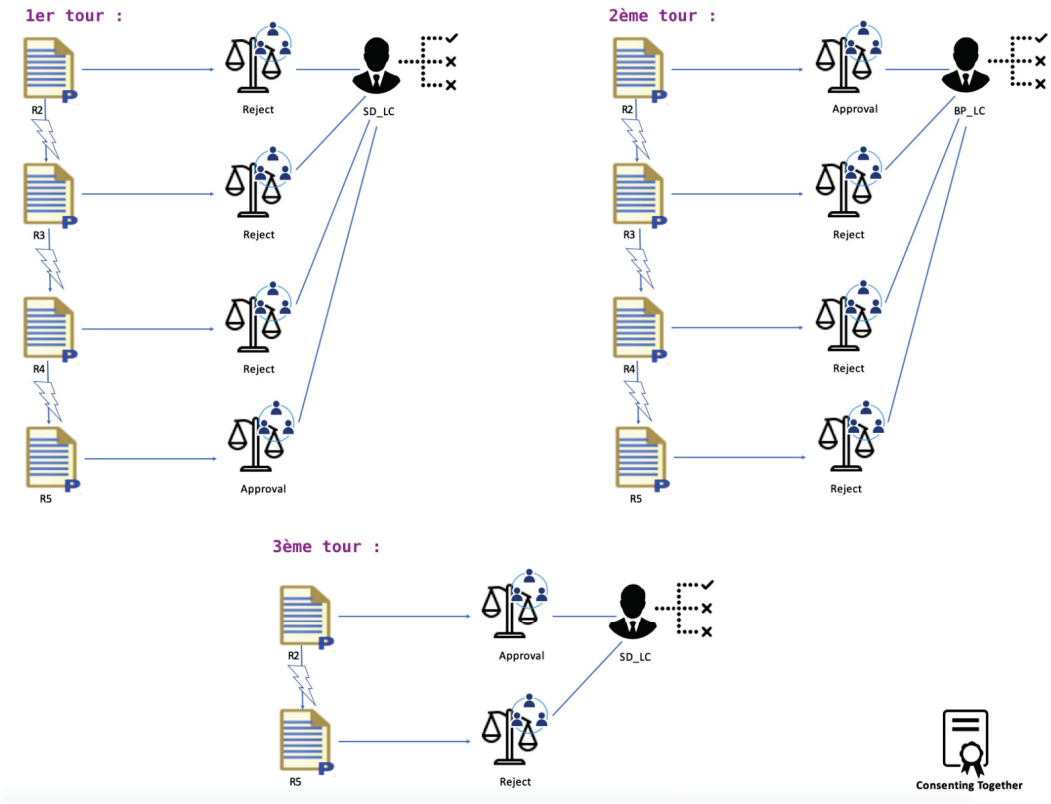


FIGURE 5.24 – Déroulement de la collaboration *Choose resolution to apply*.

5.6 Conclusion

Dans ce chapitre nous avons présenté notre approche CAHM d'alignement collaboratif de modèles hétérogènes. Elle permet aux concepteurs d'établir des correspondances entre les modèles et de les maintenir pour préserver la cohérence globale du système en cas d'évolution. Pour cela, elle se base sur le méta-modèle de Collaboration (MMCollab) et le méta-modèle des correspondances (MMC). Le principal intérêt de l'utilisation de MMCollab est de supporter la collaboration lors de l'alignement et d'exploiter la formalisation des processus GDM. MMC quant-à-lui permet de relier des modèles et d'exprimer des correspondances n-aires en fournissant un ensemble extensible de relations. Des expressions sémantiques sont associées à ces relations et intégrées dans un outil pour assurer une propagation semi-automatique des (méta-)correspondances. Le lien entre MMC et MMCollab s'établit par le fait que les propositions, au sens de MMCollab, sont des relations ou des méta-correspondances au sens de MMC.
L'approche CAHM est formalisée selon deux sous-processus : un pour la mise en correspondance collaborative (CAHM - Phase 1) et l'autre pour le maintien de la cohérence lors des évolutions des modèles (CAHM - Phase 2). Le tableau 5.6 suivant reprend la synthèse par critères de l'analyse de l'état de l'art du chapitre 2 en intégrant l'approche CAHM.
Trois approches (CAHM, OpenFlexo et AHM) se distinguent concernant l'arité des correspondances et le type des relations proposées, en supportant des correspondances n-aires avec des relations extensibles selon le besoin. Cependant, seule CAHM considère à la fois le maintien de la cohérence lors des évolutions, la persistance des correspondances et le support à la collaboration. En effet, d'une part, CAHM assure un maintien de la cohérence lors des évolutions en offrant un ensemble de recommandations de résolutions basé sur un catalogue extensible contenant des résolutions automatiques et supervisées. D'autre part, CAHM supporte la collaboration et la participation des acteurs métier dans l'élaboration et le maintien du modèle de correspondances, ceci garantit que les correspondances identifiées reflètent bien les intérêts métier et qu'il n'y a pas de surcharge du modèle de correspondances par des informations non utiles pour la vue globale.

Etablissement de correspondances	***VirtualEMF***	***AHM***	***EMF Views***	***Appr. Shosha***	***OpenFlexo***	***Appr. Vanherpen***	***CIMA***	***CAHM***
Arité binaire	●	●	●	●	●	●	●	●
Arité n-aire (n≥3)	-	●	●	-	●	●	-	●
Type de relation libre	◐	●	-	-	●	-	-	●
Persistance des correspondances	●	●	◐	-	-	-	-	●
Outil support	●	●	●	●	●	◐	◐	●
Maintien de cohérence								
Détection des changements	◐	◐	◐	-	◐	◐	-	◐
Résolution des incohérences	-	◐	-	-	◐	◐	-	◐
Outil support	◐	◐	◐	-	◐	◐	-	◐
Support de collaboration								
Coordination	-	-	-	●	●	-	●	●
Codécision	-	-	-	●	-	-	●	●
Outil support	-	-	-	●	-	-	●	●

● : Critère au centre de l'approche, ◐ : Partiellement considéré, - : Non considéré

TABLEAU 5.6 – Synthèse des caractéristiques supportées par les approches d'alignement de modèles hétérogènes incluant l'approche CAHM.

Conclusion

L'objectif de cette deuxième partie était de présenter les contributions conceptuelles de cette thèse. Le résumé de ces contributions, décrites dans les chapitres 4 et 5, peut se formuler comme suit :

1. Extension du Méta-Modèle de Correspondances (MMC), pour supporter les évolutions des modèles et des méta-modèles des points de vue métier ;
2. Élaboration d'un méta-modèle de collaboration (MMCollab) support à la description et au déroulement des prises de décision en groupe ;
3. Définition de politiques de décision configurables, fondées sur la notion de patrons de prise de décision en groupe (*GDMPattern*) en précisant les contextes d'utilisation appropriés pour chacune d'entre elles ;
4. Proposition de CAHM, une approche d'alignement collaboratif de modèles hétérogènes supportant deux processus collaboratifs : mise en correspondance des modèles et maintien de la cohérence en cas d'évolution.

Dans la partie suivante, nous présentons d'une part l'outillage proposé pour supporter l'approche CAHM (chapitre 6) et d'autre part une étude de cas qui a permis de valider expérimentalement l'approche (chapitre 7).

Troisième partie

Implantation et Validation

Introduction 127

6 Prototype HMCS-Collab 129

7 Validation expérimentale 143

Conclusion 167

Introduction

Nous rappelons tout d'abord que l'objectif de notre travail est de fournir un cadre collaboratif pour la création d'une vue globale sur les modèles représentant un système complexe et d'assurer la cohérence de cette vue en cas d'évolution des (méta-)modèles.

Pour cela, nous avons proposé, dans la deuxième partie, un méta-modèle pour gérer la prise de décision collaborative (chapitre 4) et un processus pour l'alignement collaboratif des modèles qui exploite ce méta-modèle (chapitre 5).

Dans cette partie, nous présentons la **validation de notre contribution** qui consiste en l'élaboration d'un outil support et la mise en oeuvre de l'approche sur un cas d'étude.

Dans le **chapitre 6**, nous décrivons **le prototype de l'outil développé HMCS-Collab** qui supporte le processus collaboratif d'alignement présenté dans le chapitre 5.

Le **chapitre 7** concerne la **validation expérimentale de l'approche et de l'outil développé**. Nous décrivons un cas d'étude concernant un service d'urgence d'un hôpital en spécifiant les méta-modèles et modèles de points de vue métier. Ensuite, nous déroulons l'approche sur ce cas d'étude en utilisant l'outil HMCS-Collab et nous faisons un retour sur le cas d'étude.

Chapitre 6

Prototype HMCS-Collab

Sommaire

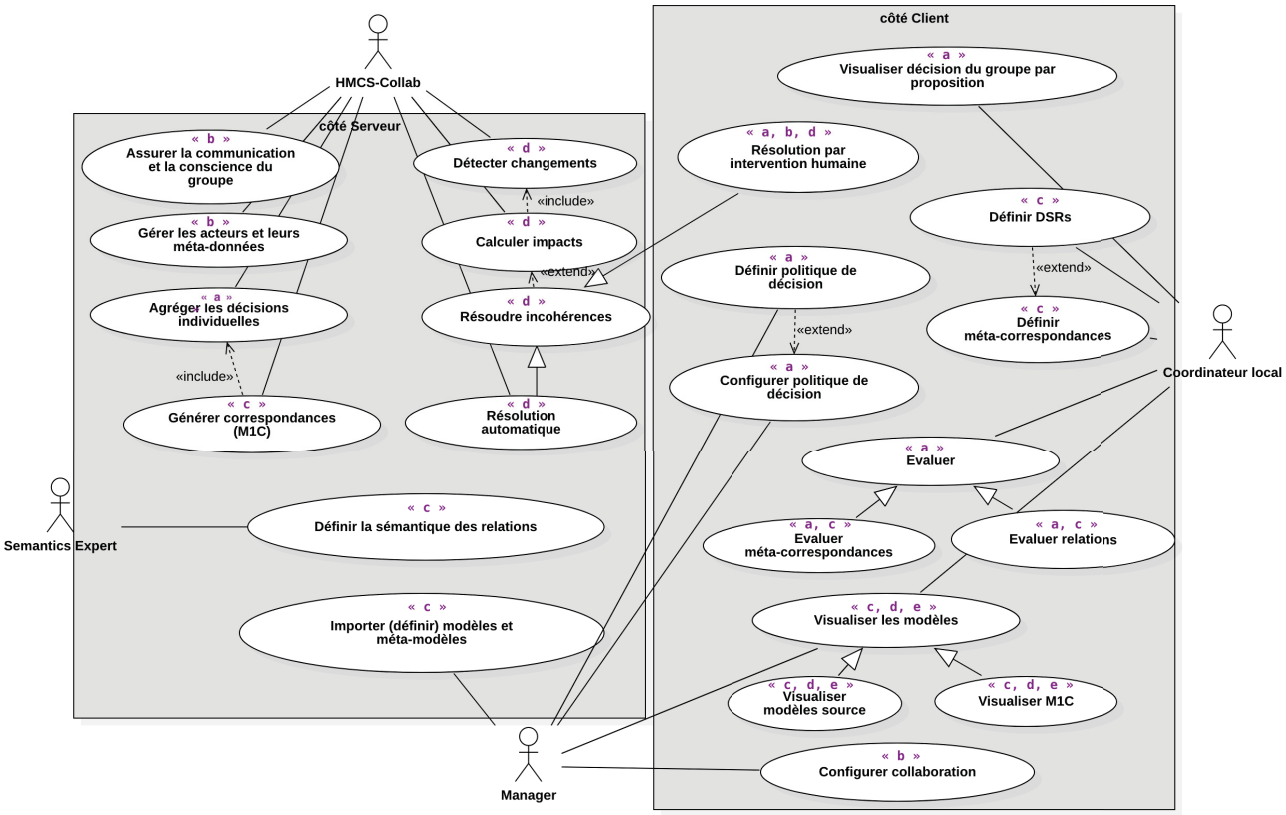
6.1 Introduction 129
6.2 Cas d'utilisation de HMCS-Collab 130
6.3 Modules support à la collaboration 130
6.3.1 Decision Making Tool (DMT) 131
6.3.2 Collaboration Tool (CollabT) 133
6.4 Modules support à l'alignement de modèles 134
6.4.1 Matching Tool (MT) 134
6.4.2 Consistency Management Tool (CMT) 134
6.4.3 Transformation Tool (TT) 135
6.5 Environnement de mise en oeuvre 136
6.5.1 Architecture globale de HMCS-Collab 136
6.5.2 Environnement de développement 137
6.5.2.1 Eclipse Modeling Project (EMP) 138
6.5.2.2 Eclipse Communication Framework (ECF) 140
6.5.2.3 Technologies web 141
6.6 Conclusion 141

6.1 Introduction

Ce chapitre présente l'outil développé pour supporter l'approche proposée dans cette thèse. L'objectif est de montrer la pertinence des concepts et de valider la démarche introduite dans les chapitres précédents.
HMCS-Collab est une extension de Heterogeneous Matching and Consistency-management Suite (HMCS) [El Hamlaoui et al., 2014, 2019]. Il a été conçu sur la plate-forme Eclipse et fournit un ensemble de plug-ins répondant aux fonctionnalités d'alignement et de prise de décision collaborative.
La deuxième section de ce chapitre présente les cas d'utilisation de l'outil. Dans la section 6.3 nous présentons les modules de HMCS-Collab dédiés à l'aspect collaboratif et qui constituent le coeur de cette thèse. La section 6.4 présente les modules support à l'alignement et la section 6.5 décrit en détail l'environnement de mise en oeuvre de l'outil, tandis que la section 6.6 conclut ce chapitre.

6.2 Cas d'utilisation de HMCS-Collab

HMCS-Collab se compose de deux parties : l'une concernant le *côté serveur*, l'autre *le côté client*. La Figure 6.1 illustre les cas d'utilisation (au sens UML) de ces deux parties. Nous distinguons quatre acteurs dont trois acteurs humains (le manager, le coordinateur local (concepteur) et l'expert en sémantique) et un acteur logiciel (HMCS-Collab lui-même) qui exécute des tâches automatiques, comme le raffinement de méta-correspondances par exemple.
Pour chaque cas d'utilisation de la Figure 6.1, les modules assurant la fonctionnalité sont repérés avec des lettres (allant de « a » à « e »). Ainsi les fonctionnalités des modules Decision Making Tool (DMT), Collaboration Tool (CollabT), Matching Tool (MT), Consistency Management Tool (CMT) et Transformation Tool (TT) sont respectivement repérées par les lettres a, b, c, d et e. Les fonctionnalités assurées par chaque module sont détaillées dans les sections suivantes.



(a) : Decision Making Tool (DMT), **(b)** : Collaboration Tool (CollabT), **(c)** : Matching Tool (MT), **(d)** : Consistency Management Tool (CMT) et **(e)** : Transformation Tool (TT)

FIGURE 6.1 – Diagramme des cas d'utilisation de HMCS-Collab.

6.3 Modules support à la collaboration

La Figure 6.2 présente les modules développés dans cette thèse et qui concernent l'aspect collaboratif. L'architecture globale de HMCS-Collab est présentée dans la section 6.5.1.

— **Decision Making Tool (DMT)** : implante un ensemble de politiques de prise de décision et la mise en œuvre de leurs méthodes d'agrégation. Il est divisé en deux sous-modules : Decision-policies Repository Tool (DRT) et Decision Aggregator Tool (DAT).

— **Collaboration Tool (CollabT)** : assure les mécanismes de collaboration (par exemple, la communication et la gestion de groupe, la sensibilisation du groupe, etc.) via un outil de communication (CommT), un outil de gestion de groupe (GMT) et un outil de sensibilisation de groupe (GAT).

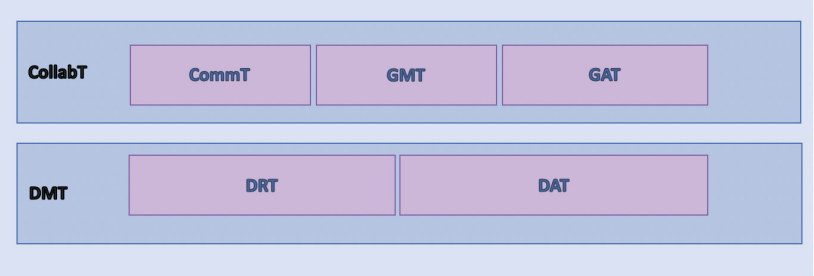


FIGURE 6.2 – Modules de HMCS-Collab dédiés au support de la collaboration.

6.3.1 Decision Making Tool (DMT)

Ce module a pour objectif d'aider à l'élaboration des décisions collectives. Les besoins pour ce module correspondent aux cas d'utilisation repérés par la lettre « a » sur la Figure 6.1 et qui sont pour rappel :

— Définition et stockage de politiques de prise de décision ;

— Agrégation des décisions individuelles afin de constituer la décision du groupe pour chaque proposition.

Le module DMT se compose de deux sous-modules : DRT et DAT.
DRT est un dépôt de politiques de prise de décision (Decision Making Policies (DMP)).
DAT implémente les méthodes d'élaboration des décisions collectives correspondant aux DMP.
La Figure 6.3 détaille le fonctionnement du module DMT. En plus du dépôt DMP, le module DRT exploite les données (DB) des utilisateurs (UDB), les propositions élaborées (PDB) et leurs évaluations (EDB). Ces quatre dépôts de données sont accessibles respectivement à travers quatre gestionnaires : UDB Manager, DMP Manager, PDB Manager et EDB Manager.

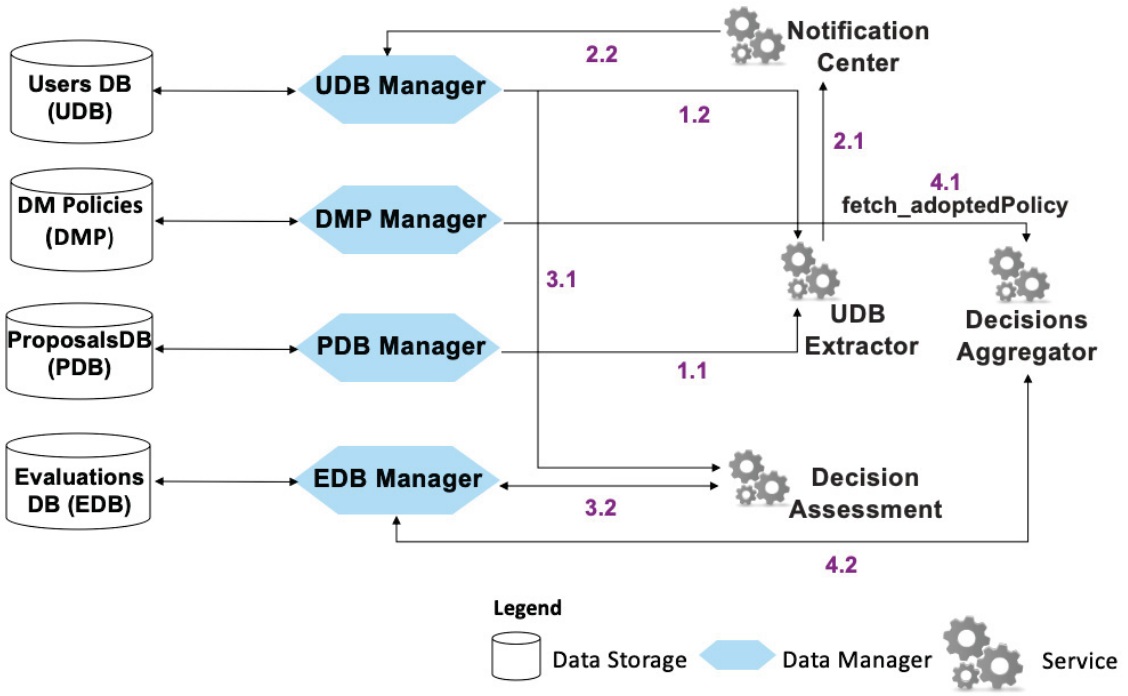


FIGURE 6.3 – Enchaînement fonctionnel de DMT.

L'enchaînement fonctionnel du DMT est le suivant :

— UDB extractor extrait pour chaque proposition (1.1) la liste des utilisateurs concernés (1.2) ;

— Cette liste est transférée vers *Notification Center* (2.1) qui notifie les utilisateurs concernés (2.2) ;

— Les utilisateurs évaluent individuellement les propositions et fournissent les décisions (3.1) par le service d'évaluation des décisions. Ces décisions modifient EDB via *EDB Manager* (3.2) ;

— *Decisions Aggregator* produit une décision de groupe en combinant les décisions individuelles (4.2) conformément à la politique adoptée (4.1).

Decisions Aggregator et *Decision Assessment* sont les services fondamentaux du DAT. *Notification Center* et *UDB Extractor* sont des services extérieurs à DMT ; ils font partie du module CollabT (section 6.3.2).
Les politiques de décision étant configurables, le module DMP permet de les adapter en attribuant des valeurs à leurs caractéristiques qui ne sont pas figées. Ceci est implémenté en respectant les bonnes pratiques de conception étant donné que les politiques de décision exploitent les patrons de conception *State* et *Observer* [GAMMA, 1995].
Le patron *State* est utilisé pour distinguer les états communs aux politiques de décision de ceux qui sont spécifiques. Les états communs sont des états adaptés à toutes les politiques de décision qu'elles soient itératives ou à cycle unique, tandis que les états spécifiques concernent les politiques de décision à un seul tour.
Le patron *Observer* est utilisé pour sélectionner et informer les décideurs concernés pour chaque proposition. Il permet à un objet, appelé *Observable* sur la Figure 6.4, de maintenir une liste de ses dépendants, appelés *Observer*(s), et de les avertir automatiquement de tout changement d'état, généralement en appelant l'une de leurs méthodes (*update()*). La responsabilité des *observer*(s) est de s'inscrire/désinscrire de *Observable* et de mettre à jour leurs états lorsqu'ils sont notifiés.

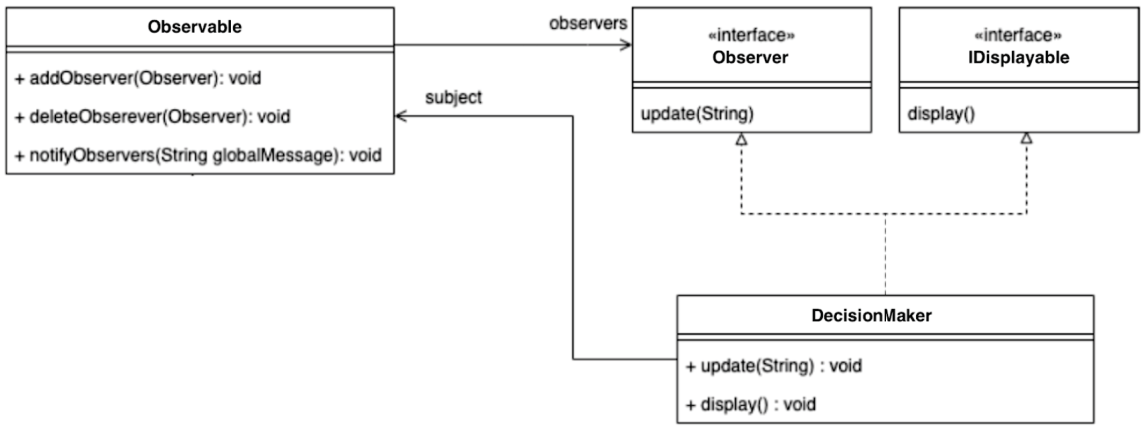


FIGURE 6.4 – Utilisation du patron *Observer* pour notifier les décideurs d'une proposition.

6.3.2 Collaboration Tool (CollabT)

CollabT permet de gérer les acteurs et d'assurer les communications entre eux. Les besoins pour ce module correspondent aux cas d'utilisation repérés par la lettre « b » dans la Figure 6.1 :

— Assurer la communication des acteurs (communication instantanée et différée) ;

— Assurer la conscience du groupe par rapport à l'avancement des collaborations et l'attente de leur participation ;

— Faciliter la gestion et l'organisation des acteurs en permettant d'attribuer et de consulter des méta-données les concernant (disponibilité, expertise, etc).

Le module CollabT se compose de trois sous-modules : CommT, GAT et GMT.
CommT (*Communication Tool*) assure la communication des acteurs.
GAT (*Group Awareness Tool*) assure la sensibilisation des acteurs par rapport à la présence d'autres acteurs et aux différents types de notifications. Il intègre un système de notifications (*Notification Center*) concernant les étapes clés des processus d'alignement et de prise de décision, i.e., l'attribution d'un rôle, l'attente de propositions, l'attente d'évaluation des propositions, l'existence d'un commentaire associé à une proposition, l'achèvement de la collaboration, etc.
GMT (*Group Management Tool*) assure la gestion des groupes d'acteurs pour connaître les acteurs disponibles, leurs rôles et pouvoir les contacter.
La Figure 6.5 détaille l'enchaînement fonctionnel du module CollabT.

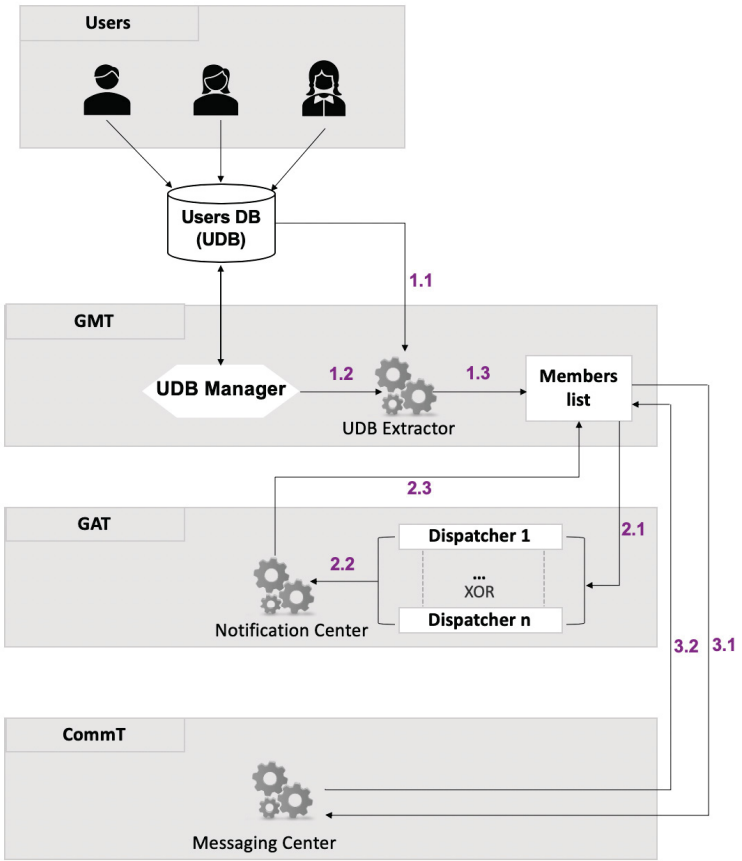


FIGURE 6.5 – Enchaînement fonctionnel de CollabT.

— *UDB extractor* extrait à partir de *UDB manager* (1.2) et de *Users DB* (1.1) les informations sur les utilisateurs, afin de constituer la liste des membres concernés par la collaboration (1.3) ;

— Cette liste est envoyée aux expéditeurs des notifications (dispatcher 1, ..., dispatcher n) (2.1). Un seul dispatcher est activé en fonction de l'événement associé à la notification (2.2). Une

fois qu'un dispatcher est activé, le *Notification Center* transmet la notification aux acteurs concernés qui satisfont les contraintes de sélection du dispatcher (2.3) ;

— Les membres concernés répondent à la notification ; ils transmettent leurs messages au *Messaging Center* (3.1) qui se charge de les transmettre ensuite aux autres acteurs (3.2) par *broadcast, multicast* ou bien message ciblé.

6.4 Modules support à l'alignement de modèles

Des modules support à l'alignement ont été initialement proposés dans HMCS [El Hamlaoui et al., 2014, 2019] : MT, CMT et TT. Dans cette thèse, nous avons étendu les fonctionnalités des modules MT et CMT en intégrant le volet collaboratif.

6.4.1 Matching Tool (MT)

Les besoins pour ce module correspondent aux cas d'utilisation repérés par la lettre « c » dans la Figure 6.1. Ils concernent généralement la définition des (méta-)modèles source, l'élaboration des méta-correspondances et la génération du Modèle de Correspondances (M1C).
Le module MT contient deux sous-modules : *Assisted Matching Tool (AMT)* et *Refining Tool (RT)*. AMT fait appel à différents acteurs humains pour élaborer les méta-correspondances, tandis que les fonctionnalités de RT sont opérées automatiquement par l'outil.
Le rôle de RT est de propager les méta-correspondances pour produire le M1C. La première fonctionnalité de RT est la *duplication* qui consiste à créer les correspondances entre les instances des méta-éléments qui participent dans une même méta-correspondance. La deuxième fonctionnalité est le *filtrage* qui consiste à utiliser les expressions sémantiques des types de relation pour filtrer les correspondances créées en éliminant celles qui ne sont pas valides vis-à-vis de leur sémantique.
Dans cette thèse, nous avons enrichi le processus de filtrage en ajoutant une phase de pré-processing comme dans les travaux de Vijayarani et al. [2015] et Eyal Salman et al. [2018] et en exploitant les bases de connaissances ConceptNet [1] et WordNet [2] tel qu'illustré sur la Figure 6.6. En effet, nous partons de l'hypothèse d'existence d'une relation nommée « R » entre n éléments (termes ou textes courts).
Le processus de validation des relations sémantiques prend en entrée la liste des termes candidats à être reliés par la relation « R » et aussi ladite relation. Ensuite, les concepts candidats suivent une étape de pré-processing qui consiste à éliminer les caractères spéciaux, les majuscules afin que les termes soient exploitables par les différentes bases de connaissances utilisées.
Une fois le pré-processing terminé, les termes (modifiés par l'étape de pré-processing) et la relation « R » entrent dans l'étape de validation qui se charge de vérifier si la relation est valable entre ces termes. Cette vérification se base sur la sémantique associée à la relation ; dans l'annexe C.2, nous présentons les sémantiques proposées pour les relations utilisées dans ce manuscrit.

6.4.2 Consistency Management Tool (CMT)

Les besoins pour ce module correspondent aux cas d'utilisation repérés par la lettre « d » dans la Figure 6.1. Ce module permet d'identifier les changements et d'analyser puis traiter leurs impacts sur les modèles source et le modèle de correspondances.
Le module CMT comporte trois sous-modules : *Change Detector Tool (CDT), Consistency Checker Tool (CCT)* et *Inconsistency Resolver Tool (IRT)*.
Le module CDT permet de capturer les changements qui ont été listés dans la section 5.5.2. Pour ce faire, ce module se base sur les UUID des éléments de modèles et méta-modèles. Les modèles source ne sont pas forcément élaborés par le même outil de modélisation, le fait qu'ils soient importés sur EMF permet d'avoir un cadre unifié pour les gérer ; cependant, pour tracer les changements

1. http://conceptnet.io
2. https://wordnet.princeton.edu

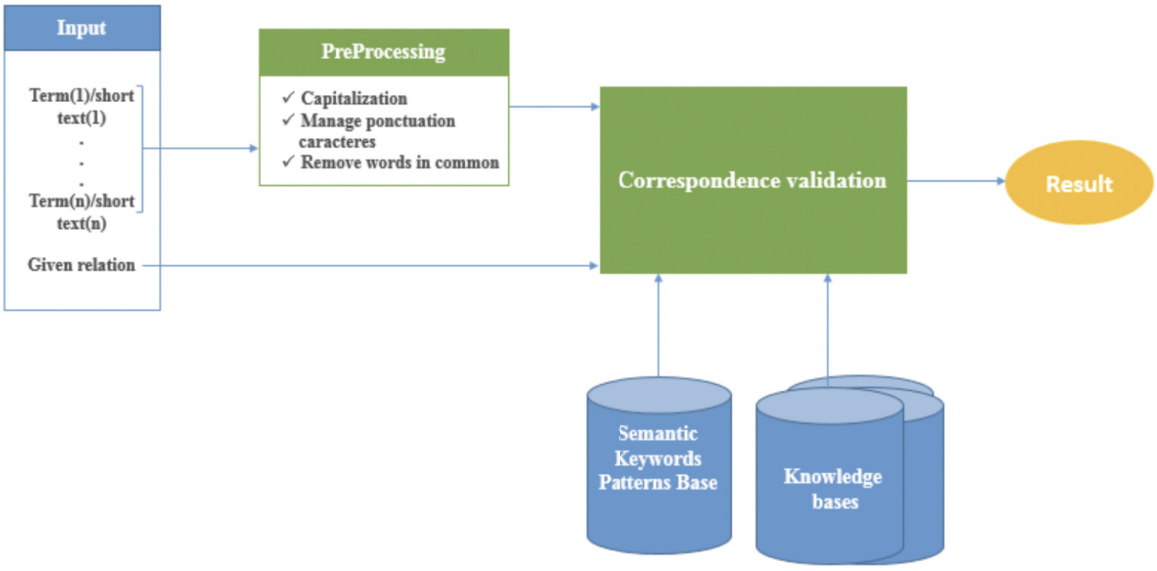


FIGURE 6.6 – Déroulement général du processus de filtrage des correspondances.

composites de type *Move*, *Pull up*, *Push down*, *Inline class* et *Flatten class*, nous utilisons une interface web pour définir les éléments concernés par ces changements.
Le module CCT vérifie l'impact de tout changement supporté par l'approche en lui appliquant l'algorithme 1 de l'annexe B.1 ; cet algorithme, dont le rôle a été présenté dans le chapitre 5, permet d'attribuer un ordre de traitement selon le nombre d'éléments impactés, de façon directe ou indirecte, par un changement.
Le module IRT applique les résolutions fournies dans le catalogue de résolutions (annexe B.3). Dans le cas où une intervention humaine est requise il fait appel aux modules DMT et CollabT, pour que les coordinateurs locaux puissent résoudre l'incohérence, ses impacts sur le modèle de correspondances et, par transitivité, sur les modèles source.

6.4.3 Transformation Tool (TT)

Ce module a pour objectif d'assurer la visualisation des modèles sous deux formats, textuel et graphique, qu'il s'agisse des modèles source ou des modèles de correspondances (M2C et M1C). La visualisation graphique inclut la visualisation en tant qu'arborescence *Ecore* ou en tant que diagramme.
Le module TT fournit deux sous-modules pour les transformations *Model-to-Text (M2T)* et *Text-to-Model (T2M)*. Ainsi, chaque modèle peut avoir plusieurs représentations synchronisées.
Le module M2T a pour objectif de « sérialiser », à travers la technologie Java Emitter Templates (JET), le modèle graphique afin de produire sa représentation textuelle.
Le module T2M a pour objectif de retrouver le modèle graphique en analysant la représentation textuelle. T2M a été codé en Java en s'appuyant sur les bibliothèques de gestion de modèles.
Notons que le module TT est utilisé pour visualiser les modèles sur le côté serveur, tandis que des composants de PrimeFaces [3] sont exploités dans HMCS-Collab sur le côté client pour visualiser les modèles (e.g., *Diagram*, *Mindmap*, *HorizontalTree*, etc.).

3. https ://www.primefaces.org/showcase/ui/data/

6.5 Environnement de mise en oeuvre

6.5.1 Architecture globale de HMCS-Collab

La Figure 6.7 présente l'architecture globale de HMCS-Collab. Elle inclut à la fois les modules développés et les technologies utilisées.

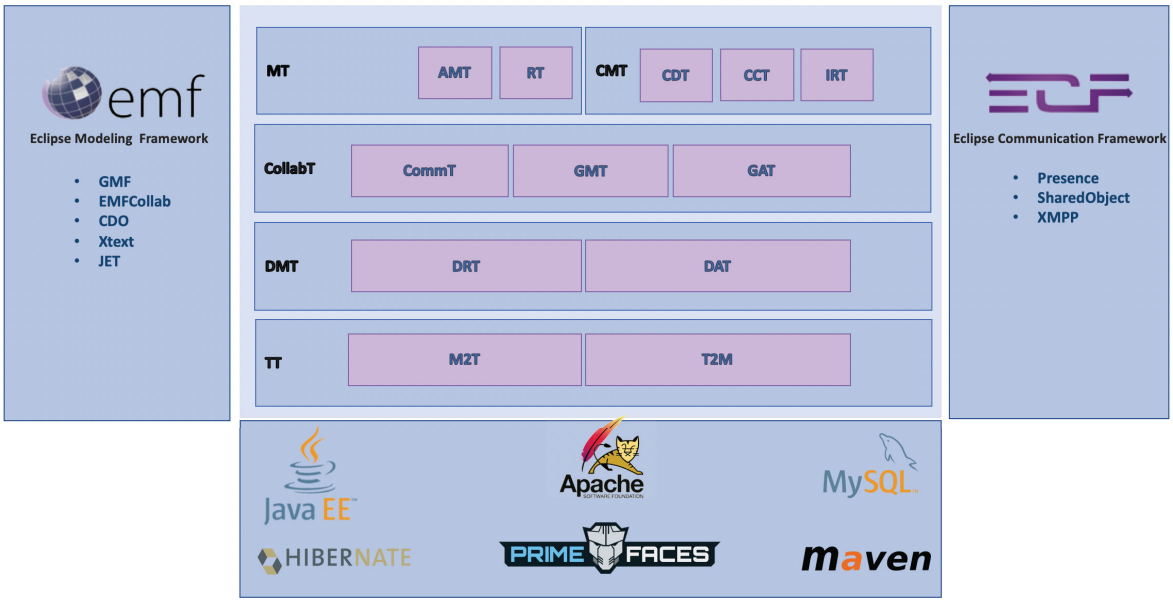


FIGURE 6.7 – Architecture globale de HMCS-Collab.

Les modules de HMCS-Collab sont divisés en deux catégories : (i) modules pour l'alignement et (ii) modules pour la prise de décision collaborative. Les modules des deux catégories communiquent et interagissent ; la Figure 6.8 illustre un diagramme des composants de HMCS-Collab qui décrit les interactions des modules en spécifiant le sens pour chaque communication. Ainsi les modules pour l'alignement, en l'occurrence MT et CMT, utilisent les modules DMT, CollabT et TT :

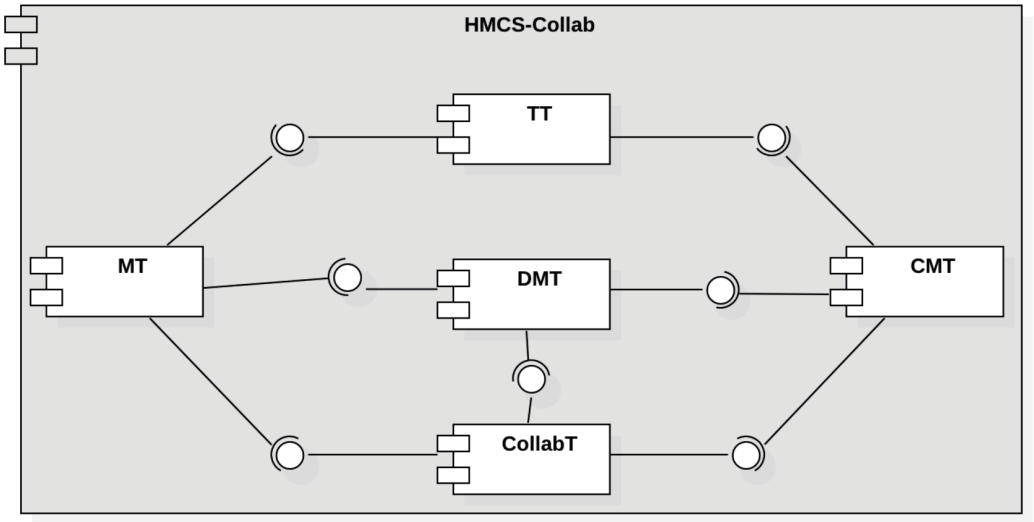


FIGURE 6.8 – Diagramme des composants de HMCS-Collab.

— MT et CMT utilisent le module TT pour pouvoir passer d'une représentation textuelle à une représentation graphique des différents modèles et méta-modèles ;

— MT et CMT utilisent le module DMT pour dérouler les prises de décisions collaboratives et produire des décisions du groupe, que cela soit pour l'élaboration du modèle de correspondances (module MT) ou pour le maintien de la cohérence (module CMT) ;

— MT et CMT utilisent le module CollabT pour assurer les communications entre les acteurs et gérer les groupes d'acteurs. Notons que le module DMT utilise lui aussi le module CollabT.

6.5.2 Environnement de développement

Le développement des cinq modules de HMCS-Collab repose sur des outils de deux projets issus de l'environnement Eclipse, à savoir *Eclipse Modeling Framework (EMF)* du projet *Eclipse Modeling Project (EMP)* et *Eclipse Communication Framework (ECF)* du projet *Runtime* comme présenté sur la Figure 6.9.

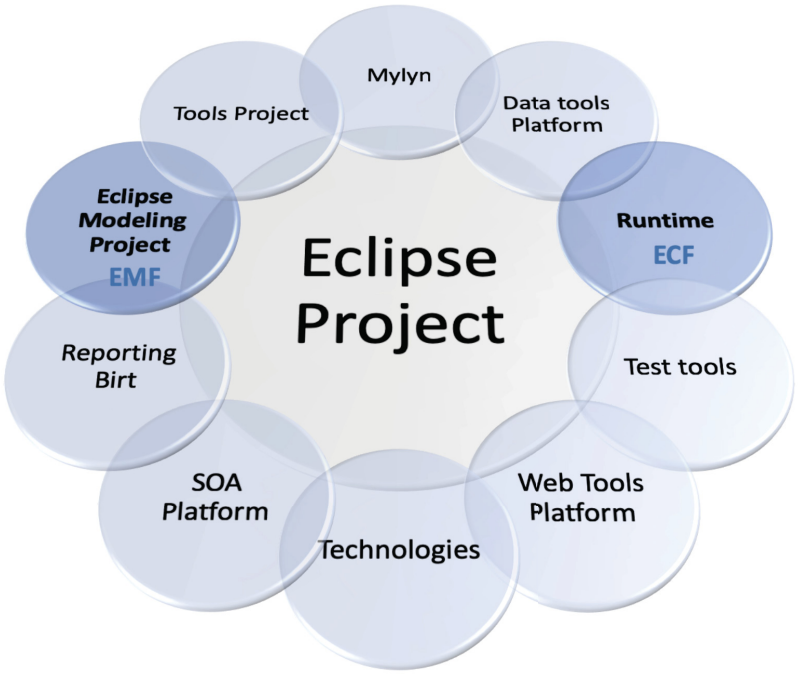


FIGURE 6.9 – Projets Eclipse du premier niveau.

EMP sert à la mise en œuvre de technologies de développement pilotées par les modèles dans le cadre du projet Eclipse. Les outils issus de ce projet et qui ont été utilisés pour mettre en oeuvre HMCS-Collab sont présentés dans la section 6.5.2.1.
Runtime a été conçu pour promouvoir les efforts d'exécution dans la communauté Eclipse. Son objectif est de fournir un modèle de composant uniforme dans une grande variété d'environnements informatiques, ce qui permet aux développeurs de se concentrer sur le problème métier et de disposer de nombreuses options d'architecture système au moment du déploiement. ECF qui fait partie de ce projet a été utilisé dans HMCS-Collab et fait l'objet de la section 6.5.2.2.
HMCS-Collab exploite aussi les technologies de développement d'applications web puisqu'il fournit des IHM dédiées pour le déroulement de la prise de décision. Parmi ces technologies, nous notons : *Jakarta EE, Hibernate, Maven, Tomcat, PrimeFaces, MySQL*. Un aperçu de ces technologies est donné dans la section 6.5.2.3

6.5.2.1 Eclipse Modeling Project (EMP)

Le projet de modélisation Eclipse se concentre sur l'évolution et la promotion des technologies de développement basées sur des modèles au sein de la communauté Eclipse en fournissant un ensemble unifié de cadres de modélisation et d'outils de mise en œuvre. Son cadre principal est EMF avec un ensemble de technologies et solutions développées autour de ce dernier dont EMFCollab, Connected Data Objects (CDO), Xtext, JET que nous présentons dans la suite de cette section.

Eclipse Modeling Framework (EMF)
EMF permet d'intégrer les concepts de l'IDM. Il fait partie du projet Eclipse Modeling Project. EMF est un framework de modélisation, une infrastructure de génération de code et des applications basées sur des modèles de données structurées. C'est l'outil le plus utilisé pour la modélisation, dans un cadre IDM, par les chercheurs et les académiques malgré les difficultés du maintien des composants nécessaires et des plug-ins dans un état cohérent [CICCOZZI et al., 2018b]. Partant d'une spécification décrite généralement sous la forme d'un modèle en XMI [4], EMF fournit des outils permettant de produire des classes Java représentant le modèle avec un ensemble de classes pour adapter les éléments du modèle afin de pouvoir les visualiser, les éditer avec un système de commandes et les manipuler dans un éditeur.
EMF est fondé sur trois briques tel que présenté sur la Figure 6.10 :

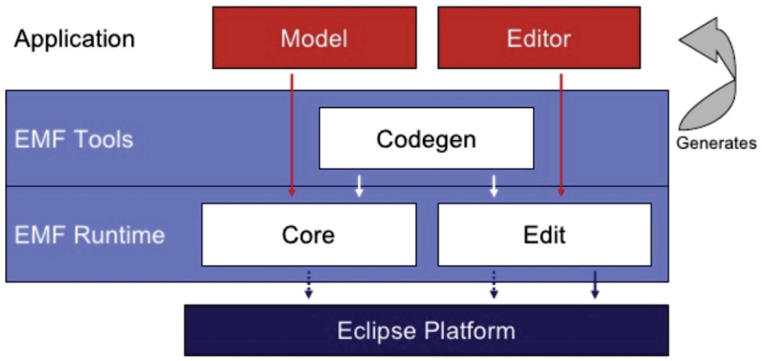


FIGURE 6.10 – Architecture de EMF. [STEINBERG, 2010]

— **EMF.Core**, le framework EMF de base : il comprend un méta-modèle (Ecore) pour décrire les modèles (syntaxe abstraite) et le support d'exécution des modèles, un support de persistance avec la sérialisation XMI, et une API pour manipuler des objets EMF ;
— **EMF.Edit**, le framework d'édition : il comprend des classes génériques réutilisables pour la construction des éditeurs pour les modèles EMF ;
— **EMF.Codegen**, le générateur de code EMF : il permet de générer ce qui est nécessaire pour construire un éditeur complet d'un modèle EMF. Il comprend une interface graphique à partir de laquelle les options de génération peuvent être spécifiées, et des générateurs invoqués.

4. https://www.omg.org/spec/XMI/ISO/19503/PDF

EMFCollab
EMFCollab est une solution légère permettant à plusieurs utilisateurs de modifier simultanément un même modèle EMF. EMFCollab a une architecture client-serveur. Le client peut être intégré à n'importe quel éditeur Eclipse basé sur EMF. La Figure 6.11 montre comment EMfCollab fonctionne :

— Il stocke sur le serveur la copie principale du modèle.

— Il stocke sur le client une copie esclave du modèle maître dans la mémoire. Ce modèle est synchronisé sur le réseau en sérialisant et en distribuant les commandes qui le concernent. Tous les clients voient la même copie principale sur une seule pile de commandes.

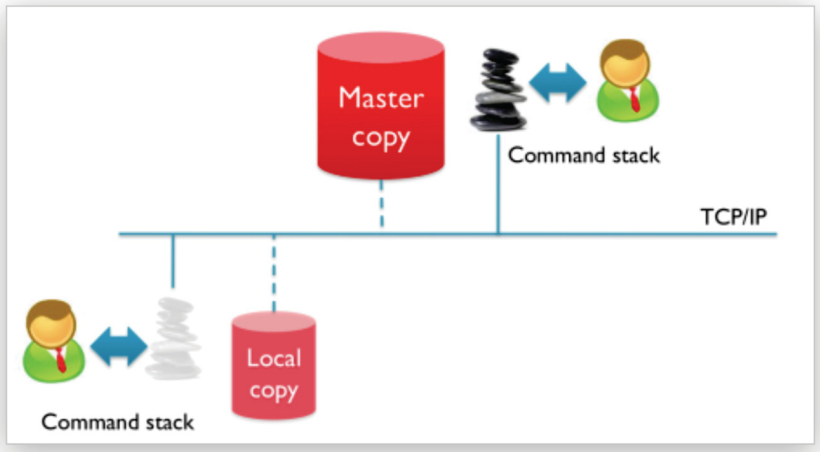


FIGURE 6.11 – Fonctionnement de EMFCollab.[SCHMIDT, 2011]

Xtext
Xtext est un langage de développement d'éditeur textuel s'appuyant sur EMF pour la manipulation de modèles. Il fournit un langage de définition de grammaire similaire à EBNF [WIRTH, 1996] pour la description de la syntaxe du langage.
L'éditeur Xtext généré propose un ensemble de fonctionnalités : coloration syntaxique, complétion automatique, indentation automatique, navigation, recherche à travers le *outline* et finalement la validation par l'intermédiaire d'un langage d'expression de contraintes de type OCL.

Connected Data Objects
Connected Data Objects [5] (CDO) est à la fois une technologie pour les modèles EMF distribués et une solution pour le *mapping* objet-relationnel basée sur un serveur.
CDO permet d'améliorer et mettre à jour les modèles existants et possède une architecture à trois niveaux prenant en charge les applications client basées sur EMF, il comprend un serveur centralisé des modèles qui exploite différents types de stockage de données en back-end, tels que des bases de données relationnelles, des bases de données objet et des systèmes de fichiers comme illustré sur la Figure 6.12. Le protocole de communication client/serveur par défaut est implémenté avec la plate-forme de signalisation Net4j [6].

5. https://wiki.eclipse.org/CDO
6. https://wiki.eclipse.org/Net4j

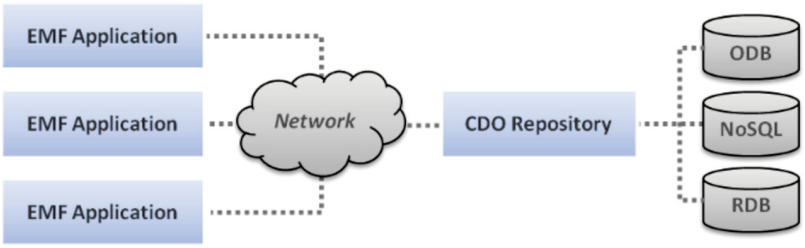


FIGURE 6.12 – Fonctionnement de CDO. [STEPPER, 2010]

JET
Java Emitter Templates (JET [7]) est un outil de génération de code source. Il permet d'utiliser une syntaxe de type JSP qui facilite l'écriture de modèles exprimant le code que l'on souhaite générer. JET est un moteur de modèle générique qui permet de générer un code source SQL, XML, Java et d'autres sorties issues de modèles. Il fait partie du plug-in org.eclipse.emf.codegen inclus dans le téléchargement d'exécution d'EMF.

6.5.2.2 Eclipse Communication Framework (ECF)

ECF fait partie du projet Runtime. Il fournit un cadre pour la messagerie et les communications dans des applications et des services. Il intègre une implémentation modulaire et indépendante du transport et des normes OSGi Remote Services et Remote Service Admin.
ECF s'appuie sur les réseaux de communication existants. Il propose des fournisseurs pour *XMPP*[8], *Jabber*[9], ou *IRC*[10] et implémente ses propres mécanismes via un adaptateur générique.
Le noyau de ECF (Figure 6.13) comprend un cadre extensible : l'API *SharedObject* pour les applications distribuées créées à l'aide du modèle MVC, car elles doivent partager et synchroniser le ou les modèles sur le réseau. Ainsi, l'API *SharedObject* offre un moyen de partager des données au niveau de l'application, sans avoir à se soucier des détails spécifiques du protocole. Les autres

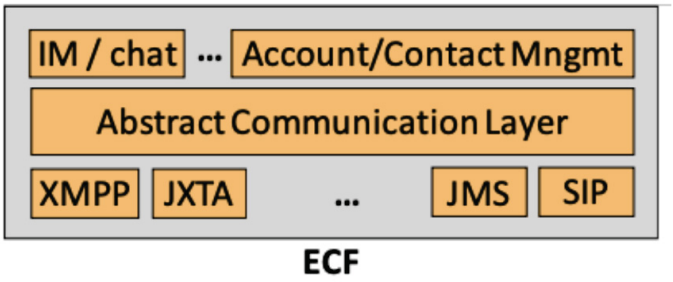


FIGURE 6.13 – Quelques plug-ins natifs de ECF. [CALEFATO et al., 2007]

composants notables, disponibles dans ECF, incluent l'API *Presence*, qui gère les événements de présence, l'API de transfert de fichier pour le partage de contenu entre utilisateurs distants et l'API *Remote Services*, qui fournit un mécanisme de type RPC pour les appels de procédure distants.
Toutes ces API fournissent une couche d'abstraction de haut niveau qui permet aux applications basées sur ECF de prendre en charge plusieurs protocoles, en ignorant les détails d'implémentation gérés de manière transparente par l'infrastructure sous-jacente.

7. https://www.ibm.com/support/knowledgecenter/fr/SSQ2R2_9.1.1/org.eclipse.emf.doc/tutorials/jet1/jet_tutorial1.html
8. https://xmpp.org
9. https://www.jabber.org
10. http://www.irchelp.org

6.5.2.3 Technologies web

L'outil HMCS-Collab côté client se base sur des technologies web pour faciliter la collaboration des acteurs (concepteurs locaux des modèles source) et leur cacher les aspects techniques liés à l'exploitation de EMF. Le côté client de HMCS-Collab est une application web, développée en java, et notamment en JSF, et utilisant différents outils dont les plus représentatifs sont les suivants :

— **MySQL** [11] : le SGBD utilisé pour la persistance des données. L'ensemble des données de l'application sont stockées dans une base de données MySQL.

— **Apache tomcat 9** [12] : un conteneur web libre de servlets et JSP. Issu du projet Jakarta [13], il est paramétrable par des fichiers XML, et inclut des outils pour la configuration et la gestion. Il comporte également un serveur HTTP.

— **PrimeFaces** [14] : une librairie open source de composants graphiques pour les applications JSF.

— **Hibernate** [15] : une solution open source de type ORM qui facilite le développement de la couche persistance d'une application. Hibernate permet donc de représenter une base de données en objets Java et vice versa. Nous utilisons d'autres composants indispensables au fonctionnement d'une application web JSF.

6.6 Conclusion

Le travail présenté dans ce chapitre permet de fournir à l'utilisateur un environnement pour l'alignement collaboratif de modèles hétérogènes. Le prototype développé supporte le processus de l'approche CAHM en intégrant les volets *alignement* et *prise de décision collective*.
Nous avons présenté au début du chapitre les cas d'utilisation du prototype HMCS-Collab. Ensuite, nous avons détaillé le fonctionnement des modules développés. Après cela, nous avons mis l'accent sur l'architecture globale de l'outil HMCS-Collab et les principales technologies utilisées pour le mettre en oeuvre.
Le framework tire profit des fonctionnalités du projet de modélisation d'Eclipse (EMF) notamment en terme de minimisation du temps nécessaire pour développer le prototype, comme par exemple la génération du code de base pour l'éditeur EMF et l'éditeur textuel. La combinaison projet Eclipse et application web offre l'avantage de cacher la technicité d'EMF aux concepteurs et leur permet de focaliser leur travail sur le déroulement de l'alignement collaboratif grâce à l'application web, tandis que le manager et l'expert en sémantique interviennent lorsque des aspects sémantiques liés au domaine d'application sont nécessaires.
Des pistes sont identifiées pour améliorer les fonctionnalités du prototype. Nous envisageons tout d'abord d'assurer la détection des changements composites qui ne sont pas identifiables par EMF ; aussi nous prévoyons de développer des mécanismes pour pouvoir importer les modèles source directement et exploiter leurs UUID même s'ils sont issus d'outils de modélisation distincts. Concernant le volet collaboratif, nous comptons mettre en place des mécanismes de discussion instantanée au sein de l'application web.

Dans le chapitre suivant, nous illustrons l'application de l'approche avec l'outil à travers une étude de cas modélisant un service d'urgence d'un hôpital.

11. https://www.mysql.com/fr/
12. https://tomcat.apache.org/download-90.cgi
13. https://jakarta.ee
14. https://www.primefaces.org
15. https://hibernate.org

Chapitre 7

Validation expérimentale

Sommaire

7.1 Introduction 144
7.2 Cas d'étude : Service d'urgence d'un hôpital 144
7.2.1 Vue d'ensemble du cas d'étude 144
7.2.2 Méthodologie de mise en oeuvre du cas d'étude 145
7.2.2.1 Étape 1 : Élaboration du cas d'étude « SU » 146
7.2.2.2 Étape 2 : Application de CAHM au cas d'étude « SU » 147
7.2.3 Description des points de vue du cas d'étude 147
7.2.3.1 Spécification du point de vue données (SD) 147
7.2.3.2 Spécification du point de vue processus (BP) 149
7.2.3.3 Spécification du point de vue fonctionnel (ER) 150
7.2.3.4 Spécification du point de vue organisationnel (MAS) 152
7.3 Application de CAHM - phase 1 au système SU 154
7.3.1 Illustration de la problématique de mise en correspondances collaborative 154
7.3.2 Configuration du système SU sur l'outil HMCS-Collab 155
7.3.3 CAHM - phase 1 - Activité 3 : Définir les méta-correspondances 155
7.3.4 CAHM - phase 1 - Activité 4 : Générer le modèle de correspondances . . . 158
7.4 Application de CAHM - phase 2 au système SU 159
7.4.1 CAHM - phase 2 - Activité 1 : Détecter les changements 159
7.4.2 CAHM - phase 2 - Activité 2 : Calculer l'impact et l'ordre de traitement des changements 160
7.4.3 CAHM - phase 2 - Activité 3 : Traiter les incohérences 161
7.4.3.1 Changement N° 1 : Ajout du méta-modèle MAS 161
7.4.3.2 Changement N° 2 : Ajout du modèle MAS 162
7.4.3.3 Changement N° 3 : Renommage de la *Task :Control* du modèle BP 162
7.4.3.4 Changements N° 4, 5, 6, 7 et 8 : Ajout de quatre *Field* au modèle ER 163
7.5 Retours sur le cas d'étude 164
7.5.1 Synthèse 164
7.5.2 Menaces de validité 164
7.5.2.1 Validité interne 164
7.5.2.2 Validité externe 165
7.5.2.3 Validité de la conclusion 165
7.6 Conclusion 165

7.1 Introduction

Ce chapitre présente une validation expérimentale de notre approche CAHM à partir du cas d'étude du service d'urgence d'un hôpital. Il décrit l'application de l'approche sur le cas d'étude en exploitant l'outil HMCS-Collab. La section 7.2 décrit le cas d'étude, y compris la méthodologie de mise en oeuvre du cas d'étude et la spécification des méta-modèles et modèles représentant le service d'urgence. La section 7.3 illustre le déroulement de la phase de mise en correspondance collaborative, alors que la section 7.4 met en oeuvre la phase de maintien de la cohérence lors des évolutions des modèles. Avant de conclure le chapitre (section 7.6), nous présentons quelques retours sur l'expérimentation, notamment les menaces de validité (section 7.5).

7.2 Cas d'étude : Service d'urgence (SU) d'un hôpital

7.2.1 Vue d'ensemble du cas d'étude

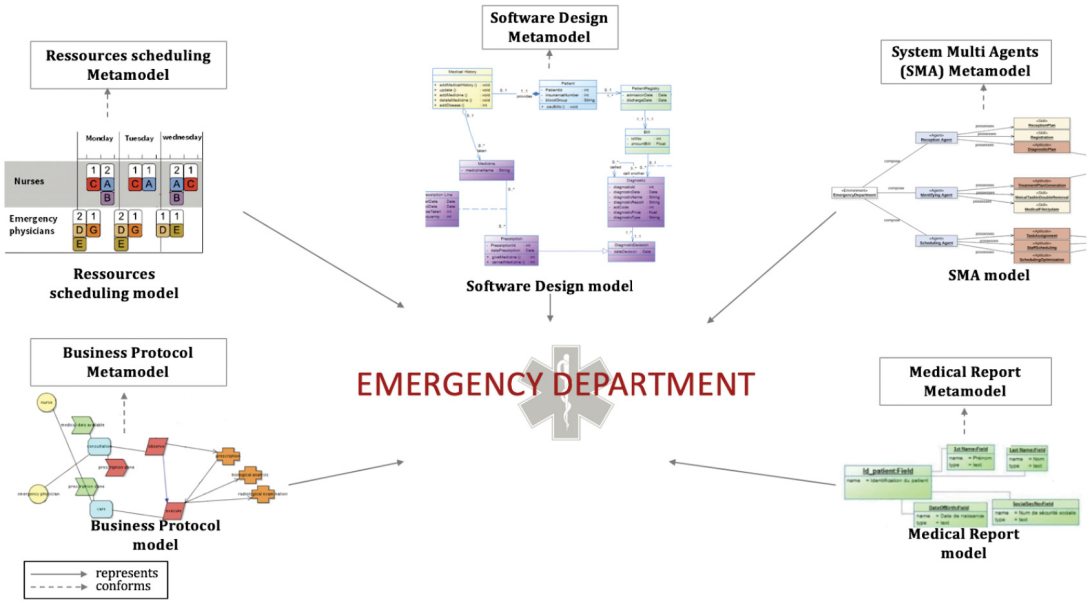


FIGURE 7.1 – Quelques points de vue métier sur le service d'urgence.

Un Service d'urgence (SU) représente une branche essentielle du système de santé dans tous les pays. Ce service nécessite des compétences particulières fournies par une approche multidisciplinaire où les points de vue sont complémentaires. La complexité du service d'urgence est due à deux facteurs majeurs :

- La complexité de son sous-système technique liée aux moyens matériels assurant :
  - Les examens médicaux (appareils, scanner, salles, etc.).
  - Le transport des patients (chariots, fauteuils, etc.).
  - Le suivi de la prise en charge des patients.
- La complexité de son sous-système social représenté par le personnel, les procédures et les protocoles de soin :
  - Le personnel non médical (ambulanciers, agents d'entretien et administratifs, etc.).
  - Le personnel soignant (hôtesses d'accueil, aides-soignantes, infirmières, médecins urgentistes, chirurgiens, anesthésistes, etc.).
  - Les procédures et protocoles de soins qui formalisent les activités des personnels et leurs modes d'intervention.

La gestion d'un tel système requiert donc une coordination entre les acteurs dès la phase de conception, afin que les différents modèles soient synchronisés et que, par la suite, le fonctionnement du service ne soit pas défaillant. Parallèlement à ceci, la gestion d'un tel système doit prendre en compte le changement du cadre régissant le domaine d'application. Par exemple, changement des lois, de la réglementation, de procédures d'exploitation ou même de contraintes de sécurité et de protection des données personnelles, etc. Ce type de changement étant transverse, l'évolution d'un modèle peut en impacter d'autres et si les modèles ne sont pas synchronisés, l'impact risque d'être difficilement repérable.
Dans cette étude, nous nous plaçons spécifiquement dans le contexte de l'urgence d'un hôpital public Français. Le cas d'étude a été développé en coopération avec des membres du personnel médical de cet hôpital. Nous avons identifié un ensemble représentatif de points de vue métier nécessaires pour le bon fonctionnement du SU.
La Figure 7.1 montre certains points de vue d'un système SU : protocole métiers (*Business Protocol*), planification des ressources (*Ressources Scheduling*), conception logicielle (*Software Design*), organisation des agents (*Multi Agent System*), maquette des rapports médicaux (*Medical Report*), etc.
Dans la suite de cette section, nous présentons d'abord la méthodologie de mise en oeuvre du cas d'étude ; ensuite, nous décrivons les méta-modèles et modèles des points de vue considérés pour représenter le SU.

7.2.2 Méthodologie de mise en oeuvre du cas d'étude

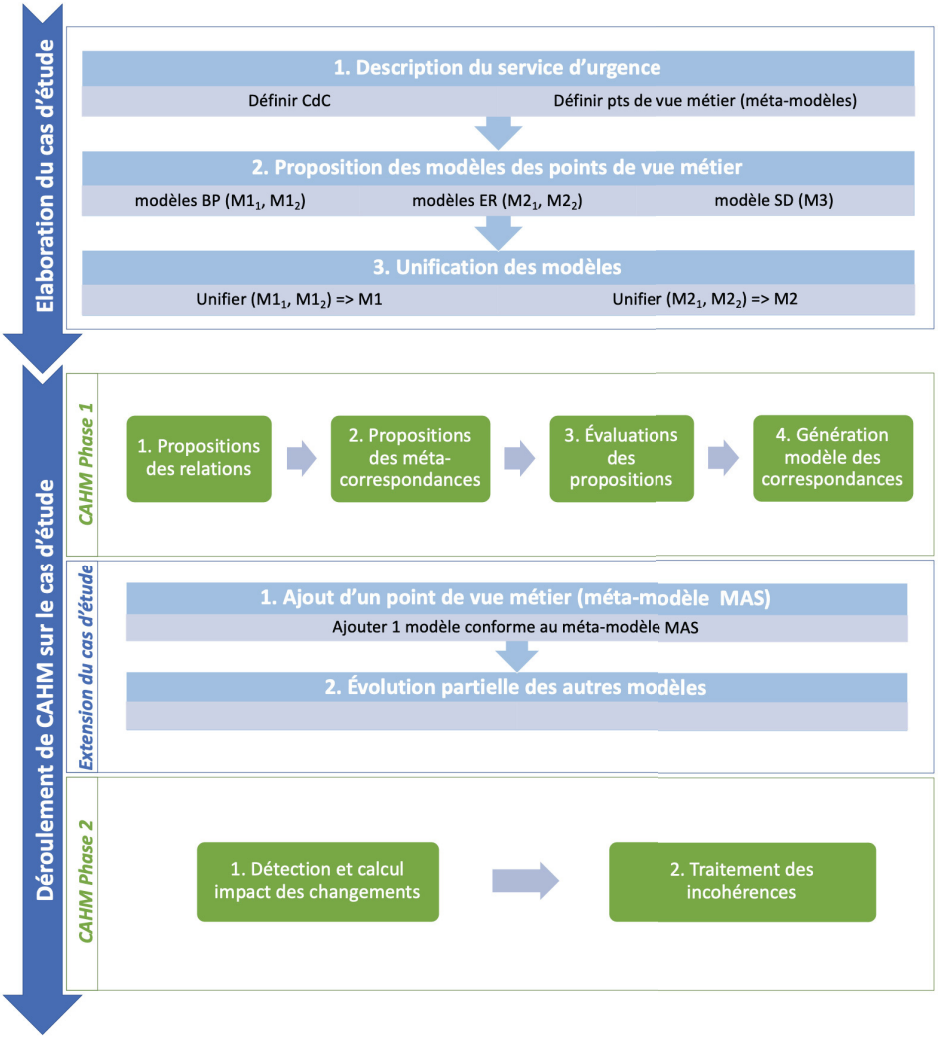


FIGURE 7.2 – Processus informel de validation de l'approche par cas d'étude.

L'objectif global de cette étude de cas est d'évaluer l'application de l'approche CAHM sur le service d'urgence (SU) d'un hôpital. Le choix du cas SU est justifié par le fait que ce système est relativement complexe, par l'existence de nombreux acteurs ayant des métiers distincts susceptibles d'interagir sur ce système en exploitant des modèles hétérogènes qu'il faut coordonner. Ainsi, nous visons d'élaborer une vue globale sur ces différents modèles en appliquant l'approche CAHM et en utilisant l'outil HMCS-Collab.
L'application de l'approche au cas d'étude est réalisée en deux grandes étapes tel que résumé sur la Figure 7.2 : (i) élaboration du cas d'étude, et (ii) application de CAHM au cas d'étude.

7.2.2.1 Étape 1 : Élaboration du cas d'étude « SU »

Dans cette étape, nous avons défini le périmètre du cas d'étude et dressé la liste des points de vue métier à prendre en considération lors de sa conception. Ensuite, nous avons élaboré le descriptif du cas d'étude (*définir Cahier des charges (CdC)*) en s'appuyant sur des échanges avec un médecin urgentiste d'un hôpital français et des travaux de la littérature décrivant la gestion des patients dans les services d'urgence [Daknou et al., 2008; Ajmi et al., 2015, 2018].
Une fois le descriptif du cas d'étude réalisé, nous avons défini les points de vue métier à prendre en compte (*définir points de vue métier*) et nous avons lancé un appel à des équipes de recherche pour proposer des modèles conformes aux méta-modèles des points de vue métier sélectionnés [El Hamlaoui et al., 2016]. Le choix des acteurs pour l'élaboration du cas d'étude a respecté les compétences des équipes pour que les modèles soient produits par des acteurs experts dans le domaine. Ainsi, on ne remet pas en cause la cohérence interne des modèles. Parmi les points de vue illustrés sur la Figure 7.1, nous avons choisi de considérer dans un premier temps les trois points de vue métier suivants :

- Point de vue Données (Software Design - SD),
- Point de vue Processus (Business Process - BP),
- Point de vue Présentation maquettes (Medical Examination Report - ER) ;

Le tableau 7.1 récapitule les équipes impliquées et les modèles qu'elles ont fournis. Ayant obtenu deux modèles $M1_1$ et $M1_2$ et $M2_1$ et $M2_2$ respectivement pour les points de vue BP et ER, nous avons procédé à une opération manuelle d'unification des deux modèles de chaque point de vue. Cette opération résulte en deux modèles M1 et M2 décrivant respectivement les points de vue BP et ER.

Tableau 7.1 – Équipes impliquées dans l'élaboration du cas d'étude SU

Équipe/Laboratoire, Pays	**Modèle produit**
IMS/ADMIR, Maroc	modèle SD
PASS/IRISA, France	modèle BP
RIADI-GDL, Tunisie	modèle ER
UMASS, USA	modèle BP
ARGOS/IRIT, France	modèle ER

7.2.2.2 Étape 2 : Application de CAHM au cas d'étude « SU »

Cette étape concerne le déroulement des deux phases de CAHM. Dans cette étape, nous avons sollicité des doctorants du laboratoire ADMIR qui n'avaient pas participé à l'élaboration du cas d'étude. Chaque doctorant a pris le rôle d'un concepteur d'un point de vue, et on lui a présenté le descriptif du cas d'étude, le modèle et le méta-modèle du point de vue concerné.
Les doctorants participants avaient des connaissances variées en IDM, allant de notions de base jusqu'à celles de concepteur confirmé. L'intérêt est de vérifier la faisabilité de l'approche malgré des connaissances préalables non équivalentes en IDM qui est le cadre de base de l'approche.
Dans la phase 1 de CAHM, nous avons impliqué trois doctorants pour mettre en correspondance les trois modèles des points de vue SD, BP et ER. Pour dérouler la phase 2 de CAHM, nous avons ajouté le point de vue organisationnel (MAS) et opéré des modifications sur les modèles des trois points de vue précédents suite aux feedbacks des doctorants ayant réalisé la première phase de CAHM.
Par la suite nous nommons les acteurs impliqués dans le déroulement de l'approche sur le cas d'étude comme suit : Alice, Bob, Claire, Dan et Jack pour faire référence respectivement aux concepteurs des modèles SD, BP, ER, et MAS, tandis que Jack est le modérateur des collaborations.

7.2.3 Description des points de vue du cas d'étude

7.2.3.1 Spécification du point de vue données (SD)

Méta-modèle de conception logicielle (SD)
Le méta-modèle de conception logicielle est une version allégée du méta-modèle UML. La Figure 7.3 donne un aperçu de ses concepts. Le concept de base est *package* qui peut se composer de *classes* et *interfaces*. Ces deux derniers possèdent des *attributs* et *méthodes*.

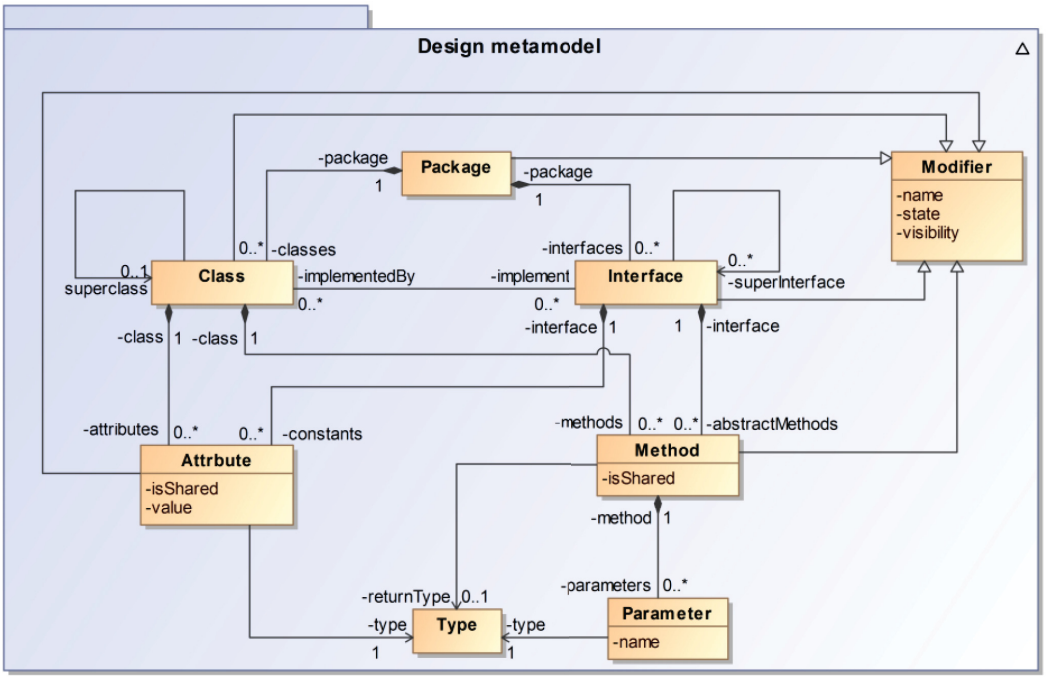


FIGURE 7.3 – Méta-modèle du point de vue conception logicielle.

Modèle de conception logicielle (SD)

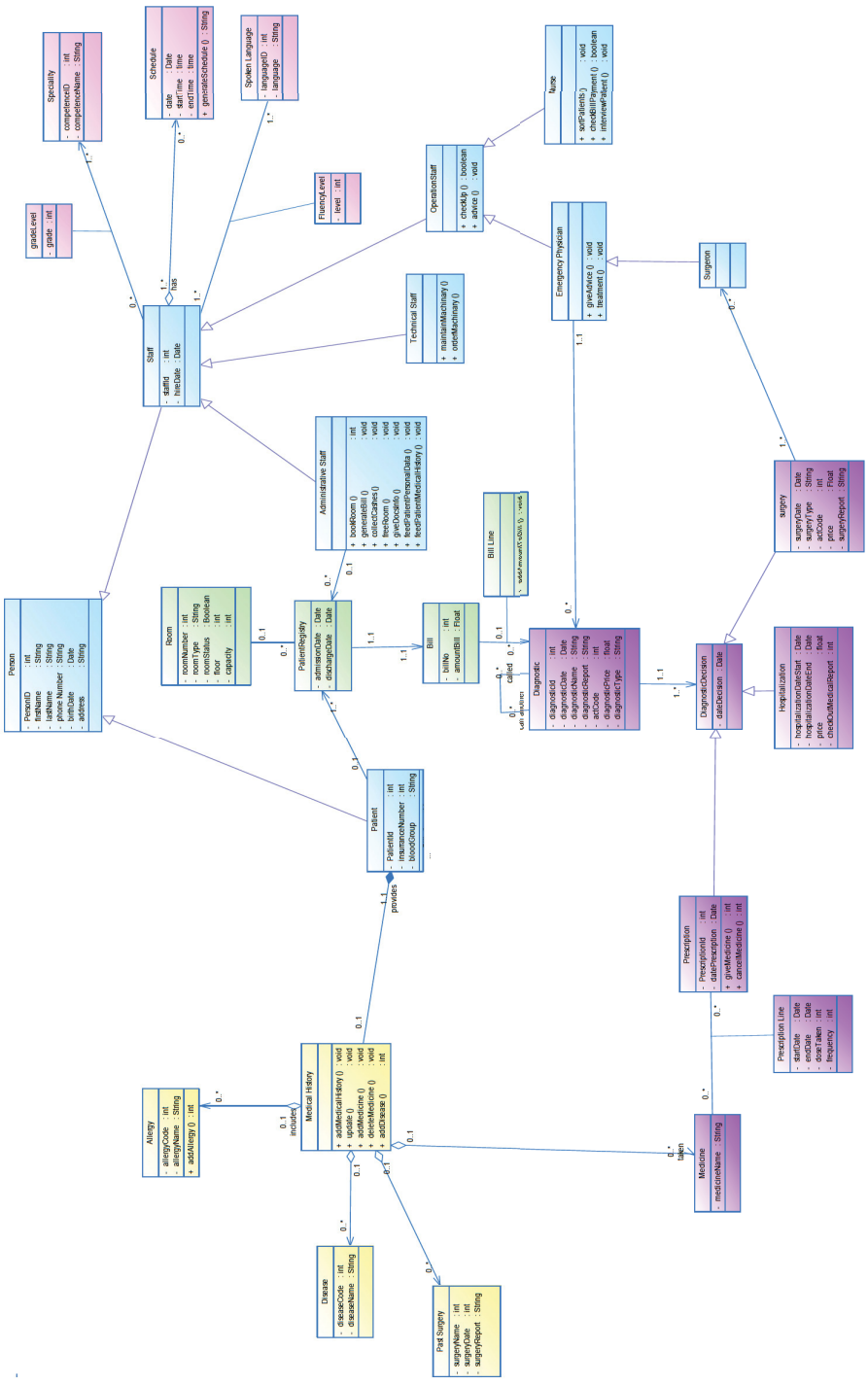


FIGURE 7.4 – Modèle du point de vue conception logicielle.

La Figure 7.4 présente le modèle de conception logicielle du service d'urgence, que nous appelons SD comme le méta-modèle correspondant. Un SU se caractérise par son personnel (personnel administratif, personnel technique, infirmière, urgentiste et chirurgien), ses patients et ses salles d'examen. Pour chaque membre du personnel, il est possible de connaître ses certifications et compétences, les langues parlées, sa date d'embauche et son emploi du temps. Le rôle d'un personnel technique est de maintenir et de commander des machines. Le rôle du personnel administratif consiste à réserver des chambres, donner des informations sur un médecin, etc. Le médecin urgentiste donne des conseils et traite ses patients. Un diagnostic a un nom, une

description et un code d'acte. Un chirurgien effectue des interventions chirurgicales, chacune étant caractérisée par un patient, une date, un type et un code d'acte. Le modèle contient également des informations sur chaque patient telles que son numéro d'assurance (le cas échéant), ses allergies, ses antécédents médicaux, ses diagnostics et ses ordonnances, etc.

7.2.3.2 Spécification du point de vue processus (BP)

Méta-modèle de processus métier (BP)
Le méta-modèle représenté sur la Figure 7.5 est celui utilisé pour construire le modèle des processus métier. C'est le même que celui utilisé dans le chapitre 5 pour décrire le point de vus processus du CMS ; il est inspiré de BPMN [1] et comprend entre autres les concepts pool, *lane*, et *task*. Pool décrit la représentation graphique d'un participant au processus. *Lane* est une sous-partition dans un processus (souvent dans un *pool*). Une tâche (*task*) est une activité atomique dans un flux de processus.

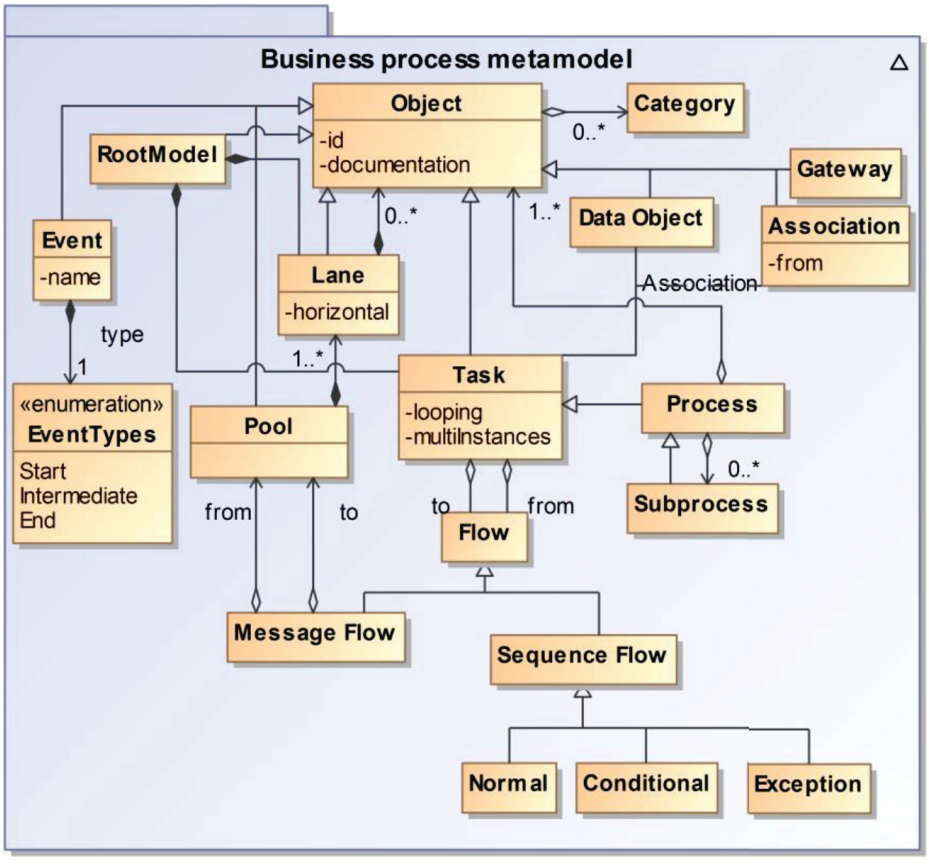


FIGURE 7.5 – Méta-modèle du point de vue processus métier.

1. https://www.omg.org/spec/BPMN/2.0/PDF

Modèle de processus métier (BP)
La Figure 7.6 présente le modèle BP du SU. Il décrit les différents protocoles à appliquer par chaque type de personnel. Tout d'abord, un assistant médical (*Administrative Agent*) reçoit chaque patient, l'enregistre sur la base de données du SU et le redirige vers une salle d'attente. Ensuite, une des infirmières (*Nurse*) disponibles est notifiée de l'arrivée du patient, elle lui pose des questions sur ses antécédents et allergies pour alimenter son dossier médical, puis elle fait le tri des patients selon la gravité de leur état. Un médecin urgentiste (*Emergency Physician*) réalise un examen initial du patient suite auquel il peut procéder à une réanimation, un transfert dans un autre hôpital (si le cas ne peut être traité en interne) ou bien effectuer une consultation approfondie. A la suite de la consultation, il peut définir un traitement que doit suivre le patient ou procéder à l'hospitalisation de ce dernier. L'infirmière revient vers le patient et exécute les prescriptions du médecin qui peuvent comprendre des examens biologiques et/ou radiologiques, etc. Par la suite, le médecin urgentiste revoit le patient et, selon les résultats des prescriptions, peut définir un nouveau traitement ou bien produire un rapport d'examen d'urgence (EER) pour procéder à la sortie du patient.

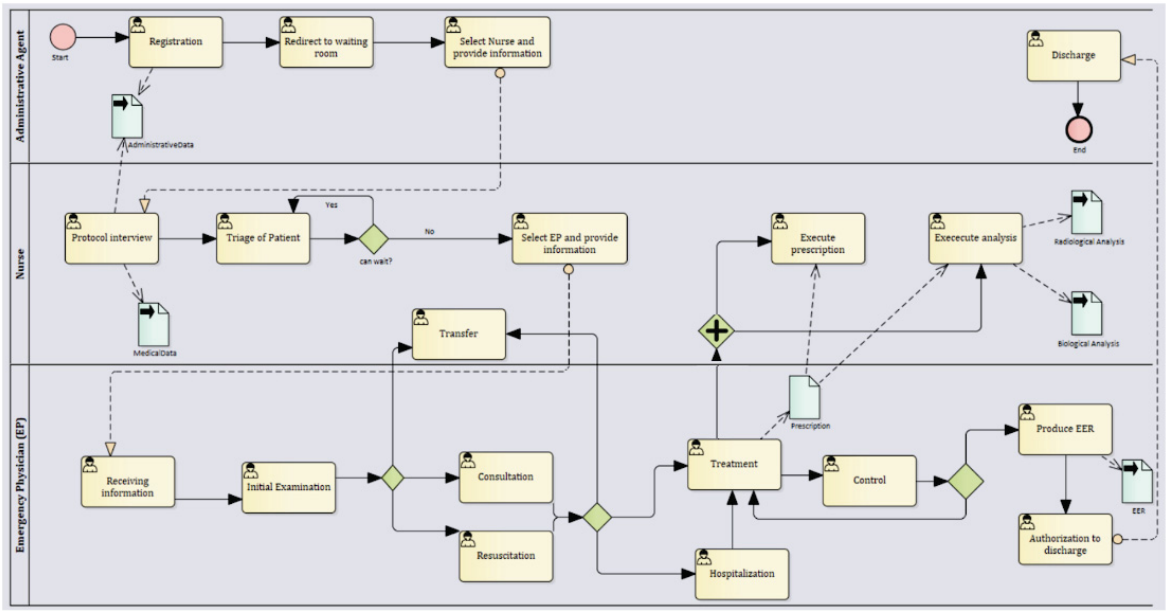


FIGURE 7.6 – Modèle du point de vue processus métier.

7.2.3.3 Spécification du point de vue fonctionnel (ER)

Méta-modèle des rapports médicaux (ER)
Le méta-modèle proposé sur la Figure 7.7 représente les maquettes numériques d'un rapport d'examen du système SU. Son concept clé est un *formulaire* comportant un ensemble de *champs* qui peuvent être éventuellement composites.

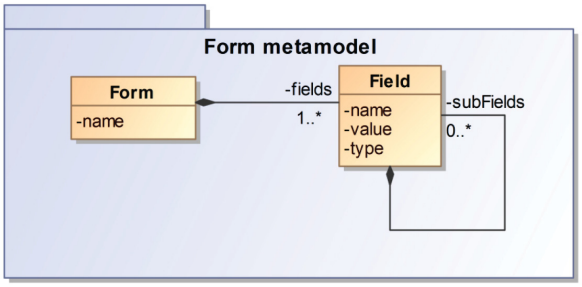


FIGURE 7.7 – Méta-modèle du point de vue des rapports médicaux.

Modèle des rapports médicaux (ER)

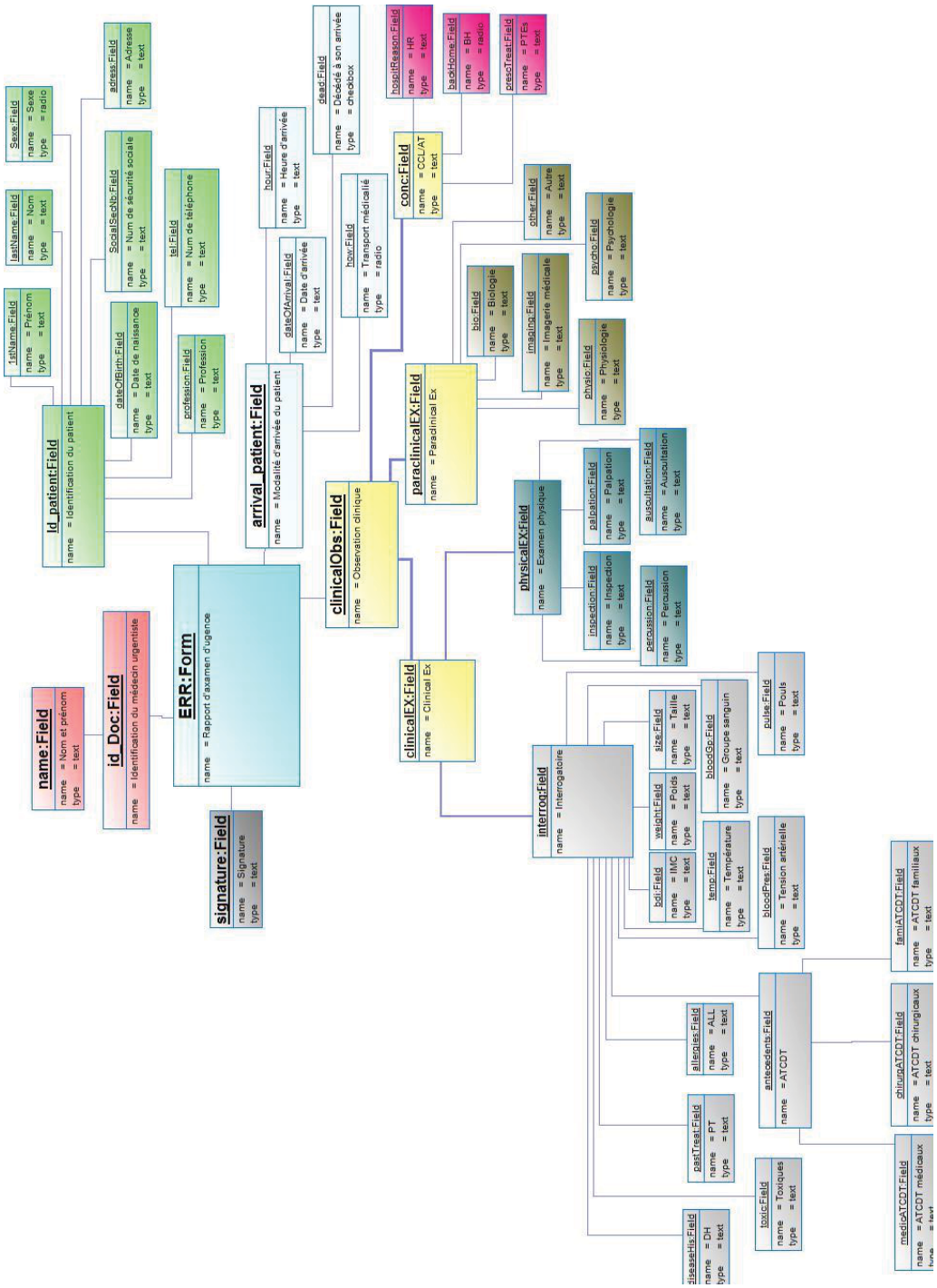


FIGURE 7.8 – Modèle du point de vue rapports médicaux.

La Figure 7.8 représente le modèle ER du service d'urgence. Il contient les informations qui sont produites par un médecin urgentiste pour un patient donné. De manière générale, il contient l'identifiant du médecin, des informations sur le patient (numéro de sécurité sociale, sexe, prénom, nom et date de naissance), la date d'arrivée du patient et une observation clinique. Cette dernière est réalisée à l'aide d'abréviations codifiées. Il identifie la pathologie afin d'orienter le patient vers le service approprié au sein du même établissement ou d'un autre hôpital. Afin de pouvoir appliquer la sémantique des relations aux éléments de ce modèle, nous utilisons le nom complet de l'élément au lieu de l'abréviation (e.g., nous utilisons *Field :physiologicalExamination* au lieu de *Field :physio*).

7.2.3.4 Spécification du point de vue organisationnel (MAS)

Ce point de vue peut être vu comme la rencontre de divers domaines : l'intelligence artificielle pour les aspects prise de décision de l'agent (au sens système multi-agents), les systèmes distribués pour les interactions, et le génie logiciel pour l'approche agents et processus [ŠPERKA, 2011]. Son utilisation d'agents autonomes communicants et orientés objectif offre un degré élevé de simultanéité et de décentralisation des tâches, des informations et des ressources impliquées dans un processus contrairement à ce qui est fait dans le point de vue processus (section 7.2.3.2).

Méta-modèle de système multi-agents (MAS)
Le méta-modèle proposé sur la Figure 7.9 représente les caractéristiques d'un système multi-agents qui est composé d'un ensemble d'éléments situés dans un certain environnement. Ces éléments sont des agents (e.g., un processus, un robot, un être humain), pouvant interagir selon certaines relations. Ce méta-modèle est une version simplifiée du méta-modèle proposé dans ADELFE [2] [BONJEAN et al., 2014]. Un agent possède des représentations qui sont des perceptions concernant d'autres éléments de l'environnement. Ces représentations sont utilisées par l'agent pour déterminer son comportement. Un agent possède des compétences et peut avoir des aptitudes et des caractéristiques. Une compétence est une connaissance spécifique qui permet à un agent de réaliser sa propre fonction. Une aptitude montre la capacité d'un agent à raisonner à la fois sur ses connaissances. Par exemple, une aptitude d'un agent logiciel peut être exprimée par un moteur d'inférence basé sur des règles (*EngineRule*). Les caractéristiques sont des propriétés intrinsèques ou physiques.

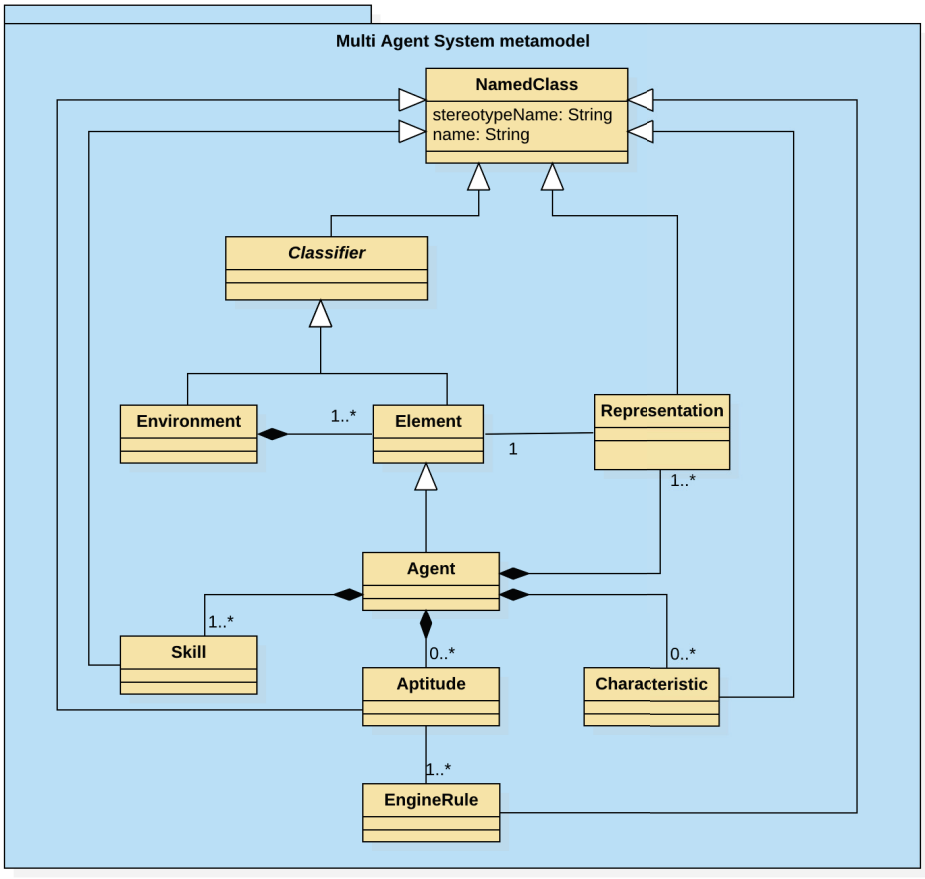


FIGURE 7.9 – Méta-modèle du point de vue système multi-agents.

2. ADELFE est un acronyme pour Atelier de Développement de Logiciels à Fonctionnalité Émergente.

Modèle de système multi-agents (MAS)

La Figure 7.10 représente le modèle MAS du service d'urgence qui a pour objectif d'optimiser le flux des patients, d'ordonnancer l'activité du processus de soin et de construire des plannings pour le personnel médical. L'instanciation de ce modèle garantit à terme une prise en charge rapide et de qualité, tout en planifiant de façon semi-automatique les ressources du système hospitalier. Le modèle est basé sur le travail de thèse de [AJMI, 2015] qui décrit trois agents : Agent Accueil (*ReceptionAgent*), Agent Identificateur (*IdentifyingAgent*), et Agent Ordonnanceur (*SchedulingAgent*). Les rôles de chaque Agent sont résumés comme suit :

— **Agent Accueil** : son rôle est d'annoncer l'arrivée d'un patient aux autres agents. Il enregistre le patient dans le service grâce aux informations données par les assistants médicaux.

— **Agent Identificateur** : son rôle est d'identifier les ressources nécessaires pour les diagnostics des patients. Il reçoit le dossier médical complet du patient, contenant un certain nombre de diagnostics établis par le médecin. Il génère le plan de traitement et des soins du patient, en créant une liste des tâches à effectuer. Il détermine également l'équipe médicale nécessaire à chaque tâche de soin.

— **Agent Ordonnanceur** : son rôle est d'optimiser le temps d'attente dans le service. Cet Agent joue le rôle de planificateur des tâches pour tous les patients présents dans le service.

L'action de l'ordonnanceur dépend de l'activité dans les urgences. En effet, si le service des urgences n'est pas surchargé, et que le personnel médical n'est pas totalement occupé, l'ordonnanceur s'occupe d'attribuer la tâche de soin au personnel qui, selon le protocole médical, peut se charger de la réalisation de cette tâche. En cas de suractivité, l'ordonnanceur fait appel à l'algorithme génétique qui va revoir les plannings de tous les personnels, en respectant les contraintes imposées.

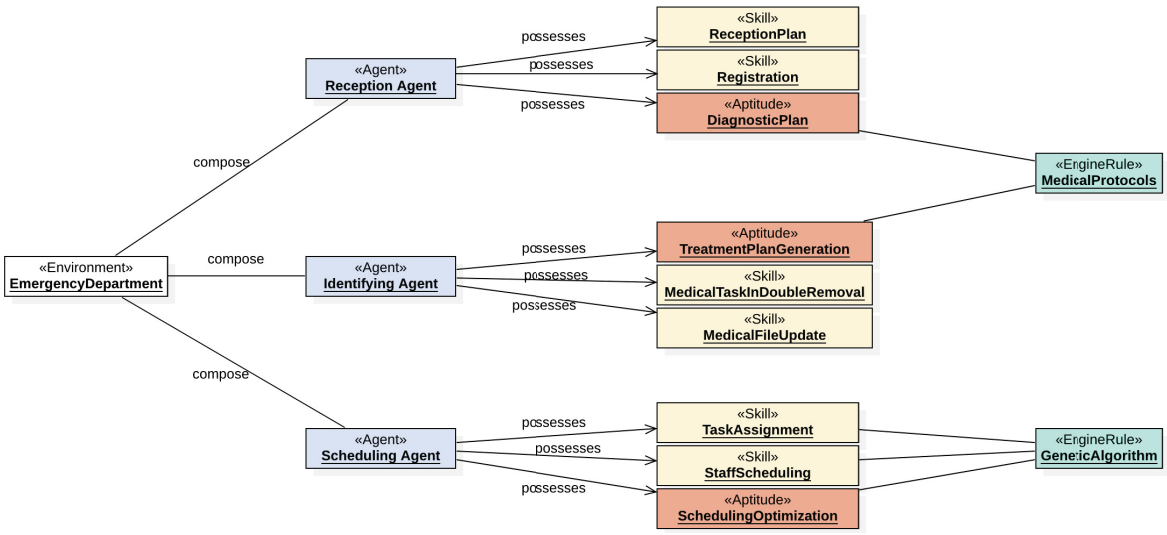


FIGURE 7.10 – Modèle du point de vue système multi-agents.

7.3 Application de CAHM - phase 1 : Mise en correspondance collaborative des modèles du système SU

Dans cette phase, nous considérons les trois points de vue métier SD, BP et ER. HMCS-Collab est utilisé pour réaliser les différentes activités automatisables. La section 7.3.1 illustre la problématique de mise en correspondance collaborative sur le système SU. La section 7.3.2 présente la configuration des acteurs et points de vue métier du système SU sur l'outil. La section 7.3.3 présente l'activité 3 du processus de mise en correspondance collaborative (cf. section 5.4.3) en détaillant la configuration de cette activité collaborative, de sa politique de prise de décision, la proposition des méta-correspondances et leurs évaluations. La section 7.3.4 présente l'activité 4 du processus de mise en correspondance collaborative qui correspond à la génération du modèle de correspondances. Les activités 1 et 2 (Verifier l'adéquation du MMC et Étendre MMC) ne sont pas présentées puisqu'on considère que le MMC existant est suffisant avec les six relations vues dans le chapitre 5.

7.3.1 Illustration de la problématique de mise en correspondances collaborative

En analysant les modèles source présentés dans la section précédente, on constate que certains éléments de modèles portent les mêmes noms dans plus d'un modèle (par exemple les éléments « a1 » et « a2 » sur la Figure 7.11). D'autres éléments ont des noms dépendants (les éléments « b1 » et « b2 »). Ainsi, le premier objectif est de déterminer la nature des liens qui unissent ces éléments. Il faut aussi pouvoir mettre en évidence des liens entre des éléments qui ne sont pas synonymes ou dépendants, mais qui peuvent avoir entre eux une autre relation sémantique, cachée du fait de la diversité des dictionnaires de données utilisés pour exprimer les modèles.

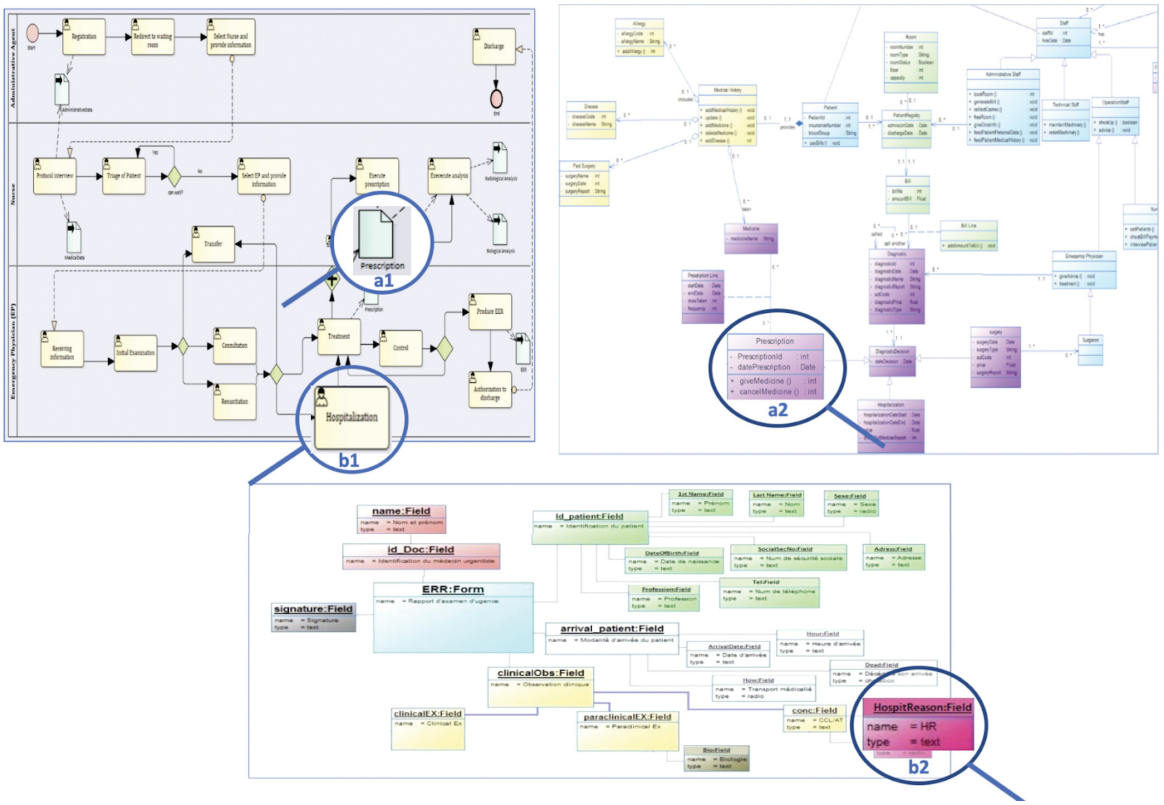
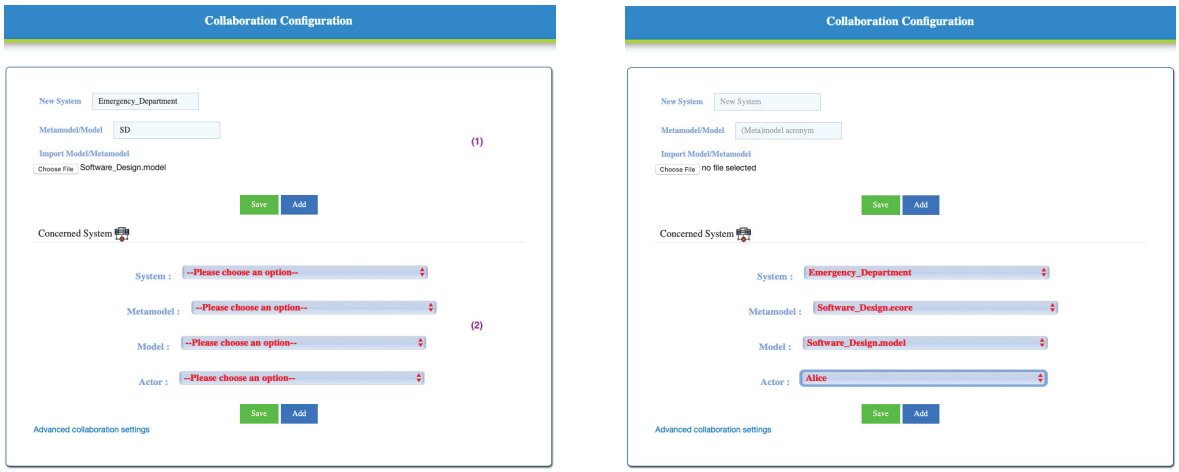


FIGURE 7.11 – Exemple illustrant la problématique de mise en correspondances sur le cas d'étude « service d'urgence ».

7.3.2 Configuration du système SU sur l'outil HMCS-Collab

Lors de la configuration du système, le modérateur, Jack, définit le système en question, et importe ses méta-modèles et modèles (partie 1 de la Figure 7.12a). Ensuite, Il précise pour chaque méta-modèle du système, le modèle et l'acteur qui le représente (partie 2 de la Figure 7.12a). Sur la Figure 7.12a, Jack renseigne respectivement les valeurs *Emergency_Department* et *SD* dans les champs *New System* et *Metamodel/Model*; il importe aussi le fichier « *Software_Design.model* » via le bouton *Choose File*. A chaque fois qu'il importe un (méta-)modèle, il utilise le premier bouton *add* de la Figure 7.12a. Sur la Figure 7.12b, Alice est indiquée comme acteur du point de vue conception logicielle (*software design*) du SU.



(a) (b)

FIGURE 7.12 – Interface de configuration du système SU dans HMCS-Collab.

7.3.3 CAHM - phase 1 - Activité 3 : Définir les méta-correspondances

Configuration de la politique de décision

Lors de la configuration de la politique de décision, Jack choisit l'une des sept politiques prédéfinies (cf. section 4.4.2). Sur la Figure 7.13, il opte pour la politique *Delegated Negotiating*.
Une fois son choix fait (i.e., le champ *Chosen policy* renseigné), les paramètres avancés (*Advanced Settings*) sont exploitables, les paramètres figés de la politique de décision sont listés. Il s'agit des paramètres *Process Kind* fixé à *Negotiation* et *Participation Type* fixé à *restricted* sur l'illustration, étant donné que la politique choisie est *Delegated Negotiating*.
Le reste des paramètres peut être précisé par Jack sur cette même interface. Jack coche aussi le champ correspondant à la collaboration pour laquelle il souhaite appliquer cette politique (activité « Define meta-correspondences » pour la Figure 7.13).

Propositions des méta-correspondances

La Figure 7.14 présente l'interface de proposition de méta-correspondance, l'utilisateur renseigne le type de relation que la méta-correspondance va exploiter (a), et choisit ensuite dans les listes déroulantes les méta-modèles (b) et leurs méta-éléments (c) impliqués en précisant pour chaque méta-élément s'il est considéré comme concept source ou cible de la correspondance. Sur l'exemple, Bob choisit (a) la relation d'induction, il renseigne les méta-modèles *Business_Process.ecore* et *Software_Design.ecore* comme méta-modèles source et leurs méta-éléments respectifs *Task* et *Method* comme méta-élements source (d). Ensuite, il renseigne les valeurs *Emergency_Report.ecore* et *Field* comme méta-modèle et méta-élément cible (*target*) (e) de la méta-correspondance.
La Figure 7.15 liste les propositions de méta-correspondances pour le système. Pour chaque proposition sont renseignés le corps de la proposition, son initiateur, ses décideurs et la décision collective le cas échéant.

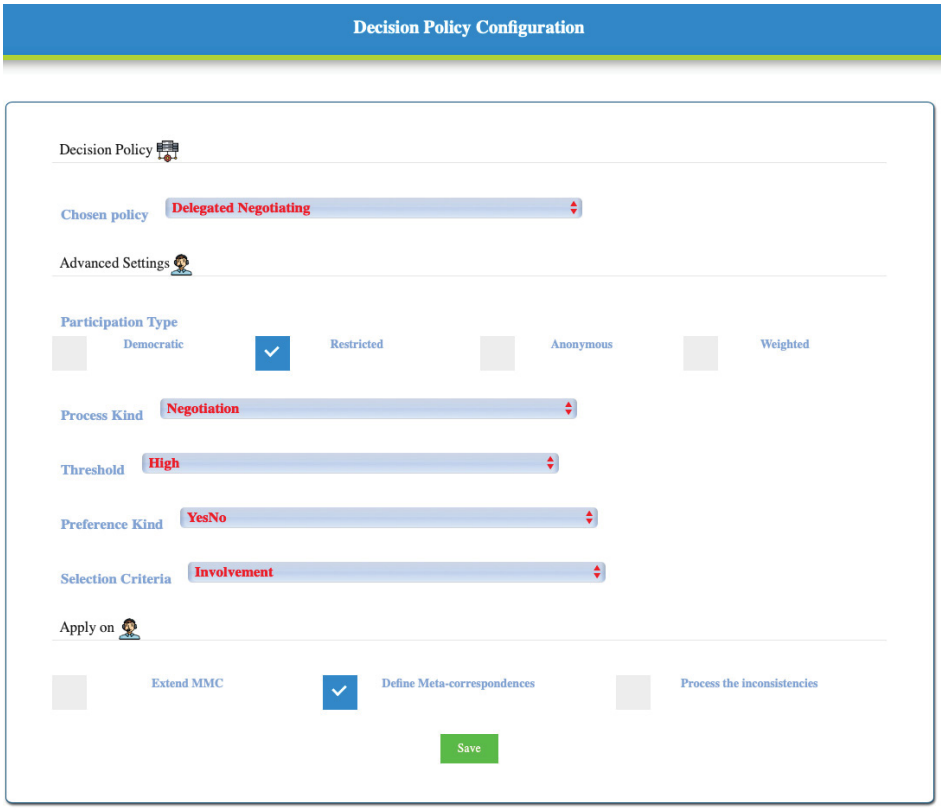


FIGURE 7.13 – Interface de configuration de la politique de décision de l'activité collaborative « Définir les méta-correspondances » dans HMCS-Collab.

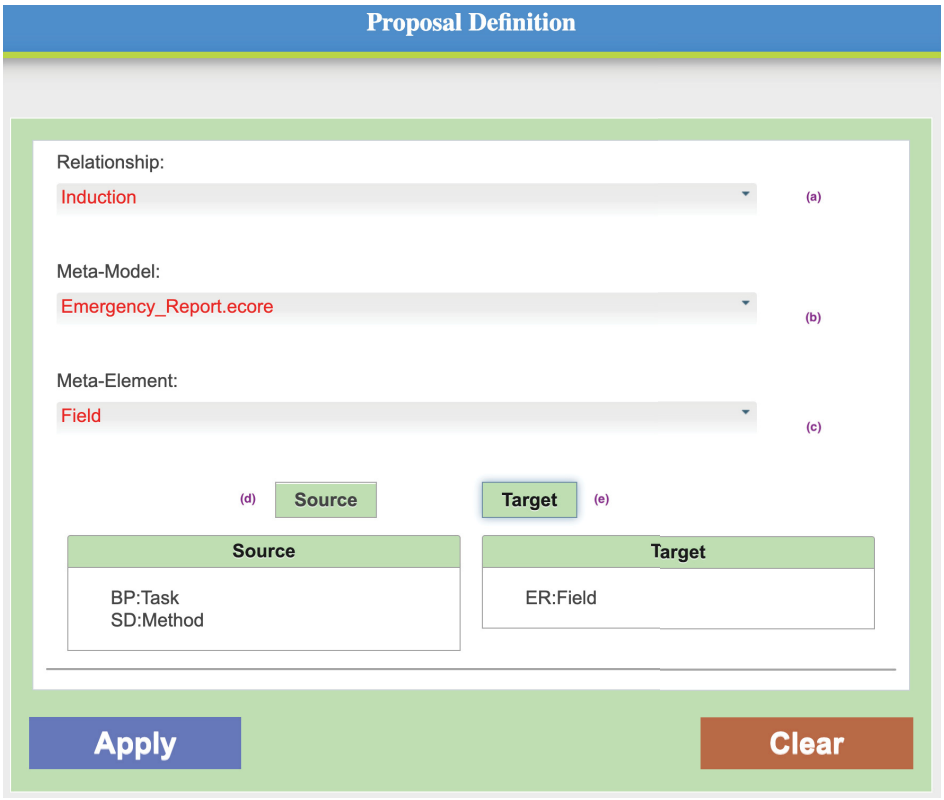


FIGURE 7.14 – Interface de proposition de méta-correspondance dans HMCS-Collab.

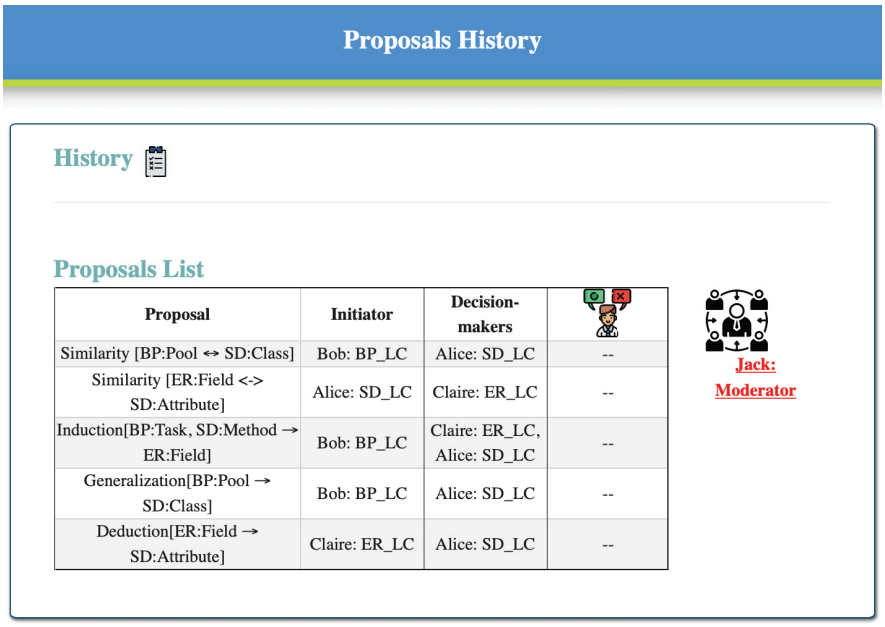


FIGURE 7.15 – Interface de la liste des méta-correspondances proposées.

Évaluations des méta-correspondances proposées

FIGURE 7.16 – Interface d'évaluation des méta-correspondances par Alice.

Une fois les propositions élaborées, chaque décideur est notifié. Il peut renseigner son évaluation individuelle grâce à l'interface illustrée sur la Figure 7.16. Si la politique de décision est une politique itérative, il pourra proposer d'autres propositions durant la collaboration. Sinon, il évalue celles qui sont déjà établies et doit fournir obligatoirement un commentaire en cas de rejet de la proposition. Sur la Figure 7.16, Alice a approuvé la première et la dernière méta-correspondance et a rejeté la troisième ; un champ s'affiche pour qu'Alice puisse renseigner le motif de son rejet. Alice a choisi de raffiner la deuxième méta-correspondance. Quand elle clique sur le bouton *submit*, elle est redirigée vers une autre interface (Figure 7.17) où elle renseigne la proposition alternative. Pour cela, elle précise quelle proposition fait l'objet du raffinage ; elle coche la case *not conflictual* pour indiquer que les deux propositions ne sont pas en conflit et remplit le corps de la proposition alternative.

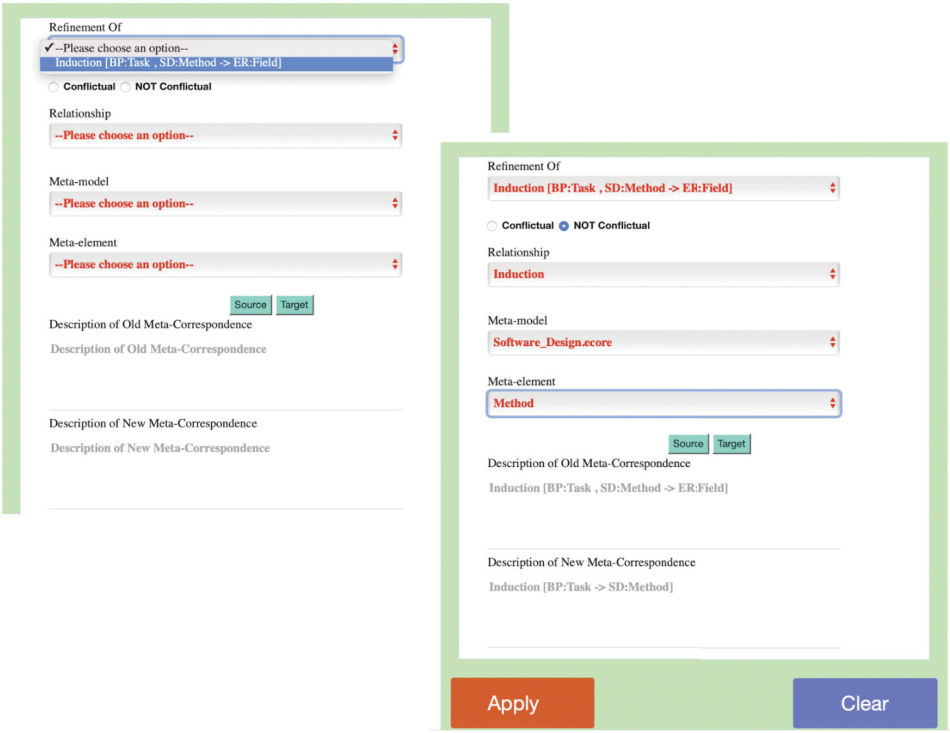


FIGURE 7.17 – Interface de la proposition alternative initiée par Alice.

7.3.4 CAHM - phase 1 - Activité 4 : Générer le modèle de correspondances

L'interface suivante (Figure 7.18) permet de voir les décisions collectives prises par proposition.

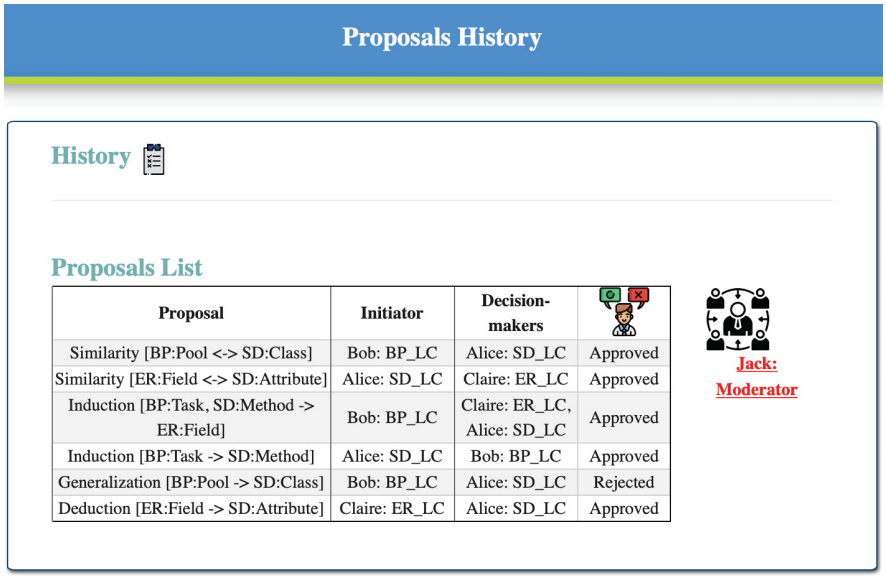
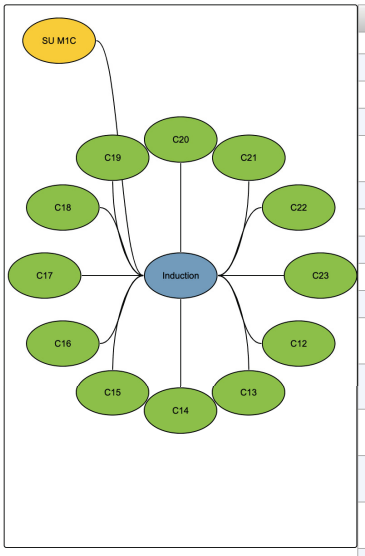


FIGURE 7.18 – Interface des décisions collectives des méta-correspondances proposées dans HMCS-Collab.

Une fois que la décision collective est établie par proposition, ou bien si la durée fixée pour la collaboration arrive à échéance, l'outil génère le modèle de correspondances du système et notifie les acteurs pour qu'ils puissent le visualiser. La Figure 7.19 présente un extrait du modèle de correspondances du service d'urgence. La visualisation se fait de deux façons en exploitant une *mindmap* interactive (côté gauche de la Figure) et en parcourant un tableur (côté droit de la Figure).

Model Of Correspondences

ED System



Id : Relation	Sources Elements	Target Elements
C9 : Similarity	[SD:Class:EmergencyPhysician]	[BP:Pool:EmergencyPhysician]
C10 : Similarity	[SD:Class:Nurse]	[BP:Pool:Nurse]
C11 : Similarity	[SD:Class:AdministrativeStaff]	[BP:Pool:AdministrativeAgent]
C12 : Induction	[BP:Task:Control]	[SD:Method:treatment]
C13 : Induction	[BP:Task:Control, SD:Method:treatment]	[ER:Field:PrescriptionTreatment]
C14 : Induction	[BP:Task:Treatment]	[SD:Method:treatment]
C15 : Induction	[BP:Task:Consultation]	[SD:Method:giveAdvice]
C16 : Induction	[BP:Task:Consultation]	[SD:Method:advice]
C17 : Induction	[BP:Task:TriageOfPatient]	[SD:Method:sortPatients]
C18 : Induction	[BP:Task:AuthorizationToDischarge]	[SD:Method:checkBillPayment]
C19 : Induction	[BP:Task:ProtocolInterview, SD:Method:interviewPatient]	[ER:Field:Interrogation]
C20 : Induction	[BP:Task:ExecuteAnalysis, SD:Method:checkUp]	[ER:Field:ClinicalExamination]
C21 : Induction	[BP:Task:ExecuteAnalysis, SD:Method:checkUp]	[ER:Field:ParaclinicalExamination]
C22 : Induction	[BP:Task:ExecuteAnalysis, SD:Method:checkUp]	[ER:Field:physicalExamination]
C23 : Induction	[BP:Task:ExecuteAnalysis, SD:Method:checkUp]	[ER:Field:physiologicalExamination]
C24 : Deduction	[ER:Field:HospitalReason]	[SD:Attribute:priceHospitalization]

FIGURE 7.19 – Extrait du modèle de correspondances M1C du service d'urgence visualisé sur HMCS-Collab.

7.4 Application de CAHM - phase 2 : Maintien de la cohérence lors des évolutions

Dans cette phase, nous ajoutons le point de vue Multi Agent System (MAS) et apportons aussi quelques changements aux modèles des trois points de vue utilisés dans la phase 1 (section 7.3). La section 7.4.1 illustre les changements soulevés sur le système SU. La section 7.4.2 présente le calcul de l'ordre de traitement des changements selon leurs impacts, tandis que la section 7.4.3 montre le traitement des impacts des changements soulevés et des incohérences introduites.

7.4.1 CAHM - phase 2 - Activité 1 : Détecter les changements

Nous supposons que le système SU a subi des changements pour répondre aux feedbacks des acteurs ayant mis en oeuvre la phase 1. Ainsi, le *Field physio* (*physiologicalExamination*) du modèle ER a été spécialisé en ajoutant les *subfields hormonalEx*, *immunologicalEx*, *serologicalEx* et *functionalEx* comme résumé sur la Figure 7.20. Ces quatre *Field* correspondent à des spécialisations d'examens physiologiques.
Le modèle BP subit aussi un changement ; la *Task Control* est renommée en : *MedicalMonitoring*.
En plus de ces changements partiels, Jack, le modérateur, ajoute le point de vue MAS pour compléter la représentation du système. Comme précisé précédemment, ce point de vue offre un degré élevé de simultanéité et de décentralisation des tâches et des ressources impliquées dans un processus contrairement à ce qui est fait dans le point de vue processus. Jack importe le méta-modèle et le modèle du point de vue MAS et choisit Dan comme acteur responsable de ce point de vue ; ceci en utilisant la même interface que celle présentée sur la Figure 7.12.
L'ajout d'un point de vue étant un changement global, il doit être traité sans attente. Nous passons donc à l'activité 2 du processus de CAHM-phase 2 (cf section 5.5.5). Notons que tous ces changements, subis par le système SU, sont impactants.

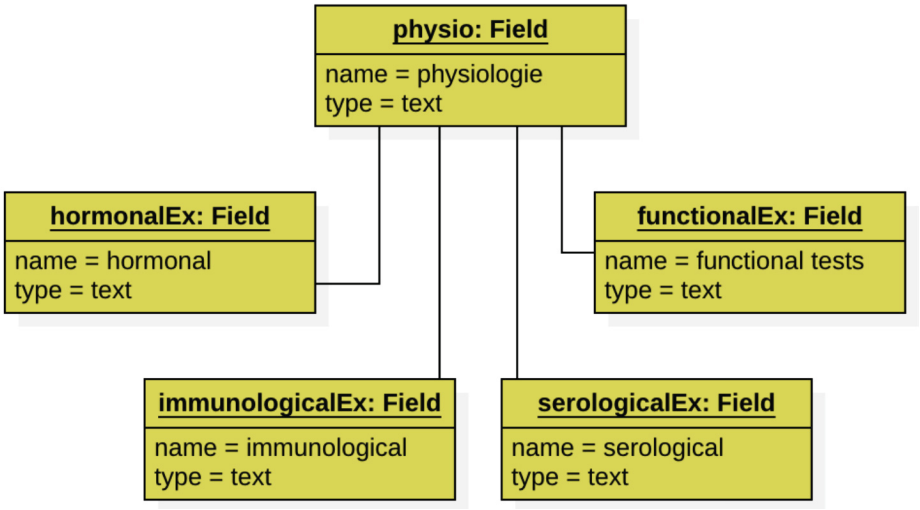
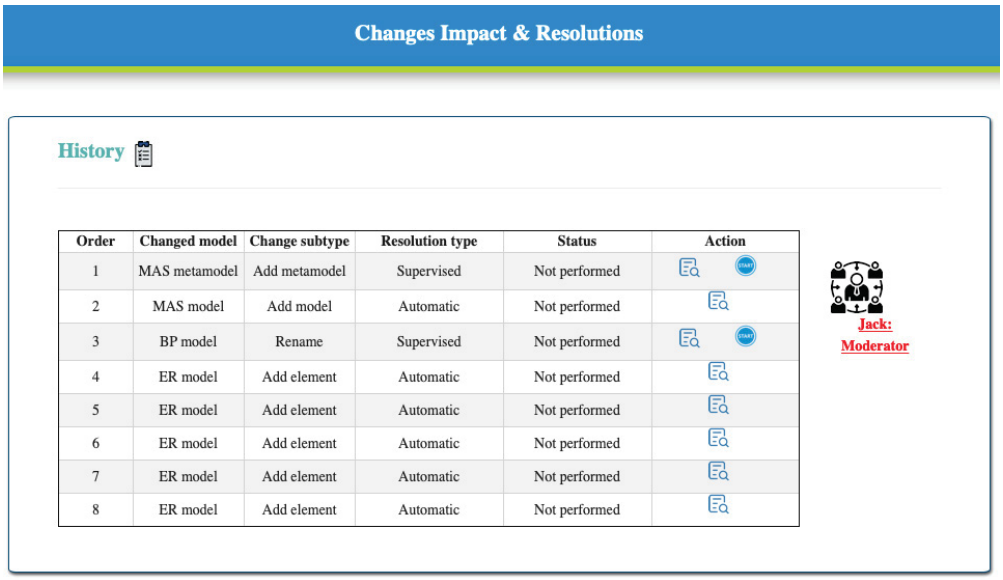


FIGURE 7.20 – Spécialisation du *Field :physio* lors de l'évolution du modèle ER.

7.4.2 CAHM - phase 2 - Activité 2 : Calculer l'impact et l'ordre de traitement des changements

L'outil HMCS-Collab classifie les changements selon leur impact et génère leurs graphes de dépendances selon l'algorithme décrit dans l'annexe B.1. L'interface suivante (Figure 7.21) permet à Jack de visualiser l'ordre de traitement des changements et de commencer le traitement d'un changement supervisé grâce au bouton *start*. Jack peut aussi, à partir de cette interface, cliquer sur le bouton *détails* pour voir les informations détaillées concernant un changement y compris son graphe de dépendances.



Order	Changed model	Change subtype	Resolution type	Status	Action
1	MAS metamodel	Add metamodel	Supervised	Not performed	
2	MAS model	Add model	Automatic	Not performed	
3	BP model	Rename	Supervised	Not performed	
4	ER model	Add element	Automatic	Not performed	
5	ER model	Add element	Automatic	Not performed	
6	ER model	Add element	Automatic	Not performed	
7	ER model	Add element	Automatic	Not performed	
8	ER model	Add element	Automatic	Not performed	

FIGURE 7.21 – Interface de classement des impacts des changements détectés.

Les changements classés premier et deuxième sont des changements globaux de type ajout de méta-modèle ou modèle ; les graphes de dépendances sont inutiles étant donné qu'il faut relancer le processus de mise en correspondance pour les traiter.

Le graphe de dépendances du troisième changement (*rename Task :Control*) est constitué des éléments reliés à l'élément changé (*Task :Control*) (cf. annexe B.2.1). La Figure 7.22 illustre ce graphe. La racine du graphe est l'élément *Task :Control* du modèle BP, le graphe comporte deux autres éléments qui sont reliés à *Task :Control* dans le M1C ; ces éléments sont : *Method :treatment* et

Field :PrescriptionTreatment[3]. Sur chaque noeud, l'interface précise le nom du noeud, sa distance à la racine et les correspondances qui justifient cette distance de la racine. A partir de la liste déroulante, Jack peut visualiser les détails des correspondances impliquées dans ce graphe ; sur l'illustration, il choisit de voir les détails de la correspondance C12.
Les changements classés en positions 4, 5, 6, 7 et 8 n'ont pas de graphe de dépendances puisqu'ils sont de type ajout d'élément de modèle.

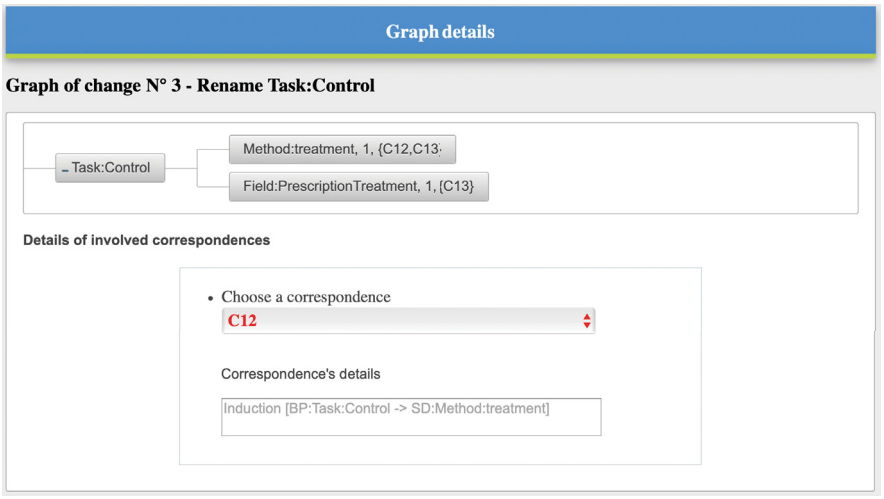
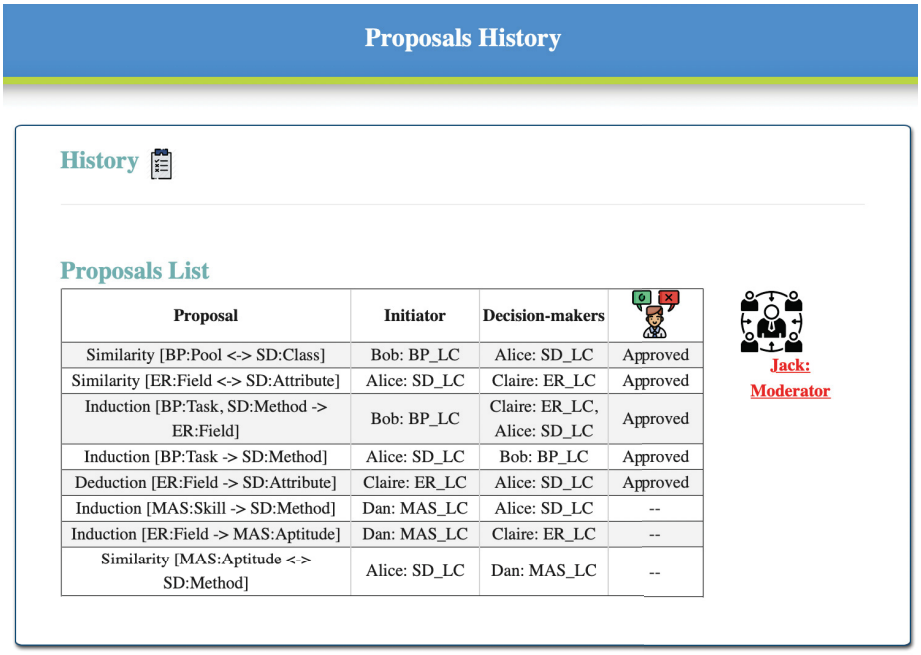


FIGURE 7.22 – Interface de visualisation du graphe de dépendances de *Task :Control*.

7.4.3 CAHM - phase 2 - Activité 3 : Traiter les incohérences

Dans cette activité, nous détaillons le traitement des changements recensés sur la Figure 7.21.

7.4.3.1 Changement N° 1 : Ajout du méta-modèle MAS



Proposal	Initiator	Decision-makers	
Similarity [BP:Pool <-> SD:Class]	Bob: BP_LC	Alice: SD_LC	Approved
Similarity [ER:Field <-> SD:Attribute]	Alice: SD_LC	Claire: ER_LC	Approved
Induction [BP:Task, SD:Method -> ER:Field]	Bob: BP_LC	Claire: ER_LC, Alice: SD_LC	Approved
Induction [BP:Task -> SD:Method]	Alice: SD_LC	Bob: BP_LC	Approved
Deduction [ER:Field -> SD:Attribute]	Claire: ER_LC	Alice: SD_LC	Approved
Induction [MAS:Skill -> SD:Method]	Dan: MAS_LC	Alice: SD_LC	--
Induction [ER:Field -> MAS:Aptitude]	Dan: MAS_LC	Claire: ER_LC	--
Similarity [MAS:Aptitude <-> SD:Method]	Alice: SD_LC	Dan: MAS_LC	--

FIGURE 7.23 – Liste des méta-correspondances après l'ajout du point de vue MAS, et avant leur évaluation.

3. Sur la Figure 7.8, le nom de cet élément est *prescTreat*, mais, pour pouvoir lui appliquer la sémantique des relations, son nom est *prescriptionTreatment* sur l'outil HMCS-Collab.

L'ajout du méta-modèle MAS implique de proposer de nouvelles méta-correspondances et d'évaluer celles déjà définies au cas où elles seraient en conflit avec les nouvelles. La Figure 7.23 illustre les huit méta-correspondances considérées pour le système SU ; les cinq premières méta-correspondances sont celles approuvées dans la phase 1, elles ont été conservées puisque les méta-correspondances nouvellement proposées (le reste de la liste) n'ont pas été déclarées en conflit avec elles. La collaboration concernant les méta-correspondances nouvellement proposées adopte la politique de décision **Consenting Together** et les trois propositions sont approuvées suite à l'évaluation.

7.4.3.2 Changement N° 2 : Ajout du modèle MAS

L'ajout du modèle MAS implique la propagation des méta-correspondances qui viennent d'être ajoutées et la suppression des correspondances issues des méta-correspondances non retenues (cf. annexe B.3). Pour le système SU, aucune méta-correspondance de la phase 1 n'a été écartée. Ainsi, l'ajout du modèle MAS revient à propager uniquement les trois méta-correspondances nouvellement approuvées ; cette action est faite automatiquement par l'outil HMCS-Collab.
La propagation des trois méta-correspondances résulte en neuf correspondances, à savoir :

- Induction [MAS :Skill :ReceptionPlan − > SD :Method :feedPatientPersonalData]
- Induction [MAS :Skill :MedicalFileUpdate − > SD :Method :addMedicalHistory]
- Induction [MAS :Skill :MedicalFileUpdate − > SD :Method :addDisease]
- Induction [MAS :Skill :MedicalFileUpdate − > SD :Method :feedPatientMedicalHistory]
- Induction [MAS :Skill :MedicalFileUpdate − > SD :Method :addAllergy]
- Induction [MAS :Skill :TaskAssignment − > SD :Method :generateSchedule]
- Induction [MAS :Skill :StaffScheduling − > SD :Method :generateSchedule]
- Induction [ER :Field :HospitalReason − > MAS :Aptitude :DiagnosticPlan]
- Induction [ER :Field :HospitalReason − > MAS :Aptitude :TreatmentPlanGeneration]

7.4.3.3 Changement N° 3 : Renommage de la *Task :Control* du modèle BP

Le traitement de ce changement est lancé une fois les traitements des deux qui le précèdent achevés. Le renommage de *Task :Control* nécessite une collaboration pour décider du traitement des incohérences qu'il a engendrées (incohérence des correspondances C12 et C13 de la Figure 7.19). Pour rappel C12 : Induction [BP :Task :Control -> SD :Method :treatment] et C13 : Induction [BP :Task :Control, SD :Method :treatment -> ER :Field :PrescriptionTreatment]).
Le catalogue de résolutions (cf. annexe B.3) envisage deux résolutions pour ce type de changement, à savoir R2 et R6, avec *R2* : *remove correspondences(e)* et *R6* : *maintain correspondences(e)*. Il faut donc démarrer une *Collaboration* pour choisir la résolution adéquate.
Le graphe de dépendances de ce changement contient des éléments issus des modèles SD et Emergency Report (ER) en plus du modèle BP ; de ce fait, la *Collaboration* implique les coordinateurs locaux de ces trois modèles : Bob, Alice et Claire. La Figure 7.24 illustre le choix d'Alice concernant la résolution à appliquer pour ce changement. Elle choisit de maintenir la correspondance C12 et explique son choix aux autres concepteurs. De ce fait C12 et C13 sont maintenues avec comme nouvel élément source *Task :MedicalMonitoring*.

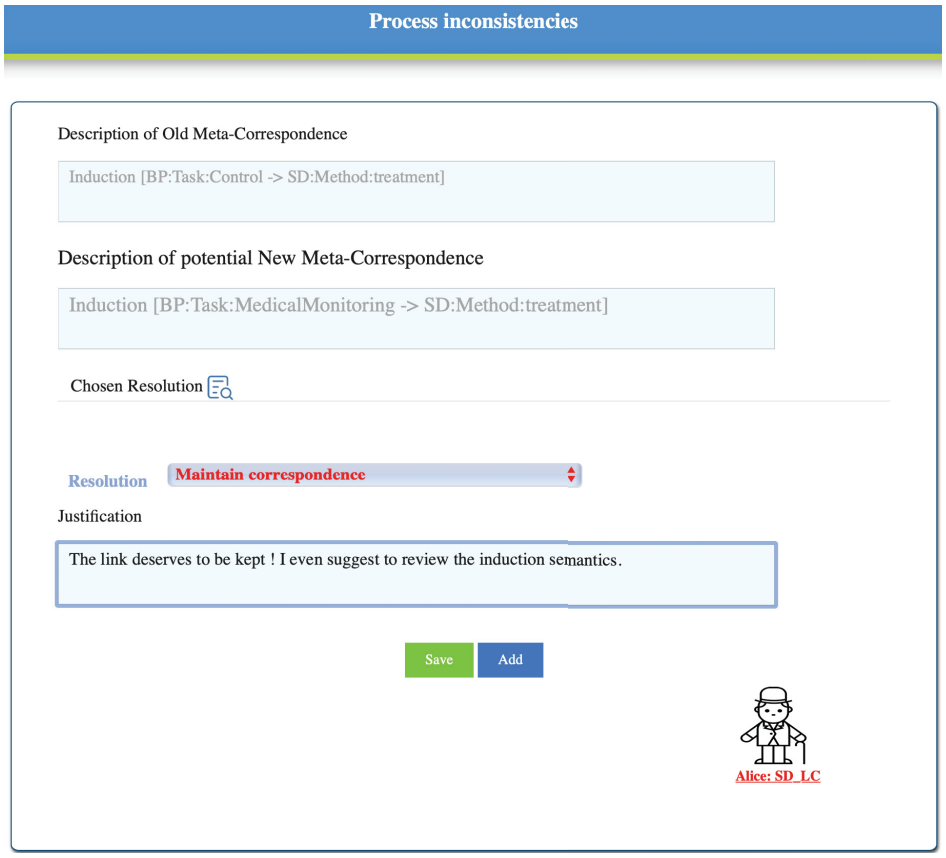


FIGURE 7.24 – Choix de la résolution à appliquer pour le changement *rename Task :Control*.

7.4.3.4 Changements N° 4, 5, 6, 7 et 8 : Ajout de quatre *Field* au modèle ER

Les ajouts des *Field HormonalEx*, *ImmunologicalEx*, *SerologicalEx*, et *FunctionalEx* (cf. Figure 7.20) sont traités de façon supervisée ; en effet, le catalogue de résolutions prévoit de supprimer la correspondance impliquant la classe parente et d'étudier la possibilité de forcer le lien pour ses sous-classes.
Field physio[4] est impliqué dans la correspondance : **Induction [BP :Task :ExecuteAnalysis, SD :Method :checkUp − > ER :Field :physiologicalExamination].** Ainsi, cette correspondance est forcée entre les sous-classes de *physiologicalExamination*, ce qui produit les correspondances suivantes :

- Induction [BP :Task :ExecuteAnalysis, SD :Method :checkUp − > ER :Field :HormonalExamination]
- Induction [BP :Task :ExecuteAnalysis, SD :Method :checkUp − > ER :Field :ImmunologicalExamination]
- Induction [BP :Task :ExecuteAnalysis, SD :Method :checkUp − > ER :Field :SerologicalExamination]
- Induction [BP :Task :ExecuteAnalysis, SD :Method :checkUp − > ER :Field :FunctionalExamination]

4. Sur la Figure 7.8, l'élément est nommé *physio* mais sur l'outil HMCS-Collab, on a détaillé le nom « *physiologicalExamination* ».

7.5 Retours sur le cas d'étude

7.5.1 Synthèse

L'approche CAHM a été mise en oeuvre sur deux cas d'étude : le système de gestion de conférences (CMS) (présenté dans le chapitre 5) et le service d'urgence (SU) présenté dans ce chapitre. Pour CMS, nous avons considéré trois points de vue métier alors que pour le SU nous avons considéré initialement trois points de vue et avons ajouté un quatrième point de vue plus tard pour évaluer l'approche en cas de changement de type global.
En ce qui concerne la taille des modèles, dans le système SU, les modèles ont une taille moyenne de 45 concepts (sans compter les sous-éléments caractérisant chaque concept) alors que pour le système CMS, la taille moyenne des modèles est de 10 concepts. Les méta-correspondances approuvées pour chaque système sont au nombre de 7 en moyenne ; la reproduction de ces dernières au niveau modèle produit environ 6200 correspondances et le filtrage sémantique des relations garde une trentaine de correspondances.
La persistance du modèle de correspondances lors des évolutions de modèles permet de fluidifier le maintien de la cohérence ; les résolutions fournies, qu'elles soient supervisées ou automatiques, l'exploitent et mettent à jour ses correspondances. Avec le système SU, la gestion de l'évolution s'est faite automatiquement par l'outil sauf pour l'ajout du méta-modèle et le renommage d'un élément de modèle. Pour le premier changement, l'implication des acteurs est nécessaire puisqu'il faut établir de nouvelles méta-correspondances, tandis que pour le deuxième changement, la reconsidération de la sémantique des relations pourrait nous dispenser de la détection de fausses incohérences.

7.5.2 Menaces de validité

Dans cette section, nous discutons des menaces de validité de notre approche ; nous les classifions selon qu'elles constituent un biais possible à (i) la validité interne, (ii) la validité externe ou (iii) la validité de la conclusion comme proposé par [WOHLIN et al., 2012].
La validité interne mesure le fait que si une relation est observée entre le traitement et le résultat, il s'agit bien d'une relation causale et qu'elle n'est pas le résultat d'un facteur sur lequel nous n'avons aucun contrôle ou que nous n'avons pas mesuré. La validité externe concerne la généralisation des résultats. S'il existe une relation causale entre la construction de la cause et l'effet, le résultat de l'étude peut-il être généralisé en dehors de son champ ? La validité de la conclusion concerne la relation entre le traitement et le résultat. Elle permet d'assurer qu'il existe une relation statistique, c'est-à-dire avec une signification donnée.

7.5.2.1 Validité interne

Dans notre validation expérimentale, nous avons impliqué des participants du monde académique et industriel. En effet, notre validation s'est basée sur des modèles élaborés par des équipes de recherche travaillant dans des thématiques associées aux points de vue pour lesquels ils ont proposé des modèles. Ceci nous permet d'assurer que les modèles respectent les règles de construction des modèles de ces points de vue ; ceci d'autant plus que la description des modèles a été faite en collaboration avec l'équipe médicale d'un service d'urgence pour assurer que les modèles reflètent les vrais besoins métier du domaine d'application.
En ce qui concerne les acteurs ayant mis en oeuvre l'expérimentation, nous avons choisi des doctorants en informatique mais avec des connaissances variées en IDM. Ceci dans l'objectif de simuler les différences dans les niveaux d'expertise des concepteurs métier dans le domaine de l'IDM et de voir la faisabilité de l'approche même avec des acteurs novices en IDM. Au début, nous n'avons pas informé les participants de ce que nous étudions pour éviter de les influencer ; une fois qu'ils ont proposé les méta-correspondances et que le modèle de correspondances a été établi, nous leur avons montré le résultat (modèle de correspondances) et avons enchaîné avec la deuxième phase. Le déroulement de l'expérimentation a été mené la première fois sans l'outil ;

de ce fait les collaborations ont été faites en utilisant un outil d'édition collaborative en ligne, ce qui nous a permis d'une part de définir les besoins de l'outil en termes de collaboration et d'autre part d'élaborer le *golden* M1C, i.e., modèle élaboré manuellement contenant les correspondances jugées essentielles par les concepteurs, afin de l'exploiter pour mesurer la qualité du M1C produit par l'outil.

7.5.2.2 Validité externe

Notre approche a été mise en oeuvre sur des modèles émanant de cinq points de vue métier (conception logicielle, processus métier, système multi-agents, base de données relationnelles, présentation d'IHM). Nous sommes convaincus que l'approche est applicable à tout autre DSL, à condition qu'il soit conforme au MOF et que des liens sémantiques puissent être définis entre les éléments des modèles issus de ces langages. Cependant, nous n'avons pas encore réalisé l'étude avec d'autres catégories de modèles (e.g., modèles comportementaux, modèles de tests, etc.).
Un autre point important concernant la validation externe correspond au passage à l'échelle ; les mécanismes de réduction de l'effort mis en place par CAHM visent à alléger les collaborations même avec des modèles de grande taille. Cependant, l'outil risque de prendre plus de temps pour produire les correspondances (reproduction par produit cartésien et filtrage sémantique). Aussi, le traçage des changements opérés sur les modèles risque de devenir un facteur de bruit.

7.5.2.3 Validité de la conclusion

Nous avons mis en oeuvre notre expérimentation sur deux cas d'étude, en impliquant 4 acteurs pour chaque cas. D'un point de vue statistique, il serait certainement préférable d'inclure davantage de participants et de mettre en oeuvre l'approche sur plus de cas d'étude avec des tailles de modèles et des points de vue métier variés afin d'obtenir plus de données pour tirer les conclusions de notre étude. Cependant, les résultats obtenus sont suffisants pour valider l'intérêt de l'approche et justifier les axes d'amélioration qui doivent être pris en compte par la suite.

7.6 Conclusion

Dans ce chapitre, nous avons proposé une validation expérimentale de notre approche CAHM. Plus précisément, nous avons décrit l'application de notre démarche sur un cas d'étude issu d'un service d'urgence d'un hôpital (SU). Ce cas d'étude nous a permis de valider un certain nombre de propriétés ; nous avons principalement validé la pertinence des concepts de MMCollab et l'apport de la collaboration lors de l'alignement des modèles dans les deux phases de CAHM.
La collaboration permet un gain en temps et en flexibilité. En effet, dans CAHM, l'alignement des modèles est effectué collectivement et les actions des acteurs sont combinées, de sorte qu'une solide connaissance de tous les points de vue du système global n'est pas nécessaire ; celles des acteurs se complètent. Chaque acteur peut ainsi concevoir librement sa solution en dehors de toute considération des autres points de vue. Cela permet de satisfaire les parties prenantes puisqu'ils n'ont pas eu à faire une analyse approfondie des autres méta-modèles.
La collaboration apporte aussi un gain en terme de qualité des décisions. Chaque acteur traite son point de vue et la possibilité qui lui est offerte de discuter avec les autres concepteurs lui permet de mieux appréhender non seulement le système dans sa globalité, mais aussi son propre point de vue sur celui-ci. Par ailleurs, la définition des méta-correspondances en collaboration sur deux étapes (proposition, évaluation) répartit les responsabilités entre les coordinateurs locaux concernés et renforce le fait que les correspondances établies sont pertinentes pour le système étudié.
Bien entendu, certains inconvénients sont inhérents à la collaboration dans ce contexte. Les acteurs ont par exemple relevé le problème des délais de communication. Nous sommes conscients que ces biais doivent être traités pour assurer un alignement collaboratif fructueux.

Conclusion

L'objectif de cette partie était de fournir une validation expérimentale de notre approche en mettant en oeuvre à la fois les processus de CAHM et en instanciant les concepts de MMCollab. Nous avons illustré et validé l'approche conceptuelle du travail de thèse, ainsi que son implémentation, sur un cas d'étude.
Dans le chapitre 6, nous avons présenté le prototype HMCS-Collab supportant l'approche CAHM en détaillant ses modules et l'architecture technique sur laquelle il est fondé. Ensuite (dans le chapitre 7), nous avons appliqué l'approche sur un cas d'étude réel issu du domaine médical, à savoir un service d'urgence, en exploitant l'outillage fourni.

Dans le chapitre suivant, nous faisons une brève synthèse de notre travail en rappelant les contributions de cette thèse et en les positionnant par rapport aux questions de recherche initiales et aux travaux de la littérature ; par la suite, nous évoquons les limites de l'approche et ses pistes d'amélioration conceptuelles, techniques et applicatives.

Chapitre 8

Conclusion et Perspectives

> What we call the beginning is often the end. To make an end is to make a beginning. The end is where we start from.
>
> *T. S. Eliot.*

Sommaire

8.1 Rappel des questions de recherche 169
8.2 Bilan des contributions conceptuelles 170
8.3 Positionnement des contributions 171
8.3.1 Retour sur les questions de recherche 171
8.3.2 Positionnement par rapport à la littérature 172
8.4 Limites et perspectives de travail 173
8.4.1 Limites des contributions 173
8.4.2 Perspectives conceptuelles 173
8.4.3 Perspectives techniques et applicatives 174

Dans ce chapitre de conclusion, nous commençons par rappeler les questions de recherche de notre travail de thèse (section 8.1). Dans la section 8.2, nous présentons le bilan de nos contributions, puis nous positionnons notre travail par rapport aux questions de recherche de cette thèse et à la littérature dans la section 8.3. Dans la section 8.4, nous abordons les limites de nos contributions et présentons nos perspectives conceptuelles, techniques et applicatives à court et moyen termes.

8.1 Rappel des questions de recherche

Dans les chapitres précédents de ce manuscrit, nous avons décrit une approche permettant de produire une vision globale sur les points de vue métier d'un système en optant pour une solution collaborative qui supporte la participation des acteurs des différents domaines métier. La démarche proposée dans l'approche passe par l'élaboration des correspondances entre les modèles. Ceci présente des défis relatifs (i) à *l'établissement des correspondances inter-modèles*; en effet, les correspondances doivent refléter les liens existants entre modèles, elles ne doivent pas être limitées ni en terme d'arité ni en type de relation qui les précise (ii) au *maintien de la cohérence inter-modèles en cas d'évolution des modèles*; en effet les interactions entre ces domaines impliquent de détecter les incohérences inter-domaines durant la conception, et de répercuter les changements opérés dans un modèle sur les autres modèles (iii) à *l'hétérogénéité des points de vue métier impliqués*, dont les représentations ne sont pas conformes au même méta-modèle. De ce fait, l'élaboration de la vue globale doit être réalisée collaborativement par les acteurs métier pour assurer le reflet de leurs préoccupations.

Comme nous l'avons vu dans l'introduction de ce manuscrit, la question de recherche globale de notre travail de thèse est : **Comment exprimer et assurer la cohérence entre les modèles hétérogènes d'un système dans le cadre d'une conception évolutive impliquant plusieurs acteurs métier ?**. Elle se décompose en sous-questions exprimées comme suit :

QR 1. Comment capturer les liens entre modèles hétérogènes (i.e., avoir une vision globale) lors d'une conception multi-vue, en impliquant les acteurs métier ? Question se décomposant elle-même en deux sous-questions :

- QR1.1. Où intervient la collaboration entre les différents acteurs métier lors de l'élaboration des liens inter-modèles ?
- QR1.2. Quelle assistance fournir aux acteurs pour définir les liens entre les modèles ?

QR 2. Comment maintenir la vision globale d'un système lors des évolutions des (méta-)modèles de points de vue métier impliqués ? Cette question se décompose elle-même en deux sous-questions :

- QR2.1. Où intervient la collaboration entre les différents acteurs métier lors du maintien de la cohérence de l'ensemble ?
- QR2.2. Quelle assistance fournir aux acteurs pour maintenir la cohérence de l'ensemble des modèles ?

8.2 Bilan des contributions conceptuelles

Pour répondre aux questions de recherche rappelées ci-dessus, nous avons développé trois contributions majeures.

(C1) **MMCollab, méta-modèle regroupant les concepts clés d'une session collaborative de prise de décision en groupe**

Nous avons proposé en premier lieu un méta-modèle, MMCollab, dédié à la description des concepts clés de la collaboration en général et de la prise de décision en groupe en particulier. MMCollab structure les prises de décision collaboratives en classifiant les concepts en cinq paquetages traitant : les acteurs, le choix de la politique de prise de décision, la collecte des propositions sur lesquelles vont porter les évaluations individuelles et les décisions collectives relatives à ces propositions.
Nous avons élaboré ensuite une liste extensible de politiques de décision, qui sont des instances du concept GDMPattern introduit dans MMCollab. L'intérêt est d'offrir un ensemble de politiques prêtes à l'utilisation et qui répondent aux stratégies récurrentes de prise de décision en groupe.

(C2) **CAHM, approche pour l'alignement collaboratif de modèles**

Nous avons exploité le méta-modèle MMCollab pour proposer une approche nommée CAHM, pour Collaborative Alignement of Heterogeneous Models. CAHM permet d'établir des liens inter-modèles et de les maintenir en cas d'évolution des modèles. Pour cela, elle offre un (i) processus global semi-automatique où les utilisateurs, acteurs métier du système, établissent collaborativement les liens typés au niveau méta-modèle (appelés méta-correspondances) en les discutant et en jugeant de leur pertinence, (ii) un processus outillé de propagation de ces méta-correspondances au niveau modèle (appelées correspondances) pour produire un modèle de correspondances qui reflète la cohérence du système global. Pour propager les méta-correspondances, l'outil (cf. C3) se base sur les expressions sémantiques associées aux types de relations qui caractérisent les correspondances. L'outil capture les changements opérés sur les modèles et leurs méta-modèles respectifs et vérifie les impacts de ces changements sur le modèle de correspondances et les autres modèles. Nous proposons aussi un catalogue de résolutions des incohérences pouvant se produire suite aux changements. Dans le cas où la résolution peut se faire automatiquement, l'outil se charge de la mettre en oeuvre. Sinon, les acteurs métier sont impliqués pour choisir collaborativement quelle résolution adopter.

(C3) **HMCS-Collab, outil supportant l'approche CAHM**

Pour mettre en oeuvre l'approche, nous avons développé un outil nommé HMCS-Collab (pour **H**eterogeneous models **M**atching and **C**onsistency management **S**uite - **Collab**orative version). HMCS-Collab implémente à la fois le langage de MMCollab, les politiques de décision définies par instanciation du concept GDMPattern de ce méta-modèle, et le processus global de CAHM.
HMCS-Collab assure principalement cinq fonctionnalités : (i) le déroulement des sessions collaboratives lors de l'élaboration des méta-correspondances ; (ii) la propagation des méta-correspondances au niveau modèle en s'appuyant sur les expressions sémantiques des relations ; (iii) la détection des changements opérés sur les modèles source ; (iv) l'application des résolutions automatiques présentes dans le catalogue des résolutions en cas d'incohérences ; (v) le déroulement des sessions collaboratives lors de l'élaboration de nouvelles résolutions ou du choix parmi une liste de résolutions applicables pour une incohérence donnée.

8.3 Positionnement des contributions

8.3.1 Retour sur les questions de recherche

Dans cette section, nous illustrons comment les contributions de ce travail répondent aux questions de recherche rappelées ci-dessus.

QR1.1. Où intervient la collaboration entre les différents acteurs métier lors de l'élaboration des liens inter-modèles ?
L'intervention de la collaboration lors de l'établissement des correspondances inter-modèles est précisée dans le processus de CAHM - phase 1 ; en effet les acteurs métier interviennent pour proposer des méta-correspondances. Le déroulement de cette collaboration se fait en instanciant les concepts de MMCollab.

QR1.2. Quelle assistance fournir aux acteurs pour définir les liens entre les modèles ?
L'outil HMCS-Collab implémente le langage de MMCollab et ses politiques de décision ; ce qui permet d'automatiser l'élaboration des décisions collectives pour les méta-correspondances. HMCS-Collab implémente aussi le mécanisme de propagation qui permet d'établir les correspondances inter-modèles.

QR2.1. Où intervient la collaboration entre les différents acteurs métier lors du maintien de la cohérence de l'ensemble ?
La collaboration intervient dans le processus CAHM - phase 2 lors du traitement des incohérences engendrées par les changements. En effet, les acteurs métier décident ensemble quelle résolution appliquer pour gérer une incohérence dans le cas où plusieurs ou aucune solution n'est définie.

QR2.2. Quelle assistance fournir aux acteurs pour maintenir la cohérence de l'ensemble des modèles ?
HMCS-Collab implémente les règles des résolutions précisées dans le catalogue des résolutions pour permettre leur application. L'outil assiste aussi les acteurs en se chargeant de détecter les types de changement supportés par l'approche (et qui sont précisés dans le méta-modèle MMC). Il permet aussi d'établir les graphes de dépendances de chaque changement.

Le tableau 8.1 positionne nos contributions par rapport aux questions de recherche.

TABLEAU 8.1 – Positionnement des contributions par rapport aux questions de recherche

	QR1		QR2	
	QR1.1	QR1.2	QR2.1	QR2.2
Contribution 1	✓		✓	
Contribution 2	✓	✓	✓	✓
Contribution 3		✓		✓

8.3.2 Positionnement par rapport à la littérature

Les schémas de la Figure 8.1 illustrent le positionnement de notre approche par rapport aux travaux de la littérature traitant l'alignement des modèles.

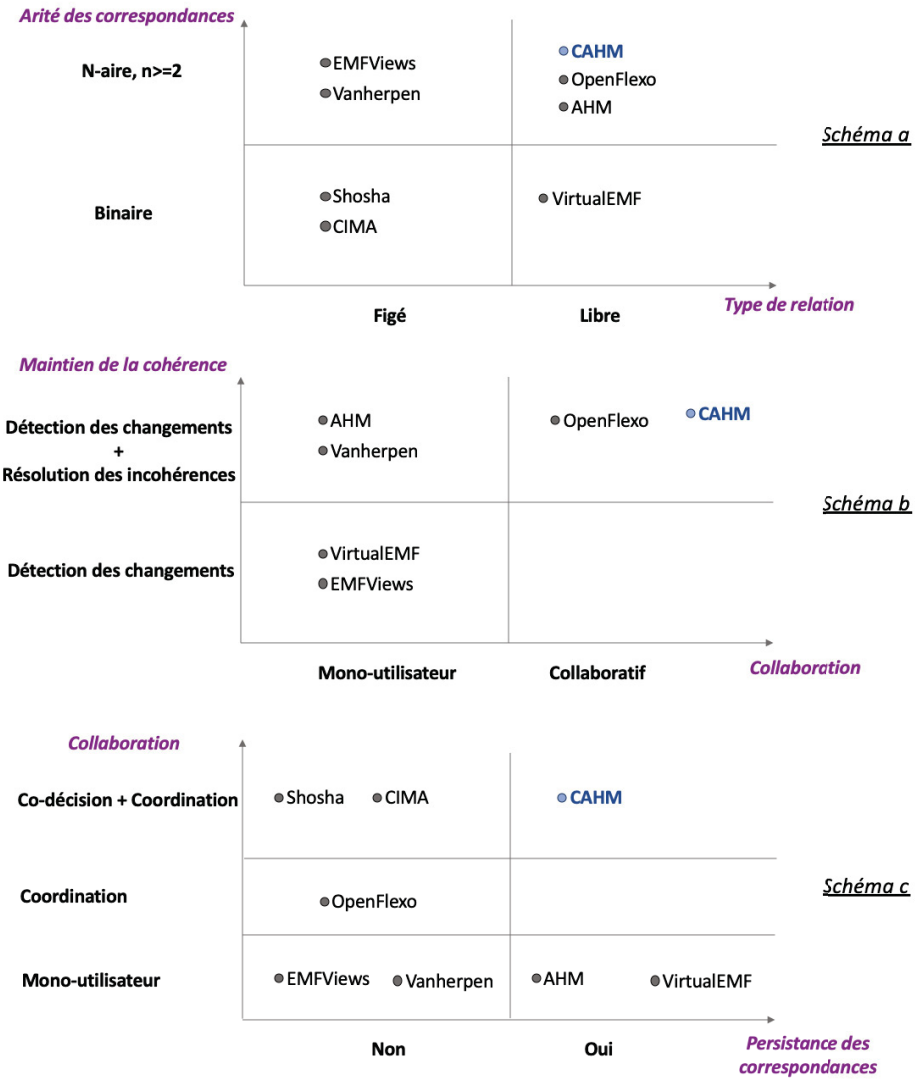


FIGURE 8.1 – Schémas récapitulatifs de positionnement de CAHM par rapport à l'état de l'art.

Concernant l'arité des correspondances et le type des relations proposées (schéma a), trois approches (CAHM, OpenFlexo et AHM) se distinguent en supportant des correspondances n-aires avec un ensemble de relations extensible selon le besoin. Deux parmi ces trois approches (OpenFlexo et CAHM) considèrent à la fois la détection des changements, la résolution des incohérences liées aux changements et le support à la collaboration (schéma b). Cependant, seule CAHM intègre à la fois la persistance des correspondances, la coordination et la co-décision dans l'élaboration des correspondances et le traitement des changements (schéma c).

8.4 Limites et perspectives de travail

8.4.1 Limites des contributions

Nous ne revendiquons pas la complétude de notre approche. Des limitations conceptuelles, techniques et applicatives sont à mentionner. En effet, nous avons fixé plusieurs hypothèses limitatives pour cadrer notre approche.
Nous avons fixé (via le méta-modèle MMC) les changements possibles pour les (méta-)modèles des points de vue métier. Toutefois, nous n'avons considéré que les changements au niveau des concepts et avons omis volontairement les changements au niveau des multiplicités et types des concepts. Cette limitation est légitime puisque les expressions sémantiques définies pour les relations ne se basent pas sur le type, la multiplicité ou le contexte d'un concept. Néanmoins, ces informations peuvent être utiles et nous amener par la suite à les considérer. Nous avons aussi considéré que tout point de vue métier est constitué d'un méta-modèle et un modèle qui lui est conforme. Nous avons fait cette hypothèse car la problématique de cette thèse porte sur l'alignement de modèles hétérogènes, même si, dans la réalité, plusieurs modèles peuvent être conformes au même méta-modèle dans le cas d'un Generic Purpose Language (GPL) par exemple.
Une autre limitation concerne dans l'outillage fourni. D'une part, les méta-modèles sont contraints d'être conformes au méta-méta-modèle MOF pour pouvoir être importés dans l'outil. D'autre part, l'efficacité de l'outil et sa capacité à alléger les tâches humaines dépendent principalement de la pertinence des expressions sémantiques associées aux relations. La troisième limitation concernant l'outillage vient du fait que HMCS-Collab est un prototype encore dans les phases de beta-testing et contient des modules qui requièrent une amélioration pour optimiser les algorithmes implémentés et assurer un bon rendement lors du passage à l'échelle.
Concernant la validation de l'approche, nous l'avons mise en oeuvre par l'intermédiaire d'exemples concrets et d'une étude de cas représentative mais simplifiée. Comme précisé dans le chapitre 7, les (méta-)modèles du cas d'étude ont des tailles moyennes qui ne correspondent pas aux tailles réelles des modèles de projets industriels de grande envergure. Cette validation nous a permis de vérifier l'applicabilité de l'approche et de mesurer ses point forts et ses faiblesses par rapport à une approche d'alignement centralisée. Mais elle ne nous permet pas de valider le passage à l'échelle de l'approche ni son applicabilité par des acteurs métier étant donné que le déroulement de l'approche sur le cas d'étude a été réalisé par des acteurs du domaine académique (des doctorants en informatique).

8.4.2 Perspectives conceptuelles

Nous commençons par exposer différentes perspectives conceptuelles dans le prolongement de nos contributions, c'est-à-dire des perspectives pour améliorer nos contributions, relatives notamment aux limites identifiées ci-dessus, ou de nouvelles pistes à intégrer pour assurer la pertinence de l'approche proposée.
Concernant CAHM, nous souhaitons intégrer les types de changement qui ont été écartés lors de la définition du processus général d'alignement des modèles (e.g., changements partiels de niveau méta-modèle, changements composites, etc). Nous comptons aussi fournir des mécanismes pour assurer le bon déroulement de l'approche pour des domaines d'application impliquant des acteurs novices en IDM, sachant que la définition des méta-correspondances passe par la manipulation des méta-modèles et leurs méta-éléments. Pour cela, nous pouvons commencer par fournir des tutoriels simples et concis pour dérouler l'approche sans mettre l'accent sur l'aspect IDM et entretenir une foire aux questions en ligne.
L'acteur humain est essentiel lors du déroulement de CAHM. En effet, le processus de CAHM se base sur les connaissances humaines pour (i) la définition formelle des relations sémantiques, (ii) la définition des correspondances et (iii) le traitement des incohérences. Nous sommes convaincus que l'application de l'Intelligence Artificielle conjointement à l'IDM pourrait alléger le travail humain tout en gardant la perspective collaborative. Notre objectif est donc d'étendre notre approche

en intégrant des techniques d'intelligence artificielle pour gérer la cohérence inter-modèles. Ceci peut passer par exemple par la définition d'une méthode d'apprentissage pour aider dans la définition des relations sémantiques, la définition d'un système de recommandations pour le choix de politiques de décision et même pour l'aide à la prise de décision en groupe. Nous envisageons aussi d'étendre l'approche pour supporter l'alignement d'un ensemble de modèles (hétérogènes et homogènes) en incluant une phase d'unification des modèles homogènes avant d'appliquer les deux phases de CAHM.

8.4.3 Perspectives techniques et applicatives

Pour améliorer l'outillage support, nous avons identifié les pistes de travail suivantes : (i) l'intégration de mécanismes pour détecter les changements composites et traiter les incohérences qui en résultent, sachant que les modèles sont définis dans différents outils de modélisation, (ii) le déploiement de la solution sur un serveur de production en donnant l'accès à la communauté pour bénéficier de l'approche et tester son passage à l'échelle.
Comme prolongement applicatif de l'approche, nous souhaitons élargir les expérimentations de validation de l'approche en nous appuyant sur un cas industriel avec des collaborateurs métier afin de faire une mise en oeuvre avec des mesures de performance de l'approche et de l'outillage fourni. Nous souhaitons aussi appliquer l'approche sur des modèles d'analyse, de tests, ou de comportements comme fait avec B-COoL [Larsen et al., 2015] ou ModHel'X [Hardebolle et Boulanger, 2007; Boulanger et al., 2013] et voir l'impact de cette transposition sur le processus de CAHM.
Nous comptons aussi - et ce pour mesurer la validité de MMCollab - l'appliquer sur d'autres scénarios de prise de décision et faire des transpositions à d'autres domaines de recherche. Pour cela, nous envisageons dans un premier temps d'exploiter MMCollab pour l'élaboration collaborative des modèles source.

Annexes

Annexe A

Méthodes de prise de décision en groupe

Les méthodes de prise de décision aident à spécifier comment les préférences des différents participants sont prises en compte. Nous distinguons deux catégories de méthodes : méthodes *structurées* versus méthodes *non structurées* [MALAVOLTA et al., 2014]. Dans le cas de méthodes structurées, les préférences individuelles sont agrégées comme dans AHP [SAATY, 1988, 2008] grâce à des calculs mathématiques qui sont parfois complexes. Dans le cas de méthodes moins structurées (ou semi formelles), comme le Brainstorming ou le vote, les règles d'unanimité, de majorité, de pluralité et de minorité sont utilisées.
Dans cette annexe, nous donnons un aperçu des méthodes de prise de décision en groupe les plus fréquemment utilisées.

A.1 Méthodes non ou peu structurées

Technique du groupe nominal. La technique du groupe nominal (abrégée TGN) [DELBECQ et VAN DE VEN, 1971] est un processus de groupe impliquant l'identification des problèmes, la production de solution, et la prise de décision. Elle peut être utilisée dans des groupes de différentes tailles, qui veulent prendre leur décision rapidement, par exemple par un vote, mais qui veulent la prise en compte des opinions de chacun (par opposition au vote traditionnel, où seul le plus grand groupe est considéré). Le procédé de dépouillement est la différence. Tout d'abord, chaque membre du groupe donne son point de vue sur sa solution, accompagné d'une brève explication. Ensuite, des solutions doubles sont éliminées de la liste des solutions, puis les membres procèdent alors à un classement des solutions.
Il existe des variations sur la façon dont on utilise cette technique. Dans la méthode de base, les numéros reçus par chaque solution sont additionnés, et la solution la plus haute du classement total est sélectionnée comme la décision finale.

Delphi. La méthode Delphi [DALKEY, 1969] est un processus de communication permettant de résoudre un problème. Elle consiste à dégager un consensus sur des sujets précis, grâce à l'interrogation d'experts. Les experts sont en mesure d'apporter un éclairage sur des secteurs d'incertitude en vue d'une aide à la décision. La méthode Delphi s'appuie sur l'approche dialectique par enquête : thèse (on établit une opinion), antithèse (opinion contradictoire) et une synthèse (un nouveau consensus).

SWOT. Dans la méthode Strengths - Weaknesses - Opportunities - Threats (SWOT) [HILL et WESTBROOK, 1997], les avantages et inconvénients de chaque alternative sont identifiés et puis comparés par les participants. L'alternative ayant le plus d'avantages et le moins d'inconvénients est préférée. Cette méthode convient aux décisions simples avec peu d'alternatives.

A.2 Méthodes structurées

Dans les méthodes semi formelles (ou peu structurées), les participants discutent des problèmes et indiquent leurs préférences sur un ensemble prédéterminé d'alternatives ; dans les méthodes structurées, Ils comparent des alternatives et des critères et pondèrent les alternatives en fonction des critères avant de prendre les décisions.

AHP. La méthode AHP , proposée par SAATY [1994], est inspirée de la façon naturelle dont les humains agissent ou réagissent lorsqu'ils résolvent un problème de prise de décision en le décomposant en sous-problèmes qui sont ensuite agrégés pour obtenir une recommandation finale. Cette méthode permet à l'expert d'organiser et d'exprimer ses jugements ou sentiments sur les éléments qui constituent un problème de prise de décision (c.-à-d. Objectif, critères d'évaluation et alternatives) par une comparaison par paire pour avoir un moyen facile et contrôlable de comprendre et de résoudre le problème. Fondamentalement, la méthode AHP guide l'expert pour décomposer le problème de décision à travers trois phases pour générer les priorités sur l'ensemble des alternatives :

1. Définir le problème de décision comme une structure hiérarchique.
2. Construire un ensemble de matrices de comparaison par paire.
3. Calculer les priorités des éléments dans la structure hiérarchique.

Dans AHP, l'ensemble des alternatives envisagées et l'ensemble des critères du problème de décision doivent être organisés en une structure hiérarchique qui a un minimum de trois niveaux. Au premier niveau qui est le sommet de la hiérarchie, est placé le but du problème de décision ; au deuxième niveau sont placés l'ensemble des critères ; et au troisième niveau sont placés l'ensemble des alternatives. Cette structure à trois niveaux peut être étendue avec d'autres niveaux pour représenter des sous-critères du problème de prise de décision.

TOPSIS. TOPSIS est une méthode dont le but est de pouvoir classer par ordre de choix un certain nombre d'alternatives sur la base d'un ensemble de critères favorables ou défavorables. Elle a été développée par Hwang et Yoon [HWANG et YOON, 1981]. Son principe est basé sur le concept de programmation de compromis qui vise à définir une solution idéale comme point de référence selon les préférences des experts, puis à rechercher les solutions dont les attributs sont les plus proches d'un attribut de solution idéale. Fondamentalement, TOPSIS définit deux alternatives fictives appelées solution idéale positive et solution idéale négative. La solution idéale positive reflète la meilleure évaluation des alternatives sur chaque critère tandis que la solution idéale négative reflète les pires. Ensuite, une distance géométrique (généralement la distance euclidienne) de chaque alternative de décision est calculée pour les solutions idéales positives et négatives. L'alternative la plus proche de la solution idéale positive et la plus éloignée de la solution idéale négative est considérée comme la meilleure alternative parmi l'ensemble des alternatives, par conséquent, cette distance est utilisée comme score pour classer les alternatives.

MAUT. Multi-Attribute Utility Theory (MAUT) [VON WINTERFELDT et FISCHER, 1975] combine des mesures variées de coût, de risqu et d'avantages, ainsi que les préférences des parties prenantes. MAUT repose sur l'utilisation de fonctions utilitaires en transformant les divers critères en une échelle commune sans dimension (0 à 1) appelée «utilitaire» à attributs multiples. Pour identifier l'option préférée, il faut multiplier le score d'utilité de chaque option normalisée par le poids du critère correspondant, puis additionner les résultats obtenus pour tous les critères de l'option. L'alternative préférée a le score total le plus élevé. MAUT est généralement utilisée lorsque des informations quantitatives sont connues pour chaque alternative, ce qui peut donner lieu à des estimations plus précises de la performance.

Annexe B

Compléments sur l'approche CAHM

Dans cette annexe, nous présentons des compléments sur l'approche CAHM. Dans B.1, nous détaillons l'algorithme de calcul d'impact d'un changement évoqué dans la Section 5.5.5. Dans B.2, nous illustrons la construction du graphe de dépendances d'un changement ; ce graphe permet de recenser les éléments impactés (directement ou indirectement) par un changement (cf. Section 5.5.6), puis, nous présentons dans B.3 le catalogue des recommandations de résolutions des incohérences en fonction du type de changement.

B.1 Algorithme du calcul d'impact d'un changement

Algorithme 1 : Calculate change processing order

```
   Input      :c // a change
               stChanges // stack of changes
   Output     :stChanges
 1 if c.type == 'Global' then
 2 |  stChanges.insertAtTop(c) // c has the highest priority
 3 else // c.type == 'Partial'
 4 |  if c.subtype!= 'A' // c.subtype == 'D' or c.subtype == 'M' then
 5 |  |  ce = c.getChangedElement()
 6 |  |  ce_corresps = getCorresps(ce) // get related elements to ce in M1C
   |  |   (directly and indirectly)
 7 |  |  for each corresp in ce_corresps do
 8 |  |  |  if verifySemantics(corresp) == False then
 9 |  |  |  |  ce.getInvalidCorresps().add(corresp)
10 |  |  |  end
11 |  |  end
12 |  |  if ce.getInvalidCorresps().size() > 0 then
13 |  |  |  stChanges.position(c) // Put c in its right position and
   |  |  |   Shift the rest of higher changes by a position to the
   |  |  |   top; a change c1 has higher priority than c if
   |  |  |   c1.getInvalidCorresps()>c.getInvalidCorresps()
14 |  |  end
15 |  else // c.subtype == 'A'
16 |  |  stChanges.insertAtBottom(c)// c has the lowest priority
17 |  end
18 end
```

Dans l'activité 2 de la phase 2 de CAHM (cf. Section 5.5.5), HMCS-Collab calcule l'ordre de traitement d'un changement, qui dépend du type de changement. Cet algorithme prend en entrée un changement c et la pile des changements *stChanges*.
Un changement global concerne un modèle ou un méta-modèle. Il inclut ainsi un ensemble d'éléments supprimés (D), ajoutés (A) ou bien supprimés et ajoutés (DA). Il change la structure du point de vue et doit donc être traité prioritairement (il est ajouté au sommet de la pile - ligne 2 de l'algorithme 1).
Un changement partiel est ajouté à la pile des changements et attend la validation du manager avant de lancer l'activité suivante. Pour chaque changement partiel, on calcule son ordre d'importance (qui correspond à son ordre de traitement dans la pile) en fonction du nombre de correspondances qu'il affecte. Si le changement est un ajout d'un élément au modèle, il est inséré en bas de la pile (ligne 16 de l'algorithme 1). Sinon, pour un changement de type suppression (D) ou modification (M), il est inséré à la bonne position dans la pile en fonction des correspondances qu'il a rendues invalides (lignes 7 à 14 de l'algorithme 1).

B.2 Construction du graphe de dépendances

Le graphe de dépendances a pour objectif de tracer les éléments directement ou indirectement reliés à l'élément changé (qu'il soit un modèle, un méta-modèle ou un élément de modèle). Que le changement soit partiel ou global, le graphe de dépendances concerne uniquement les changements de type suppression et modification étant donné qu'en cas d'ajout de modèle ou de méta-modèle (changements globaux), le graphe n'est pas utile car il faut relancer le processus de mise en correspondance ; et qu'en cas d'ajout d'éléments de modèles (changements partiels), il n'y a pas encore d'éléments dépendants de l'élément en cours d'ajout.

B.2.1 Construction du graphe de dépendances d'un changement partiel

— La racine du graphe est l'élément changé (*ce*).

— Le graphe est constitué des éléments reliés à *ce* dans le modèle de correspondances (directement ou indirectement). Les éléments directement liés à *ce* sont à une distance 1 de *ce* et les autres à une distance égale au nombre minimal d'arêtes atteignables à partir de *ce*.
Chaque nœud est caractérisé par un triplet (x, y, z) où x est le nom de l'élément, y la distance de cet élément à *ce*, et z la liste des correspondances qui font que x est à cette distance de *ce*.

— Exemple : La Figure B.1 montre un exemple de modèle de correspondances (M1C) et du graphe associé à l'élément « c ». Le modèle de correspondances est défini entre trois modèles dont les éléments sont nommés respectivement avec les alphabets latin, grec et cyrillique. C1, C2, C3 et C4 sont les noms des correspondances. α, ë et γ étant directement liés à « c », ils sont à une distance de 1. л est à une distance de 2 puisqu'il est indirectement lié à « c » alors que d et θ ne sont pas inclus dans le graphe de « c ».

B.2.2 Construction du graphe de dépendances d'un changement global

Les changements globaux concernés par le graphe de dépendances sont :

Cas 1 : Une suppression de modèle ;

Cas 2 : Une suppression suivie d'ajout de modèle ;

Cas 3 : Une suppression de méta-modèle ;

Cas 4 : Une suppression suivie d'ajout de méta-modèle.

Nous présentons par la suite les cas de suppression (cas 1) et de suppression suivie d'ajout de modèle (cas 2), sachant que nous adoptons le même raisonnement pour la construction des graphes pour le niveau méta-modèle (cas 3 et 4) ; la seule différence est qu'il s'agit de correspondances pour les modèles et de méta-correspondances pour les méta-modèles.

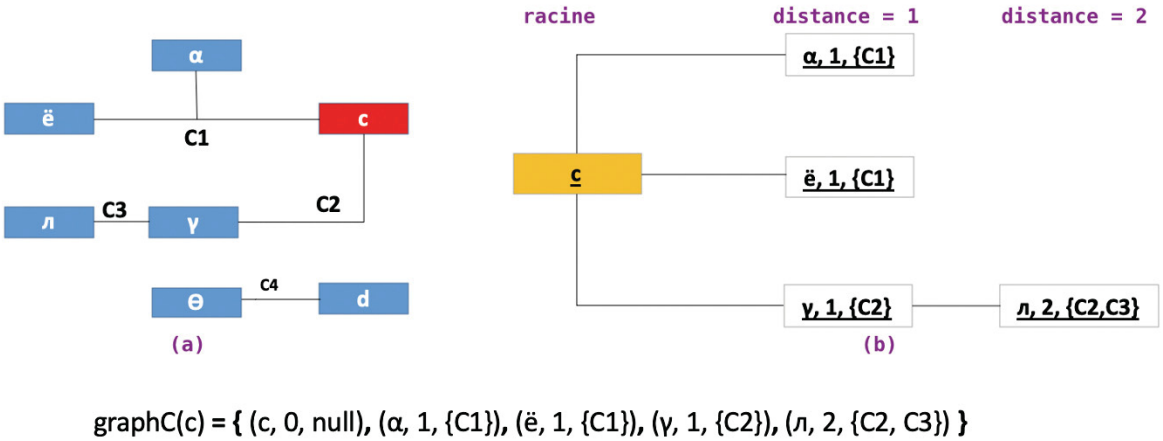


FIGURE B.1 – (a) Exemple d'un modèle de correspondances et (b) du graphe de dépendances associé à l'élément « c ».

On nomme M2 le modèle ajouté et M1 l'ancien, M1 et M2 représentant le même point de vue. M1 et M2 ne sont pas forcément présents conjointement. En effet dans le cas de suppression de modèle (cas 1), on a uniquement M1 et en cas de suppression suivie d'ajout (cas 2) on a M1 le modèle supprimé et M2 le nouveau modèle qui le remplace.

Cas 1 : *Suppression du modèle M1*

- La racine est un élément fictif nommé root(M1).
- Le graphe est constitué de :
  - (i) Éléments e_1 appartenant à M1 et impliqués dans des correspondances. Tous les éléments e_1 sont à une distance 1 de la racine car ils sont des éléments appartenant à M1.
  - (ii) Éléments e_{12} appartenant à des modèles autres que M1 et qui sont en lien (direct ou indirect) avec les éléments e_1. Les éléments e_{12} sont à une distance égale au nombre minimum d'arêtes permettant de les atteindre à partir de root(M1).
- Exemple : Nous considérons le modèle du point de vue PS du système CMS.
  En désignant par M1 le modèle du point de vue PS et d'après la modèle de correspondances du système CMS présenté sur la Figure 5.18, neuf éléments de M1 sont impliqués dans des correspondances ; le graphe de suppression de M1 est illustré sur la Figure B.2. La racine est root(M1), les éléments e_1 sont les 9 éléments du modèle PS, à savoir : *PS :Column :phoneNumber, PS :Column :address, PS :Column :organization, PS :Column :headline, PS :Table :AuthorTable, PS :Table :ReviewTable, PS :Table : ConferenceTable, PS :Column :fullname* et *PS :Column :submissionDate*. Les éléments e_{12} sont tous à une distance 2 puisqu'ils sont directement liés à l'ensemble des éléments e_1.

Cas 2 : *Suppression du modèle M1 suivie de l'ajout du modèle M2 représentant le même point de vue*

- La racine est un élément fictif nommé root(M1).
- Le graphe est constitué de :
  - (i) Éléments e_1 appartenant à M1\M2 [1] et impliqués dans des correspondances. Tous les éléments e_1 constituant le graphe sont à une distance 1 de la racine car ils sont des éléments appartenant à M1 ;
  - (ii) Éléments e_{12} appartenant à des modèles autres que M1 et qui sont en lien avec les éléments e_1. Les éléments e_{12} sont à une distance égale au nombre minimum d'arêtes permettant de les atteindre à partir de root(M1) ;

1. M1\M2 : M1 privé de M2 (ou M1-M2 en notation ensembliste).

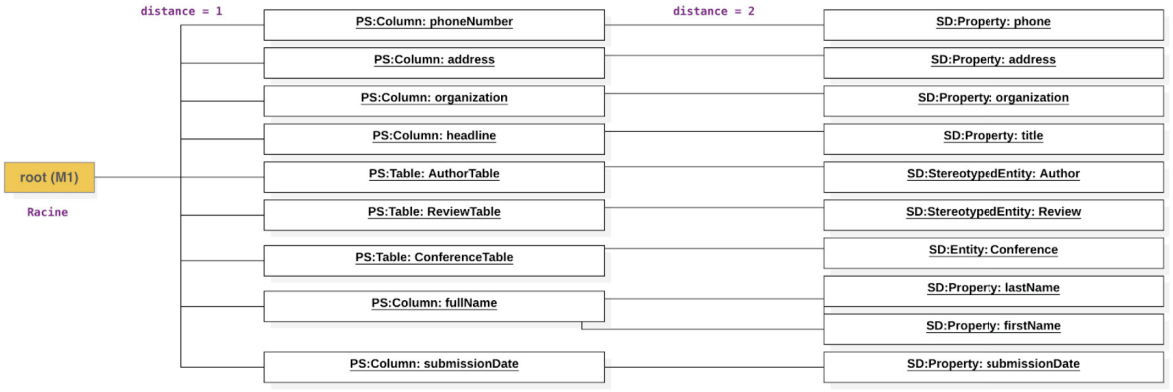


FIGURE B.2 – Graphe de dépendances associé à la suppression du modèle PS du CMS.

(iii) Éléments e_2 appartenant à M2\M1. On considère que les éléments e_2 sont à une distance 1 de la racine étant donné que M1 et M2 représentent le même point de vue.

B.3 Catalogue des résolutions d'incohérences

Le tableau B.1 contient une liste extensible de résolutions classées en fonction du type de changement ; certaines résolutions présentes dans ce catalogue permettent une mise en oeuvre automatique. Pour le reste, HMCS-Collab recherche les résolutions appropriées dans le catalogue de résolutions, et les propose en support à la prise de décision. La colonne *Resolution* résume les résolutions possibles, tandis que la colonne *Resolution Type* précise la nature de chaque résolution (i.e., *automatique* versus *supervisée*).

Level	Type	Change Subtype	Resolution	Resolution Type
Metamodel	Global	Delete MM1	(**RM1** or **RM2**) and **R1**, with : **RM1** : Remove meta-correspondences (MM1) **R1** : Remove correspondences(M1) **RM2** : Adjust meta-correspondances (MM1)	Automatic Automatic Supervised
		Add MM1	**RM2** or **RM3** **RM3** : Define meta-correspondences (MM1)	Supervised
		Delete + add MM1	**RM1** or **RM2** or **RM3**	Supervised
Model	Global	Delete M1	**R1** : Remove correspondences(M1)	Automatic
		Add M1	**RM4.0** : Propagate meta-correspondences (MM1)	Automatic
		Delete + add M1	**R1** and **RM4.0**	Automatic
Model	Partial	Create element e	**RM4.1** : Propagate meta-correspondences (MM, me)	Automatic
		Delete element e	**R2** or **R3** or **R4** or **R5**, with : **R2** : Remove correspondences(e) if their cardinality = 2 **R3** : Remove correspondence's extremity if it still holds **R4** : Adjust the other models **R5** : Restore e	Supervised
		Rename element e / Move e / Pull up e / Push down e	**R2** or **R6**, with : **R2** : Remove correspondences(e) if their cardinality = 2 **R6** : Maintain correspondences(e)	Supervised
		Inline class c to d / Flatten class c to d	**R2** or (**R2** and **R7**), with : **R2** : Remove correspondences(c) **R7** : Force correspondences(d)	Supervised

TABLEAU B.1 – Catalogue de résolutions des incohérences

Annexe C

Compléments sur l'outil HMCS-Collab

C.1 Support des DSLs de CAHM

L'outil HMCS-Collab supporte les DSLs de CAHM, définis grâce aux méta-modèles MMC et MM-Collab. Dans cette section, nous présentons l'intégration de ces méta-modèles dans le projet côté serveur de HMCS-Collab. Les méta-modèles de l'approche CAHM présentés aux chapitres 4 et 5, ont été implémentés à l'aide de l'éditeur arborescent de Ecore. La Figure C.1 illustre le méta-modèle MMC et présente la console *interactive OCL* qui permet de définir en OCL des règles de bonne formation et de les valider au niveau M1. La contrainte évaluée sur la Figure vérifie que la nature de l'élément concerné par un changement (self.concernedElt) concorde avec le type du changement, i.e., un RefModel pour un GlobalChange et un RefElement pour un PartialChange

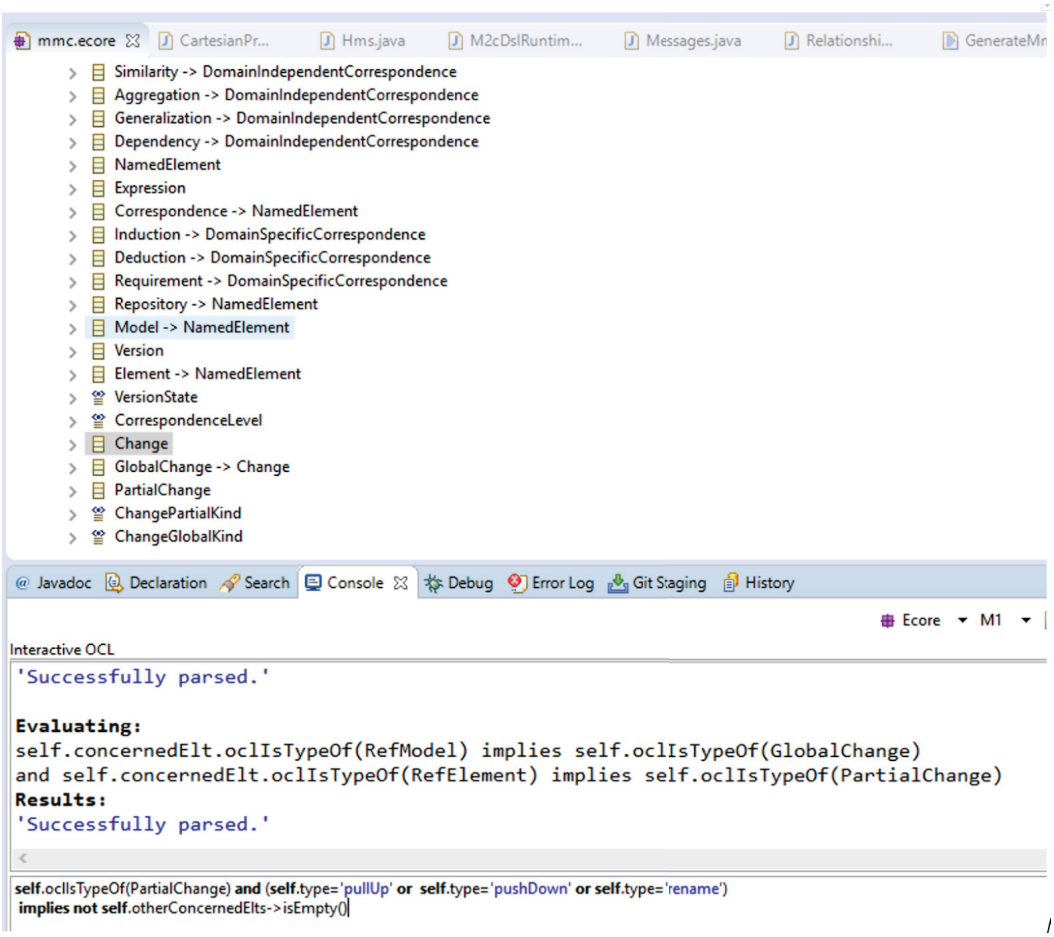
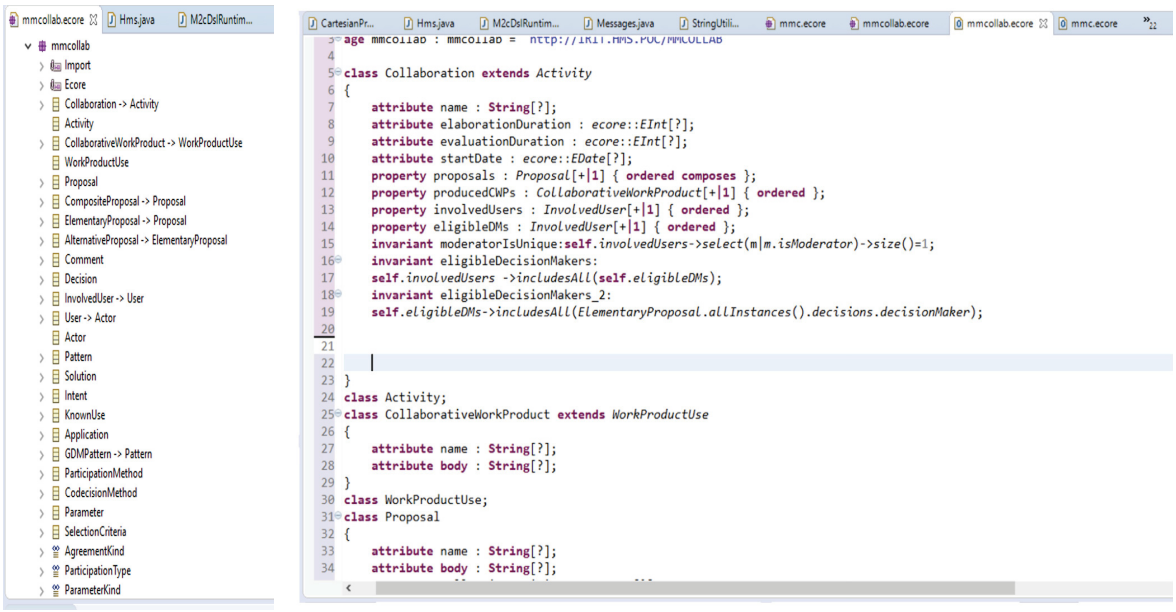


FIGURE C.1 – Utilisation de l'éditeur de modèle Ecore pour la conception du méta-modèle MMC et l'expression des contraintes OCL.

La Figure C.2 illustre le méta-modèle MMCollab saisi dans l'éditeur arborescent Ecore (a) et présente la définition de ce même méta-modèle dans l'éditeur OCLInEcore qui permet d'associer des règles OCL aux modèles Ecore.



(a) Utilisation de l'éditeur Ecore

(b) Utilisation de l'éditeur OCLInEcore pour la définition des règles OCL

FIGURE C.2 – Utilisation de l'éditeur de modèle Ecore pour la conception de MMCollab et l'expression des contraintes OCL dans l'éditeur OCLInEcore.

Une fois les méta-modèles intégrés, l'usage de l'outil EMF nous a permis de générer les classes et interfaces Java des concepts, afin qu'ils soient manipulables.

C.2 Sémantiques des relations associées aux correspondances

Dans cette section, nous détaillons les expressions sémantiques proposées pour les relations utilisées dans ce manuscrit. Nous résumons d'abord à l'aide d'un schéma *Mindmap* les bases de connaissances utilisées pour exprimer chaque relation sémantique. Ensuite nous détaillons les fonctions implémentées pour faciliter l'exploitation de ces bases de connaissances dans les sections C.2.2 et C.2.3. Dans le reste de cette annexe, nous exposons les sémantiques des relations utilisées dans ce mémoire, à savoir : similarité (Section C.2.4), agrégation (Section C.2.5), généralisation (Section C.2.6), induction (Section C.2.7), déduction (Section C.2.8).

C.2.1 Mindmap des bases de connaissances utilisées et des relations sémantiques définies

La Figure C.3 résume les fonctions que nous avons définies et implémentées pour mettre en oeuvre chacune des relations sémantiques utilisées dans ce mémoire, et précise, pour chaque relation, les relations internes des bases de connaissances wordNet [PEDERSEN et al., 2004] et ConceptNet [LIU et SINGH, 2004] exploitées.

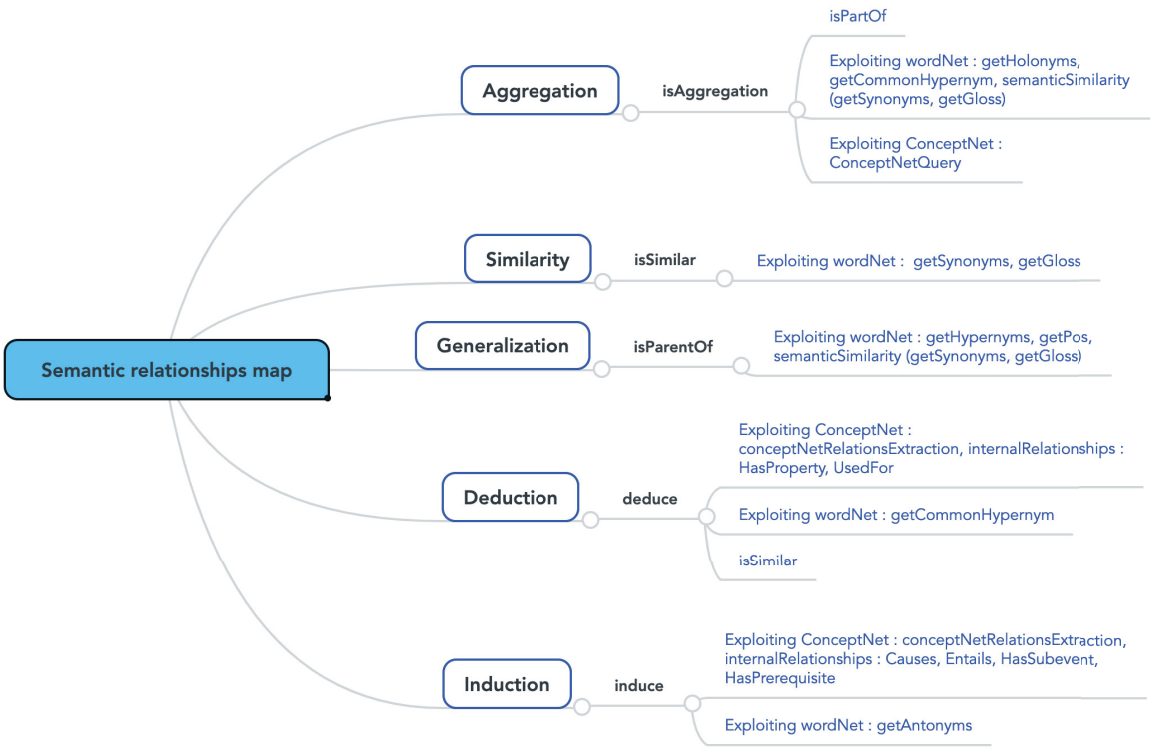


FIGURE C.3 – Mindmap des relations sémantiques et des fonctions qui les mettent en oeuvre.

C.2.2 Exploitation de WordNet

La relation principale entre les mots dans WordNet [PEDERSEN et al., 2004] est la synonymie, comme entre les mots *voiture* et *automobile*. Les synonymes - mots qui dénotent le même concept et qui sont interchangeables dans de nombreux contextes - sont regroupés en ensembles non ordonnés (synsets). Chacun des 117 000 synsets de WordNet est lié à d'autres synsets au moyen d'un petit nombre de «relations conceptuelles». De plus, un synset contient une brève définition («gloss») et, dans la plupart des cas, une ou plusieurs phrases courtes illustrant l'utilisation des membres du synset.

Synonymes. La fonction *getSynonyms* permet de récupérer la liste des synonymes d'un terme.

```
public List<String> getSynonyms(String term) throws JWNLException {
      IndexWord iw=dictionary.lookupIndexWord(getPos(term), term);
      ArrayList<String> synonym=new ArrayList<>();
      if (iw != null) {
         Synset[] senses = iw.getSenses();

            for (int i=0;i<senses.length;i++) {
             PointerTargetTree tree = PointerUtils.getInstance().
   getSynonymTree(senses[i], 1);

         for (Iterator itr = tree.toList().iterator(); itr.hasNext
   (); ) {
         if (itr.hasNext()) {
             for (Object o : ((PointerTargetNodeList) itr.next()))
   {
                 Synset t = ((PointerTargetNode) o).getSynset();
                 for(Word w : t.getWords()) {
```

```
                        synonym.add(w.getLemma().replace("_"," ").
   toUpperCase());
                    }
                  }
                }
              }
            }

        }
      return synonym;
    }

```

Listing C.1 – Fonction getSynonyms

Hypernymes. La fonction *getHypernyms* permet de récupérer la liste des hypernymes d'un terme. Un hypernyme est un mot au sens large auquel appartiennent des mots plus spécifiques. Y est un hypernyme de X si chaque X est un (genre de) Y; par exemple, la couleur est un hypernyme de rouge.

```
public List<String> getHypernyms(String term) throws JWNLException
   {
      IndexWord iw=dictionary.lookupIndexWord(getPos(term), term);
      ArrayList<String> hypernym=new ArrayList<>();
      if (iw != null) {
        for (    Synset synset : iw.getSenses()) {
            // PointerTargetNodeList tree = PointerUtils.
   getInstance().getDirectHypernyms(synset);
           PointerTargetTree tree = PointerUtils.getInstance().
   getHypernymTree(synset);
          List<PointerTargetNodeList> listTree=  tree.toList();
          for (int j=0; j < listTree.size(); j++) {
            PointerTargetNodeList nodes=listTree.get(j);
            for (int k=0; k < nodes.size(); k++) {
              PointerTargetNode node=(PointerTargetNode)nodes.get(
   k);
                  for(Word w : node.getSynset().getWords()) {
                      hypernym.add(w.getLemma().replace("_"," ").
   toUpperCase());
                  }
                }
              }

          }
      }
      return hypernym;
    }

```

Listing C.2 – Fonction getHypernyms

Holonymes. La fonction *getHolonyms* permet de récupérer la liste des holonymes d'un terme. L'holonymie est une relation partitive hiérarchisée : M1 est un holonyme de M2 si son signifié comprend le signifié de M2. Par exemple, corps est un holonyme de bras,

```
public List<String> getHolonyms(String term) throws JWNLException {
    // init();
     IndexWord iw=dictionary.lookupIndexWord(getPos(term), term);
      List<String> holonyms = new LinkedList<String>();
     if (iw != null){
     Synset[] senses = iw.getSenses();

        for (int i=0;i<senses.length;i++) {
     //The tree of all ancestors of the first meaning of given word
    that can introduce a holonym
     //we look at all possible parents and all parents of all
   synsets
         PointerTargetNodeList tree = PointerUtils.getInstance().
   getHolonyms(senses[i]);

   for (int j=0; j < tree.size(); j++) {
     PointerTargetNode node=(PointerTargetNode)tree.get(j);
            for(Word w : node.getSynset().getWords()) {
                holonyms.add(w.getLemma().replace("_"," ").
   toUpperCase());
            }
          }
        }

     }
    return holonyms;
 }
```

Listing C.3 – Fonction getHolonyms

Antonymes. La fonction *getAntonyms* permet de récupérer la liste des antonymes d'un terme. *Montée* est un antonyme de *Descente* par exemple.

```
public List<String> getAntonyms(String term) throws JWNLException {
     //init();
      IndexWord iw=dictionary.lookupIndexWord(getPos(term), term);
      ArrayList<String> antonym=new ArrayList<>();
      if (iw != null) {
         Synset[] senses = iw.getSenses();

            for (int i=0;i<senses.length;i++) {
             PointerTargetNodeList tree = PointerUtils.getInstance
   ().getAntonyms(senses[i]);
      for (int j=0; j < tree.size(); j++) {
                  PointerTargetNode node=(PointerTargetNode)tree.
   get(j);
               for(Word w : node.getSynset().getWords()) {
                   System.out.println(w.getLemma().replace("_"," "
   ).toUpperCase());
```

```
                    antonym.add(w.getLemma().replace("_"," ").
   toUpperCase());
                }
            }
         }


        }
      return antonym;
    }
```

Listing C.4 – Fonction getAntonyms

Position. La fonction *getPos* permet de préciser le réseau auquel appartient un terme. WordNet se compose de quatre réseaux : noms, verbes, adjectifs et adverbes ; sachant que la majorité des relations de WordNet connectent des mots de la même partie du discours (POS pour *Part Of Speech*).

```
public POS getPos(String term) throws JWNLException{
      IndexWord iw=dictionary.lookupIndexWord(POS.NOUN, term);
      if ( iw != null )
      return POS.NOUN;
      iw=dictionary.lookupIndexWord(POS.VERB, term);
      if ( iw != null )
      return POS.VERB;
      iw=dictionary.lookupIndexWord(POS.ADJECTIVE, term);
     if ( iw != null )
     return POS.ADJECTIVE;
     iw=dictionary.lookupIndexWord(POS.ADVERB, term);
     if ( iw != null )
     return POS.ADVERB;
 }
```

Listing C.5 – Fonction getPos

Gloss. permet d'attribuer une brève définition à un synset.

```
public List<String> getGloss(String term) throws JWNLException {
      IndexWord iw=dictionary.lookupIndexWord(getPos(term), term);
      ArrayList<String> gloss=new ArrayList<>();
      if (iw != null) {
         Synset[] senses = iw.getSenses();

            for (int i=0;i<senses.length;i++) {
                gloss.add(senses[i].getGloss().toUpperCase());

            }
         }
        return gloss;
      }
```

Listing C.6 – Fonction getGloss

C.2.3 Exploitation de ConceptNet

La base de connaissances ConceptNet [LIU et SINGH, 2004] est un réseau sémantique disponible en deux versions : concise (200 000 assertions) et complète (1,6 million d'assertions). Les connaissances communes de ConceptNet englobent les aspects spatiaux, physiques, sociaux, temporels et psychologiques de la vie quotidienne. Alors que des bases de connaissances sémantiques à grande échelle similaires comme Cyc et WordNet sont soigneusement fabriquées à la main, ConceptNet est généré automatiquement à partir des 700000 phrases du projet Open Mind Common Sense - une collaboration basée sur le World Wide Web avec plus de 14000 auteurs.
La fonction *conceptNetRelationsExtraction* permet d'extraire les relations existantes entre deux termes, tandis que la fonction *ConceptNetQuery* vérifie l'existence d'une relation donnée entre deux termes.

Fonction conceptNetRelationsExtraction.

```
public Map<String,Double> conceptNetRelationsExtraction (String
   term1, String term2){
    Map <String,Double> relationMap= new HashMap();
    try {
      // url containing the words to be lookedup

      String obj = " http://api.conceptnet.io/query?node=/c/en/"+
   term1+"&other=/c/en/"
      +term2;
      System.out.println(obj);
      // open HttpURLConnection
      HttpURLConnection hp = (HttpURLConnection) new URL(obj)
              .openConnection();
      // set to request method to get
      // not required since default
      hp.setRequestMethod("GET");
      // get the inputstream in the json format
      hp.setRequestProperty("Accept", "application/json");
      // get inputstream from httpurlconnection
      InputStream is = hp.getInputStream();
      // get text from inputstream using IOUtils
      String jsonText = IOUtils.toString(is);
      // get json object from the json String
      JSONObject json = new JSONObject(jsonText);
      // get the edges array from the JSONObject which contains all
      // content
      JSONArray edges = json.getJSONArray("edges");
      // goes through the edges array
      for (int x = 0; x < edges.length(); x++) {
          // convert the first object of the json array into a
   jsonobject
          // once it is a jsonobject you can use getString or
   getJSONArray
          // to continue in getting info
        JSONObject jsonRelation = new JSONObject(edges.
   getJSONObject(x).get("rel").toString());

        relationMap.put(jsonRelation.get("label").toString(),
   Double.parseDouble( edges.getJSONObject(x).get("weight").
```

```
   toString()));

35       }
         is.close();
37      }
     catch (Exception e) {
39       e.printStackTrace();
     }
41   return relationMap;

43  }
```

Listing C.7 – Fonction conceptNetRelationsExtraction

Fonction ConceptNetQuery.

```
1  public boolean ConceptNetQuery(String relation,String term1,String
   term2)    {
             final String CONCEPTNET_URI = " http://api.conceptnet.
   io/query?rel=/r/"+relation+"&node=/c/en/"+term1+"&other=/c/en/"
   +term2;
3            final String NBR_TO_RETRIEVE = "100";

5            // Strings identifying properties in the JSON string.
             final String EDGES = "edges";
7            System.out.println(CONCEPTNET_URI);

9                try {
                     // open HttpURLConnection
11                 HttpURLConnection hp = (HttpURLConnection) new
   URL(CONCEPTNET_URI)
                             .openConnection();
13                 // set to request method to get
                   // not required since default
15                 hp.setRequestMethod("GET");
                   // get the inputstream in the json format
17                 hp.setRequestProperty("Accept", "application/json
   ");
                   // get inputstream from httpurlconnection
19                 InputStream is = hp.getInputStream();
                   // get text from inputstream using IOUtils
21                 String jsonText = IOUtils.toString(is);
                   // get json object from the json String
23                 JSONObject json = new JSONObject(jsonText);
                   JSONArray edges = json.getJSONArray(EDGES);
25
                         if (edges.length()>0) return true;
27               } catch (IOException e) {
                     System.out.println("IOException: Can't retrieve
    message for: " + term1+ " , "+term2+" , "+relation);
29               } catch (JSONException e) {
                     System.out.println("JSONException: Can't
   retrieve message for: " + term1+ " , "+term2+" , "+relation);
```

```
                }
                return false;
            }
```

Listing C.8 – Fonction ConceptNetQuery

C.2.4 Expression sémantique de la relation « Similarité »

Fonction isSimilar. permet de détecter si deux termes sont similaires en se basant sur (1) une ontologie du domaine, (2) la relation « synonymie » de WordNet et (3) des fonctions de similarité basées sur le chemin, les termes ou l'Information Content (IC) (méthode de WuWalmer [Wu et Palmer, 1994] pour le chemin, méthode de Jaccard [Ivchenko et Honov, 1998] pour les termes et méthode de Lin pour l'IC [Lin et al., 2013]).

```
public boolean isSimilar (String source, String target) throws
   JWNLException {

     //1. background ontology

     if(listOwlClassName.contains( source)||listOwlClassName.
   contains(target)) {

         for (String className:listOwlClassName){
         if(getSynonyms(className).contains(source)||getSynonyms(
   className).contains(target))        return true;
         }
        }

     //2.1 wordnet uncomposed concepts
        if ( getSynonyms(source).contains(target) || getSynonyms(
   target).contains(source) ) return true;

         //2.2 wordnet compounded terms remove common words
        if (source.contains(" ") || target.contains(" ")){
            if  (!CommonWords.removeCommonWords(source, target).
   equals("xxx")){
            String result=CommonWords.removeCommonWords(source,
   target);
            source=result.substring(0,result.lastIndexOf('+'));
            target=result.substring(result.lastIndexOf('+')+1,
   result.length());

                if( getSynonyms(source).contains(target) ||
   getSynonyms(target).contains(source) ||getGloss(source).toString
   ().contains(target)
                        || getGloss(target).toString().contains(
   source)) return true;

            }
        }

        else{

         //3. lexical similarity after removing common words
```

```
            double score=0;

            ILexicalDatabase db = new NictWordNet();
            //Apply several methods and get the max similarity
  measures
            RelatednessCalculator rcLin = new Lin(db); //IC BASED
            RelatednessCalculator rcWup = new WuPalmer(db); //path
  based
            JaccardSimilarity jaccard=new JaccardSimilarity(); //
  term based
            score=Math.max( rcLin.calcRelatednessOfWords(source,
  target),jaccard.getSimilarity(source, target));
            score= Math.max(score,rcWup.calcRelatednessOfWords(
  source, target));
             System.out.println(score);
                if (score>=0.9) return true;
                else {TagLink tgLink = new TagLink();
                score=Math.max(score, tgLink.getSimilarity(source,
  target));
                                System.out.println(score);

                                   if (score>=0.8) return true;

                }
            }
    return false;
}
```

Listing C.9 – Fonction isSimilar

Fonction semanticSimilarity. permet de détecter si deux termes sont similaires en se basant sur (1) une ontologie du domaine, (2) la relation « synonymie » de WordNet.

```
public boolean semanticSimilarity(String source,String target)
  throws OWLOntologyCreationException, JWNLException{
    if(listOwlClassName.contains( source)||listOwlClassName.
  contains(target)) {

        for (String className:listOwlClassName){
        if(getSynonyms(className).contains(source)||getSynonyms(
  className).contains(target))        return true;
        }
       }
    else{
        // if the data is not found in the ontology or if there is
   no ontology to be imported, use the wordnet dictionary to get
  the synonyms
            // comparison
            if ( getSynonyms(source).contains(target) ||
  getSynonyms(target).contains(source)) return true;
```

```
            else if (CommonWords.extractCommonWords(target) != "xxx
    " && CommonWords.extractCommonWords(source) != "xxx"){
                if ( getGloss(source).contains(target) || getGloss(
    target).contains(source)) return true;

            }
            else if (!CommonWords.removeCommonWords(source, target)
    .equals("xxx")){
                String result=CommonWords.removeCommonWords(source,
     target);
                System.out.println("result  " +result);
                source=result.substring(0,result.lastIndexOf('+'));
                target=result.substring(result.lastIndexOf('+')+1,
    result.length());

            if( getSynonyms(source).contains(target) || getSynonyms
    (target).contains(source)) return true;
            }
      }
     return false;
  }
```

Listing C.10 – Fonction semanticSimilarity

C.2.5 Expression sémantique de la relation « Agrégation »

Fonction isAggregation. permet de détecter si une relation d'agrégation existe entre deux termes en se basant sur (1) la relation interne *isA* de ConceptNet et (2) la relation « homonymie » de WordNet.

```
public boolean isAggregation (List<String> part, List<String> whole
    ) throws FileNotFoundException, JWNLException,
    OWLOntologyCreationException
     {
        boolean flag=false;
        if(part.size()>1){
            Iterator<String> litr = part.iterator();

            flag=true;
            while ((litr.hasNext()) && flag==true){
                String s=litr.next().toString();
                if(!s.equals(whole.get(0))){
                                if (!ConceptNetQuery("IsA",s,whole.
    get(0)) ){
                                        if (!ConceptNetQuery("
    RelatedTo",s,whole.get(0)) ) flag=false;
                                }
                            }
                else flag=false;
            }
        }
        else {
        flag=isPartOf(part.get(0),whole.get(0));
        }
```

```
            return flag;
    }
```

Listing C.11 – Fonction isAggregation

Fonction isPartOf. permet de détecter si une relation d'agrégation existe entre deux termes en se basant sur la relation « homonymie » de WordNet.

```
public boolean isPartOf (String part, String whole) throws
   FileNotFoundException, JWNLException,
   OWLOntologyCreationException
    {
        // Get all the holonyms (children) of each sense of <var>
   word</var>
        List<String> holonyms= getHolonyms(part);
        if (holonyms.contains(whole)) return true;

        if (getCommonHypernym(part, whole) && semanticSimilarity(
   part, whole)) return true;
            return false;
    }
```

Listing C.12 – Fonction isPartOf

C.2.6 Expression sémantique de la relation « Généralisation »

Fonction isParentOf. permet de détecter si une relation de généralisation existe entre deux termes ou deux collections de termes en se basant sur la relation « hypernymie » de WordNet.

```
    public boolean isParentOf (String child, String parent) throws
   FileNotFoundException, JWNLException,
   OWLOntologyCreationException, URISyntaxException,
   AlignmentException
    {
        if (child.equals(parent)) return false;
        //add for the domain ontology the test abt the is child
   subclass of parent
        // Get all the hypernyms of child
        List<String> hypernyms= getHypernyms(child);
        if (hypernyms.contains(parent)) return true;

        String[] str=strUtil.splitSentence(child);
        if(str.length>1){
        for (String elt:str){
            if ( semanticSimilarity(elt,parent) ) return true;
            else if ( elt.equals(parent) && getPos(parent).equals(
   POS.NOUN)&& str.length>2) return true;
         }
        }
        return false;
    }

public boolean isParentOf (Collection<String> listchild, Collection
   <String> listparent) throws FileNotFoundException,
```

```
   OWLOntologyCreationException, JWNLException, URISyntaxException,
    AlignmentException {

        for (String child:listchild ){
            for (String parent:listparent){
                if (!isParentOf(child,parent)) return false;
        }
            return true;
        }
        return false;
    }
```

Listing C.13 – Fonction isParentOf

C.2.7 Expression sémantique de la relation « Induction »

Fonction induce. permet de détecter si une relation d'induction existe entre deux termes en se basant sur les relations internes de ConceptNet *causes*, *entails*, *hasPrerequisite*, et *hasSubevent*.

```
public boolean induce (String source, String cible ) throws
   JWNLException, OWLOntologyCreationException{
 if (source.contains(" ") || cible.contains(" ")){
     String [] splitSource=strUtil.splitSentence(source);
     String [] splitCible=strUtil.splitSentence(cible);
     List<String> listSource =new LinkedList<String>(Arrays.asList(
   splitSource));
     List<String> listCible = new LinkedList<String>(Arrays.asList(
   splitCible));
     List<String> toRemove = new ArrayList<String>();


     for (String s: listSource ){

            if(listCible.contains(s)){
                toRemove.add(s);
                        }
     }

     if (toRemove.size()>0){
         listCible.removeAll(toRemove);
         listSource.removeAll(toRemove);

         if (listSource.size()==0 || listCible.size()==0) return
   true;

         for (String s1: listSource ){
             for (String s2: listCible ){
                 if (getAntonyms(s1).contains(s2)) return false;
             }
         }
         return true;

     }
```

```
 else
        {
        for (String s1: listSource ){
         for (String s2: listCible ){
            Map <String,Double> relationMap= new HashMap();
            relationMap=conceptNetRelationsExtraction( s1.
   toLowerCase(), s2.toLowerCase());
            if (relationMap.containsKey("Causes")  ||relationMap.
   containsKey("Entails")  || relationMap.containsKey("HasSubevent"
   ) || relationMap.containsKey("HasPrerequisite")) return true;


           }
         }

        }

  }
 else{
     Map <String,Double> relationMap= new HashMap();
        relationMap=conceptNetRelationsExtraction( source.
   toLowerCase(), cible.toLowerCase());

        if (relationMap.containsKey("RelatedTo")  && !relationMap.
   containsKey("Antonym") ) return true;


   }
return false;
 }
```

Listing C.14 – Fonction induce

C.2.8 Expression sémantique de la relation « Déduction »

Fonction deduce. permet de détecter si une relation de déduction existe entre deux termes en se basant sur les relations internes de ConceptNet *usedFor* et *hasProperty*.

```
public boolean deduce (List<String> source, List<String> cible )
   throws OWLException, JWNLException{

        Iterator<String> litr = source.iterator();

           boolean flag=true;
           while ((litr.hasNext()) && flag==true){
               String s=litr.next().toString();

                Map <String,Double> relationMap= new HashMap();
                if(!s.equals(cible.get(0))){
                relationMap=conceptNetRelationsExtraction(s, cible
   .get(0));
           if (!getCommonHypernym(s, cible.get(0)) && !relationMap
   .containsKey("HasProperty")  && !relationMap.containsKey("
   UsedFor") ) flag=false;
```

```
14                  else {
                        if (getCommonHypernym(s, cible.get(0))){
16                        if (isSimilarLarge(s, cible.get(0)) || s.
  contains("_") || cible.get(0).contains("_")) flag=false;

18                      }
                    }
20                  }
                    else {
22                      if (source.size()==1) flag=false;
                    }
24              }
    return flag;
26

}
```

Listing C.15 – Fonction deduce

Bibliographie

AJMI, FATEN, OTHMAN, SARAH, BIAU, HAYFA et HAMMADI, SLIM. 2018, «Patient pathway workflow model identifying overcrowding indicators in emergency department», dans *Proceedings of 8th International Conference on Simulation and Modeling Methodologies*, Technologies and Applications. *Cité page 146*

AJMI, INES. 2015, *Outils et modèles collaboratifs pour la gestion des tensions dans les services des urgences pédiatriques*, thèse de doctorat, Ecole centrale de Lille. URL https://tel.archives-ouvertes.fr/tel-01277772/file/Ajmi_Ines_DLE.pdf. *Cité page 153*

AJMI, INES, ZGAYA, HAYFA, GAMMOUDI, LOTFI, HAMMADI, SLIM, MARTINOT, ALAIN, BEUSCART, RÉGIS et RENARD, JEAN-MARIE. 2015, «Mapping patient path in the pediatric emergency department : A workflow model driven approach», *Journal of biomedical informatics*, vol. 54, p. 315–328, Elsevier. *Cité page 146*

ALDAG, RAMON J et FULLER, SALLY R. 1993, «Beyond fiasco : A reappraisal of the groupthink phenomenon and a new model of group decision processes», *Psychological Bulletin*, vol. 113, nº 3, doi :10.1037/0033-2909.113.3.533, p. 533, American Psychological Association. *Cité page 35*

ANWAR, ADIL, EBERSOLD, SOPHIE, COULETTE, BERNARD, NASSAR, MAHMOUD et KRIOUILE, ABDELAZIZ. 2010, «A rule-driven approach for composing viewpoint-oriented models.», *Journal of Object Technology*, vol. 9, nº 2, p. 89–114, ETH Swiss Federal Institute of Technology. URL http://www.jot.fm/issues/issue_2010_03/article1/. *Cité page 5*

ATKINSON, COLIN et KÜHNE, THOMAS. 2008, «Reducing accidental complexity in domain models», *Software & Systems Modeling*, vol. 7, nº 3, doi :10.1007/s10270-007-0061-0, p. 345–359, Springer, ISSN 1619-1374. URL https://doi.org/10.1007/s10270-007-0061-0. *Cité page 5*

ATKINSON, COLIN, TUNJIC, CHRISTIAN, STOLL, DIETMAR et ROBIN, JACQUES. 2013, «A prototype implementation of an orthographic software modeling environment», dans *Proceedings of the 1st Workshop on View-Based, Aspect-Oriented and Orthographic Software Modelling*, VAO '13, ACM, New York, NY, USA, ISBN 9781450320702, p. 1–10, doi :10.1145/2489861.2489862. URL https://doi.org/10.1145/2489861.2489862. *Cité page 16*

BAKER, DENNIS, BRIDGES, DONALD, HUNTER, REGINA, JOHNSON, GREGORY, KRUPA, JOSEPH, MURPHY, JAMES et SORENSON, KEN. 2002, «Guidebook to decision-making methods», cahier de recherche, WSRC-IM-2002-00002, Department of Energy, USA. *Cité page 35*

BELTON, VALERIE et PICTET, JACQUES. 1997, «A framework for group decision using a mcda model : sharing, aggregating or comparing individual information ?», *Journal of decision systems*, vol. 6, nº 3, doi :10.1080/12460125.1997.10511726, p. 283–303, Taylor & Francis. URL https://doi.org/10.1080/12460125.1997.10511726. *Cité page 35*

BÉZIVIN, JEAN, BOUZITOUNA, SALIM, DEL FABRO, MARCOS DIDONET, GERVAIS, MARIE-PIERRE, JOUAULT, FRÉDÉRIC, KOLOVOS, DIMITRIOS, KURTEV, IVAN et PAIGE, RICHARD F. 2006, «A canonical scheme for model composition», dans *European Conference on Model Driven Architecture-*

Foundations and Applications, édité par A. Rensink et J. Warmer, Springer, ISBN 978-3-540-35910-4, p. 346–360, doi :10.1007/11787044_26. *Cité page 11*

BÉZIVIN, JEAN et KURTEV, IVAN. 2005, «Model-based technology integration with the technical space concept», dans *Proceedings of the Metainformatics Symposium*, vol. 20, Springer-Verlag, p. 44–49. URL https://hal.archives-ouvertes.fr/hal-00483587. *Cité page 13*

BONJEAN, NOELIE, MEFTEH, WAFA, GLEIZES, MARIE PIERRE, MAUREL, CHRISTINE et MIGEON, FRÉDÉRIC. 2014, «Adelfe 2.0», dans *Handbook on Agent-Oriented Design Processes*, édité par M. Cossentino, V. Hilaire, A. Molesini et V. Seidita, Springer, Berlin, Heidelberg, ISBN 978-3-642-39975-6, p. 19–63, doi :10.1007/978-3-642-39975-6_3. URL https://doi.org/10.1007/978-3-642-39975-6_3. *Cité page 152*

BOULANGER, FRÉDÉRIC, JACQUET, CHRISTOPHE, HARDEBOLLE, CÉCILE et DOGUI, AYMAN. 2013, «Heterogeneous model composition in ModHel'X : The power window case study», dans *Workshop on the Globalization of Modeling Languages at MODELS 2013*. *Cité page 174*

BRICHAU, JOHAN et D'HONDT, THEO. 2005, «Introduction to aspect-oriented software development», *European Network of Excellence on Aspect-Oriented Software Development, August 2005*, Springer. *Cité page 5*

BROY, MANFRED, FEILKAS, MARTIN, HERRMANNSDOERFER, MARKUS, MERENDA, STEFANO et RATIU, DANIEL. 2010, «Seamless model-based development : From isolated tools to integrated model engineering environments», *Proceedings of the IEEE*, vol. 98, nº 4, doi :10.1109/JPROC.2009.2037771, p. 526–545, IEEE, ISSN 1558-2256. *Cité page 5*

BRUNELIERE, HUGO, BURGER, ERIK, CABOT, JORDI et WIMMER, MANUEL. 2019, «A feature-based survey of model view approaches», *Software & Systems Modeling*, vol. 18, nº 3, doi :10.1007/s10270-017-0622-9, p. 1931–1952, Springer, ISSN 1619-1374. URL https://doi.org/10.1007/s10270-017-0622-9. *2 citations pages 13 et 14*

BRUNELIERE, HUGO, DE KERCHOVE, FLORENT MARCHAND, DANIEL, GWENDAL et CABOT, JORDI. 2018, «Towards scalable model views on heterogeneous model resources», dans *Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, MODELS '18, ACM, ISBN 9781450349499, p. 334–344, doi :10.1145/3239372.3239408. URL https://doi.org/10.1145/3239372.3239408. *2 citations pages 21 et 29*

BRUNELIERE, HUGO, PEREZ, JOKIN GARCIA, WIMMER, MANUEL et CABOT, JORDI. 2015, «EMF views : A view mechanism for integrating heterogeneous models», dans *International Conference on Conceptual Modeling*, édité par P. Johannesson, M. L. Lee, S. W. Liddle, A. L. Opdahl et Ó. Pastor López, Springer, ISBN 978-3-319-25264-3, p. 317–325, doi :10.1007/978-3-319-25264-3_23. *4 citations pages 17, 20, 21 et 29*

BRYANT, BARRETT, JÉZÉQUEL, JEAN-MARC, LÄMMEL, RALF, MERNIK, MARJAN, SCHINDLER, MARTIN, STEINMANN, FRIEDRICH, TOLVANEN, JUHA-PEKKA, VALLECILLO, ANTONIO et VÖLTER, MARKUS. 2015, «Globalized Domain Specific Language Engineering», dans *Globalizing Domain-Specific Languages : International Dagstuhl Seminar, Dagstuhl Castle, Germany, October 5-10, 2014, Revised Papers*, édité par B. Combemale, B. H. Cheng, R. B. France, J.-M. Jézéquel et B. Rumpe, Springer International Publishing, Cham, ISBN 978-3-319-26172-0, p. 43–69, doi :10.1007/978-3-319-26172-0_4. URL https://doi.org/10.1007/978-3-319-26172-0_4. *Cité page 29*

BUCCHIARONE, ANTONIO, CABOT, JORDI, PAIGE, RICHARD F et PIERANTONIO, ALFONSO. 2020, «Grand challenges in model-driven engineering : an analysis of the state of the research», *Software and Systems Modeling*, vol. 19, nº 1, doi :10.1007/s10270-019-00773-6, p. 5–13, ISSN 1619-1374. URL https://doi.org/10.1007/s10270-019-00773-6. *Cité page 6*

CALEFATO, FABIO, LANUBILE, FILIPPO et SCALAS, MARIO. 2007, «Porting a distributed meeting system to the Eclipse Communication Framework», dans *Proceedings of the 2007 OOPSLA workshop on eclipse technology eXchange*, eclipse '07, ACM, New York, NY, USA, ISBN 9781605580159, p. 46–49, doi :10.1145/1328279.1328289. URL https://doi.org/10.1145/1328279.1328289. *Cité page 140*

CARRASCOSA, IVÁN PALOMARES. 2018, «Group decision making and consensual processes», dans *Large Group Decision Making : Creating Decision Support Approaches at Scale*, Springer, ISBN 978-3-030-01027-0, p. 5–36, doi :10.1007/978-3-030-01027-0_2. URL https://doi.org/10.1007/978-3-030-01027-0_2. *Cité page 34*

CHAI, JUNYI. 2013, *Multicriteria decision analysis for structural decision problems*, thèse de doctorat, The Hong Kong Polytechnic University. *4 citations pages 37, 41, 42 et 47*

CHAI, JUNYI et LIU, JAMES NK. 2010, «An ontology-driven framework for supporting complex decision process», dans *2010 World Automation Congress*, IEEE, p. 1–6. *3 citations pages 37, 41 et 47*

CHAI, JUNYI, LIU, JAMES NK et XU, ZESHUI. 2012, «A new rule-based SIR approach to supplier selection under intuitionistic fuzzy environments», *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 20, nº 03, doi :10.1142/S0218488512500237, p. 451–471, World Scientific. URL https://doi.org/10.1142/S0218488512500237. *Cité page 35*

CICCHETTI, ANTONIO, CICCOZZI, FEDERICO et PIERANTONIO, ALFONSO. 2019, «Multi-view approaches for software and system modelling : a systematic literature review», *Software & Systems Modeling*, vol. 18, nº 6, doi :10.1007/s10270-018-00713-w, p. 3207–3233, Springer, ISSN 1619-1374. URL https://doi.org/10.1007/s10270-018-00713-w. *2 citations pages 14 et 15*

CICCHETTI, ANTONIO, DI RUSCIO, DAVIDE, ERAMO, ROMINA et PIERANTONIO, ALFONSO. 2008a, «Automating co-evolution in model-driven engineering», dans *2008 12th International IEEE Enterprise Distributed Object Computing Conference*, EDOC '08, IEEE, ISBN 9780769533735, p. 222–231, doi :10.1109/EDOC.2008.44. URL https://doi.org/10.1109/EDOC.2008.44. *Cité page 29*

CICCHETTI, ANTONIO, DI RUSCIO, DAVIDE, ERAMO, ROMINA et PIERANTONIO, ALFONSO. 2008b, «Meta-model differences for supporting model co-evolution», dans *Proceedings of the 2nd Workshop on Model-Driven Software Evolution-MODSE*, vol. 1. *Cité page 103*

CICCOZZI, FEDERICO, FAMELIS, MICHALIS, KAPPEL, GERTI, LAMBERS, LEEN, MOSSER, SEBASTIEN, PAIGE, RICHARD F, PIERANTONIO, ALFONSO, RENSINK, AREND, SALAY, RICK, TAENTZER, GABI, VALLECILLO, ANTONIO et WIMMER, MANUEL. 2018a, «Towards a body of knowledge for model-based software engineering», dans *Proceedings of the 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems : Companion Proceedings*, p. 82–89. URL https://hal.archives-ouvertes.fr/hal-01886114. *Cité page 5*

CICCOZZI, FEDERICO, FAMELIS, MICHALIS, KAPPEL, GERTI, LAMBERS, LEEN, MOSSER, SEBASTIEN, PAIGE, RICHARD F, PIERANTONIO, ALFONSO, RENSINK, AREND, SALAY, RICK, TAENTZER, GABI, VALLECILLO ANTONIO et WIMMER, MANUEL. 2018b, «How do we teach modelling and model-driven engineering? a survey», dans *Proceedings of the 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems : Companion Proceedings*, ACM, p. 122–129. URL https://doi.org/10.1145/3270129. *Cité page 138*

CLASEN, CAUÊ, JOUAULT, FRÉDÉRIC et CABOT, JORDI. 2011, «VirtualEMF : a model virtualization tool», dans *Advances in Conceptual Modeling. Recent Developments and New Directions*, édité par O. De Troyer, C. Bauzer Medeiros, R. Billen, P. Hallot, A. Simitsis et H. Van Mingroot, Springer, ISBN 978-3-642-24574-9, p. 332–335, doi :10.1007/978-3-642-24574-9_43. *4 citations pages 13, 17, 18 et 29*

CLAVEL, MANUEL, DURÁN, FRANCISCO, EKER, STEVEN, LINCOLN, PATRICK, MARTÍ-OLIET, NARCISO, MESEGUER, JOSÉ et TALCOTT, CAROLYN. 2007, *All about maude-a high-performance logical framework : how to specify, program and verify systems in rewriting logic*, Springer-Verlag. *Cité page 31*

CONRADI, REIDAR et WESTFECHTEL, BERNHARD. 1998, «Version models for software configuration management», *ACM Computing Surveys (CSUR)*, vol. 30, nº 2, doi :10.1145/280277.280280, p. 232–282, ACM, ISSN 0360-0300. URL https://doi.org/10.1145/280277.280280. *Cité page 16*

DAKNOU, AMANI, ZGAYA, HAYFA, HAMMADI, SLIM, HUBERT, HERVE, MASTORAKIS, NE, POULOS, M, MLADENOV, V, BOJKOVIC, Z et SIMIAN, D. 2008, «Toward a Multi-Agent Model for the Care of Patients at The Emergency Department», dans *The 10th WSEAS International Conference on Mathematical Methods, Computational Techniques and Intelligent Systems (MAMECTIS '08)*, vol. 10, WSEAS, Corfu, Greece, p. 264–269. URL https://hal.archives-ouvertes.fr/hal-00802536. *Cité page 146*

DALKEY, NORMAN C. 1969, «The Delphi method : An experimental study of group opinion», cahier de recherche, RAND CORP SANTA MONICA CALIF. *Cité page IV*

DANIEL, GWENDAL, SUNYÉ, GERSON, BENELALLAM, AMINE, TISI, MASSIMO, VERNAGEAU, YOANN, GÓMEZ, ABEL et CABOT, JORDI. 2017, «NeoEMF : A Multi-database Model Persistence Framework for Very Large Models», *Science of Computer Programming*, vol. 149, doi :10.1016/j.scico.2017.08.002, p. 9–14, Elsevier, ISSN 0167-6423. URL http://www.sciencedirect.com/science/article/pii/S0167642317301600, special Issue on MODELS'16. *Cité page 21*

DE VREEDE, GERT JAN et BRIGGS, ROBERT O. 2005, «Collaboration engineering : designing repeatable processes for high-value collaborative tasks», dans *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, IEEE, ISSN 1530-1605, p. 17c–17c, doi : 10.1109/HICSS.2005.144. *Cité page 33*

DEL FABRO, MARCOS DIDONET, BÉZIVIN, JEAN et VALDURIEZ, PATRICK. 2006, «Weaving models with the eclipse amw plugin», dans *Eclipse Modeling Symposium, Eclipse Summit Europe*, vol. 2006, p. 37–44. *Cité page 18*

DEL FABRO, MARCOS DIDONET et VALDURIEZ, PATRICK. 2009, «Towards the efficient development of model transformations using model weaving and matching transformations», *Software & Systems Modeling*, vol. 8, nº 3, doi :10.1007/s10270-008-0094-z, p. 305–324, Springer, ISSN 1619-1374. URL https://doi.org/10.1007/s10270-008-0094-z. *Cité page 13*

DELBECQ, ANDRE L et VAN DE VEN, ANDREW H. 1971, «A group process model for problem identification and program planning», *The Journal of Applied Behavioral Science*, vol. 7, nº 4, doi : 10.1177/002188637100700404, p. 466–492, Sage Publications Sage CA : Thousand Oaks, CA. *Cité page III*

DI RUSCIO, DAVIDE, FRANZAGO, MIRCO, MALAVOLTA, IVANO et MUCCINI, HENRY. 2017, «Envisioning the future of collaborative model-driven software engineering», dans *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, IEEE, p. 219–221, doi :10.1109/ICSE-C.2017.143. *Cité page 5*

DIJKSTRA, EDSGER WYBE. 1976, *A discipline of programming*, vol. 1, prentice-hall Englewood Cliffs. *Cité page 5*

DISKIN, ZINOVY, KÖNIG, HARALD et LAWFORD, MARK. 2019, «Multiple model synchronization with multiary delta lenses with amendment and K-Putput», *Formal Aspects of Computing*, vol. 31, nº 5, doi :10.1007/s00165-019-00493-0, p. 611–640, Springer, ISSN 1433-299X. URL https://doi.org/10.1007/s00165-019-00493-0. *4 citations pages 5, 6, 14 et 16*

EHRIG, MARC. 2006, *Ontology alignment : bridging the semantic gap*, vol. 4, Springer Science & Business Media. *Cité page 11*

EKLUND, ULRIK et ARTS, THOMAS. 2010, «A classification of value for software architecture decisions», dans *European Conference on Software Architecture*, édité par M. A. Babar et I. Gorton, Springer, ISBN 978-3-642-15114-9, p. 368–375, doi :10.1007/978-3-642-15114-9_30. *Cité page 47*

EL HAMLAOUI, MAHMOUD. 2015, *Mise en correspondance et gestion de la cohérence de modèles hétérogènes évolutifs*, thèse de doctorat, Université Toulouse Jean Jaurès en cotutelle avec l'Université Mohammed V de Rabat. *5 citations pages 6, 19, 94, 110 et 113*

EL HAMLAOUI, MAHMOUD, BENNANI, SALOUA, EBERSOLD, SOPHIE, NASSAR, MAHMOUD et COULETTE, BERNARD. 2019, «AHM : Handling Heterogeneous Models Matching and Consistency via MDE», dans *Evaluation of Novel Approaches to Software Engineering*, édité par E. Damiani, G. Spanoudakis et L. A. Maciaszek, Springer International Publishing, Cham, ISBN 978-3-030-22559-9, p. 288–313, doi :10.1007/978-3-030-22559-9_13. *3 citations pages 104, 129 et 134*

EL HAMLAOUI, MAHMOUD, BENNANI, SALOUA, NASSAR, MAHMOUD, EBERSOLD, SOPHIE et COULETTE, BERNARD. 2018, «A MDE Approach for Heterogeneous Models Consistency», dans *ENASE 2018 - Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering*, vol. 2018-March, p. 180–191, doi :10.5220/0006774101800191. *5 citations pages 6, 17, 18, 29 et 79*

EL HAMLAOUI, MAHMOUD, COULETTE, BERNARD, EBERSOLD, SOPHIE, BENNANI, SALOUA, NASSAR, MAHMOUD, ANWAR, ADIL, BEUGNARD, ANTOINE, BACH, JEAN-CHRISTOPHE, JAMOUSSI, YASSINE et TRAN, HANH NHI. 2016, «Alignment of viewpoint heterogeneous design models : Emergency department case study», dans *4th International Workshop On the Globalization of Modeling Languages (GEMOC 2016) co-located with ACM/IEEE MODELS 2016*, Saint-Malo, France, p. 18–27. URL https://hal.archives-ouvertes.fr/hal-01436169. *Cité page 146*

EL HAMLAOUI, MAHMOUD, EBERSOLD, SOPHIE, COULETTE, BERNARD, NASSAR, MAHMOUD et ANWAR, ADIL. 2014, «Heterogeneous models matching for consistency management», dans *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*, IEEE, ISSN 2151-1357, p. 1–12, doi :10.1109/RCIS.2014.6861074. *8 citations pages 6, 17, 18, 29, 79, 104, 129 et 134*

ELLIS, CLARENCE A, GIBBS, SIMON J et REIN, GAIL. 1991, «Groupware : some issues and experiences», *Communications of the ACM*, vol. 34, n° 1, doi :10.1145/99977.99987, p. 39–58, ACM, ISSN 0001-0782. URL https://doi.org/10.1145/99977.99987. *3 citations pages 11, 17 et 33*

EUZENAT, JÉRÔME, MEILICKE, CHRISTIAN, STUCKENSCHMIDT, HEINER, SHVAIKO, PAVEL et TROJAHN, CÁSSIA. 2011, «Ontology alignment evaluation initiative : six years of experience», dans *Journal on data semantics XV*, édité par S. Spaccapietra, Springer, p. 158–192, doi :10.1007/978-3-642-22630-4_6. URL https://doi.org/10.1007/978-3-642-22630-4_6. *Cité page 11*

EYAL SALMAN, HAMZEH, HAMMAD, MUSTAFA, SERIAI, ABDELHAK-DJAMEL et AL-SBOU, AHED. 2018, «Semantic clustering of functional requirements using agglomerative hierarchical clustering», *Information*, vol. 9, n° 9, doi :10.3390/info9090222, p. 222, Multidisciplinary Digital Publishing Institute. *Cité page 134*

FAUCHER, CYRIL, BERTRAND, FRÉDÉRIC et LAFAYE, JEAN-YVES. 2008, «Génération d'ontologie à partir d'un modèle métier UML annoté», *Revue des Nouvelles Technologies de l'Information*, vol. 12, p. 65–84, Hermann. URL https://hal.inria.fr/inria-00460298. *Cité page 25*

FELDMANN, STEFAN, KERNSCHMIDT, KONSTANTIN, WIMMER, MANUEL et VOGEL-HEUSER, BIRGIT. 2019, «Managing inter-model inconsistencies in model-based systems engineering : Application in automated production systems engineering», *Journal of Systems and Software*, vol. 153, doi :10.1016/j.jss.2019.03.060, p. 105 – 134, Elsevier, ISSN 0164-1212. URL http://www.sciencedirect.com/science/article/pii/S0164121219300639.
2 citations pages 15 et 16

FOSTER, J. NATHAN, GREENWALD, MICHAEL B., MOORE, JONATHAN T., PIERCE, BENJAMIN C. et SCHMITT, ALAN. 2007, «Combinators for bidirectional tree transformations : A linguistic approach to the view-update problem», *ACM Trans. Program. Lang. Syst.*, vol. 29, nº 3, doi :10.1145/1232420.1232424, p. 17–es, ACM, New York, NY, USA, ISSN 0164-0925. URL https://doi.org/10.1145/1232420.1232424.
Cité page 16

FRANCE, ROBERT et RUMPE, BERNHARD. 2007, «Model-driven development of complex software : A research roadmap», dans *Future of Software Engineering (FOSE '07)*, IEEE Computer Society, ISBN 0-7695-2829-5, p. 37–54, doi :10.1109/FOSE.2007.14.
Cité page 5

FRANZAGO, MIRCO, DI RUSCIO, DAVIDE, MALAVOLTA, IVANO et MUCCINI, HENRY. 2017, «Collaborative model-driven software engineering : a classification framework and a research map», *IEEE Transactions on Software Engineering*, vol. 44, nº 12, doi :10.1109/TSE.2017.2755039, p. 1146–1175, IEEE, ISSN 2326-3881.
2 citations pages 6 et 15

FUKS, HUGO, RAPOSO, ALBERTO B, GEROSA, MARCO A et LUCENA, CARLOS JP. 2005, «Applying the 3c model to groupware development», *International Journal of Cooperative Information Systems*, vol. 14, nº 02n03, p. 299–328, World Scientific. URL https://doi.org/10.1142/S0218843005001171.
2 citations pages 33 et 34

GALLARDO, JESÚS, MOLINA, ANA I, BRAVO, CRESCENCIO et REDONDO, MIGUEL A. 2013, «A model-driven and task-oriented method for the development of collaborative systems», *Journal of Network and Computer Applications*, vol. 36, nº 6, p. 1551–1565, Elsevier. URL https://doi.org/10.1016/j.jnca.2013.03.016.
Cité page 33

GALSTER, MATTHIAS. 2011, «Dependencies, traceability and consistency in software architecture : towards a view-based perspective», dans *Proceedings of the 5th European Conference on Software Architecture : Companion Volume*, ECSA '11, ACM, ISBN 9781450306188, p. 1–4, doi :10.1145/2031759.2031761. URL https://doi.org/10.1145/2031759.2031761.
Cité page 13

GALVAO, ISMENIA et GOKNIL, ARDA. 2007, «Survey of Traceability Approaches in Model-Driven Engineering», dans *11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007)*, IEEE, ISSN 1541-7719, p. 313–324, doi :10.1109/EDOC.2007.42.
2 citations pages 6 et 14

GAMMA, ERICH. 1995, *Design patterns : elements of reusable object-oriented software*, Pearson Education India.
3 citations pages 68, 78 et 132

GIESE, HOLGER, HILDEBRANDT, STEPHAN et NEUMANN, STEFAN. 2010, «Model Synchronization at Work : Keeping SysML and AUTOSAR Models Consistent», dans *Graph Transformations and Model-Driven Engineering : Essays Dedicated to Manfred Nagl on the Occasion of his 65th Birthday*, édité par G. Engels, C. Lewerentz, W. Schäfer, A. Schürr et B. Westfechtel, Springer, ISBN 978-3-642-17322-6, p. 555–579, doi :10.1007/978-3-642-17322-6_24. URL https://doi.org/10.1007/978-3-642-17322-6_24.
3 citations pages 6, 14 et 16

GOLRA, FAHAD R, BEUGNARD, ANTOINE, DAGNAT, FABIEN, GUERIN, SYLVAIN et GUYCHARD, CHRISTOPHE. 2016, «Addressing Modularity for Heterogeneous Multi-model Systems using Model Federation», dans *Companion Proceedings of the 15th International Conference on Modularity*, ACM, ACM, ISBN 9781450340335, p. 206–211, doi :10.1145/2892664.2892701. URL https://doi.org/10.1145/2892664.2892701.
5 citations pages 13, 17, 22, 23 et 29

GRUSCHKO, BORIS, KOLOVOS, DIMITRIOS et PAIGE, RICHARD. 2007, «Towards Synchronizing Models with Evolving Metamodels», dans *Proceedings of the International Workshop on Model-Driven Software Evolution*, Amsterdam, The Netherlands, p. 3. *Cité page 103*

GUYCHARD, CHRISTOPHE, GUERIN, SYLVAIN, KOUDRI, ALI, BEUGNARD, ANTOINE et DAGNAT, FABIEN. 2013, «Conceptual interoperability through models federation», dans *Semantic Information Federation Community Workshop*. *4 citations pages 14, 17, 22 et 29*

HARDEBOLLE, CÉCILE et BOULANGER, FRÉDÉRIC. 2007, «ModHel'X : A component-oriented approach to multi-formalism modeling», dans *International Conference on Model Driven Engineering Languages and Systems*, Springer, p. 247–258. *Cité page 174*

HARRIS, ROBERT. 1998, «Introduction to decision making», VirtualSalt. URL http://www.virtualsalt.com/crebook5.htm. *2 citations pages 33 et 35*

HASSAN, ADEL et OUSSALAH, MOURAD CHABANE. 2018, «Evolution styles : Multi-view/multi-level model for software architecture evolution.», *JSW*, vol. 13, nº 3, doi :10.17706/jsw.13.3.146-154, p. 146–154. *Cité page 13*

HEBIG, REGINA, KHELLADI, DJAMEL EDDINE et BENDRAOU, REDA. 2015, «Surveying the corpus of model resolution strategies for metamodel evolution», dans *2015 Asia-Pacific Software Engineering Conference (APSEC)*, IEEE, ISSN 1530-1362, p. 135–142, doi :10.1109/APSEC.2015.40. *Cité page 112*

HEIN, CHRISTIAN, RITTER, TOM et WAGNER, MICHAEL. 2010, «Model-Driven Tool Integration with ModelBus», dans *Proceedings of 1st International Workshop on Future Trends of Model-Driven Development*, p. 35–39, doi :10.5220/0002174800350039. *Cité page 16*

HERRMANNSDOERFER, MARKUS et KOEGEL, MAXIMILIAN. 2010, «Towards a generic operation recorder for model evolution», dans *Proceedings of the 1st International Workshop on Model Comparison in Practice*, IWMCP '10, ACM, ISBN 9781605589602, p. 76–81, doi :10.1145/1826147.1826161. URL https://doi.org/10.1145/1826147.1826161. *Cité page 16*

HILL, TERRY et WESTBROOK, ROY. 1997, «SWOT analysis : it's time for a product recall», *Long range planning*, vol. 30, nº 1, p. 46–52, Elsevier. *Cité page IV*

HWANG, CHING-LAI et YOON, KWANGSUN. 1981, «Multi-objective decision making–methods and application. A state-of-the-art study», New York : Springer-Verlag. *Cité page IV*

IVCHENKO, GI et HONOV, SA. 1998, «On the Jaccard similarity test», *Journal of Mathematical Sciences*, vol. 88, nº 6, p. 789–794, Springer. *Cité page XXI*

IZQUIERDO, JAVIER LUIS CÁNOVAS et CABOT, JORDI. 2016, «Collaboro : a collaborative (meta) modeling tool», *PeerJ Computer Science*, vol. 2, doi :10.7717/peerj-cs.84, p. e84, PeerJ Inc., ISSN 2376-5992. *4 citations pages 37, 45, 46 et 47*

JOUAULT, FRÉDÉRIC, ALLILAIRE, FREDDY, BÉZIVIN, JEAN, KURTEV, IVAN et VALDURIEZ, PATRICK. 2006, «ATL : A QVT-like Transformation Language», dans *Companion to the 21st ACM SIGPLAN Symposium on Object-Oriented Programming Systems, Languages, and Applications*, OOPSLA '06, Association for Computing Machinery, New York, NY, USA, ISBN 159593491X, p. 719–720, doi :10.1145/1176617.1176691. URL https://doi.org/10.1145/1176617.1176691. *Cité page 17*

JOUAULT, FRÉDÉRIC, VANHOOFF, BERT, BRUNELIERE, HUGO, DOUX, GUILLAUME, BERBERS, YOLANDE et BÉZIVIN, JEAN. 2010, «Inter-DSL coordination support by combining megamodeling and model weaving», dans *ACM 25th Symposium on Applied Computing (SAC 2010), Track "Coordination Models, Languages and Applications"*, ACM, p. 2011–2018, doi :10.1145/1774088.1774511. URL https://hal.archives-ouvertes.fr/hal-00534353. *Cité page 13*

KEDJI, KOMLAN AKPÉDJÉ, LBATH, REDOUANE, COULETTE, BERNARD, NASSAR, MAHMOUD, BARESSE, LAURENT et RACARU, FLORIN. 2012, «Supporting collaborative development using process models : An integration-focused approach», dans *Proceedings of the International Conference on Software and System Process*, ICSSP '12, IEEE Press, ISBN 9781467323529, p. 120–129. *2 citations pages 33 et 58*

KELLY, STEVEN. 2018, «Modelling by the people, for the people», dans *Software Technologies : Applications and Foundations*, édité par M. Seidl et S. Zschaler, Springer International Publishing, Cham, ISBN 978-3-319-74730-9, p. 178–183. *Cité page 6*

KESSI, KAHINA, OUSSALAH, MOURAD et ALIMAZIGHI, ZAIA. 2014, «Viewpoints for Requirement Engineering in a Cooperatif Information System (VpCIS)», dans *New Perspectives in Information Systems and Technologies, Volume 1*, édité par Á. Rocha, A. M. Correia, F. . B. Tan et K. . A. Stroetmann, Springer, Cham, ISBN 978-3-319-05951-8, p. 299–308, doi :10.1007/978-3-319-05951-8_29. *Cité page 13*

KHELLADI, DJAMEL EDDINE, BENDRAOU, REDA, HEBIG, REGINA et GERVAIS, MARIE-PIERRE. 2017, «A semi-automatic maintenance and co-evolution of OCL constraints with (meta)model evolution», *Journal of Systems and Software*, vol. 134, doi :10.1016/j.jss.2017.09.010, p. 242–260, Elsevier. URL https://hal.inria.fr/hal-02330211. *2 citations pages 112 et 117*

KHELLADI, DJAMEL EDDINE, KRETSCHMER, ROLAND et EGYED, ALEXANDER. 2018, «Change propagation-based and composition-based co-evolution of transformations with evolving metamodels», dans *Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, MODELS '18, ACM, ISBN 9781450349499, p. 404–414, doi :10.1145/3239372.3239380. URL https://doi.org/10.1145/3239372.3239380. *Cité page 29*

KILGOUR, D MARC et EDEN, COLIN. 2010, «Introduction to the handbook of group decision and negotiation», dans *Handbook of group decision and negotiation*, Springer, p. 1–7. *Cité page 35*

KÖNIG, HARALD et DISKIN, ZINOVY. 2016, «Advanced local checking of global consistency in heterogeneous multimodeling», dans *Modelling Foundations and Applications*, édité par A. Wąsowski et H. Lönn, Springer International Publishing, Cham, ISBN 978-3-319-42061-5, p. 19–35. *Cité page 5*

KORNYSHOVA, ELENA. 2011, *MADISE : Method Engineering-based Approach for Enhancing Decision-Making in Information Systems Engineering*, thèse de doctorat, Université Panthéon-Sorbonne - Paris I. *3 citations pages 37, 39 et 47*

KORNYSHOVA, ELENA et DENECKÈRE, RÉBECCA. 2010, «Decision-making ontology for information system engineering», dans *International Conference on Conceptual Modeling*, ER'10, Springer, ISBN 3642163726, p. 104–117. *4 citations pages 37, 39, 40 et 47*

KRAMER, MAX E. 2015, «A generative approach to change-driven consistency in multi-view modeling», dans *Proceedings of the 11th International ACM SIGSOFT Conference on Quality of Software Architectures*, ACM, p. 129–134, doi :10.1145/2737182.2737194. *2 citations pages 6 et 14*

KRAMER, MAX E., BURGER, ERIK et LANGHAMMER, MICHAEL. 2013, «View-centric engineering with synchronized heterogeneous models», dans *Proceedings of the 1st Workshop on View-Based, Aspect-Oriented and Orthographic Software Modelling*, VAO '13, ACM, New York, NY, USA, ISBN 9781450320702, p. 1–6, doi :10.1145/2489861.2489864. URL https://doi.org/10.1145/2489861.2489864. *Cité page 5*

LANGE, CHRISTIAN FJ et CHAUDRON, MICHEL RV. 2004, «An empirical assessment of completeness in UML designs», *IET Conference Proceedings*, doi :10.1049/ic:20040404, p. 111–119(8), Institution of Engineering and Technology. URL https://digital-library.theiet.org/content/conferences/10.1049/ic_20040404. *Cité page 15*

LARSEN, MATIAS EZEQUIEL VARA, DEANTONI, JULIEN, COMBEMALE, BENOIT et MALLET, FRÉDÉRIC. 2015, «A Behavioral Coordination Operator Language BCOoL», dans *2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, ACM, p. 186–195, doi :10.1109/MODELS.2015.7338249. URL https://hal.inria.fr/hal-01182773. *Cité page 174*

LE MOIGNE, JEAN-LOUIS. 1999, «La modélisation des systèmes complexes», *Paris, Dunod, réed.* *Cité page 5*

LIN, YUNG-SHEN, JIANG, JUNG-YI et LEE, SHIE-JUE. 2013, «A Similarity Measure for Text Classification and Clustering», *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, nº 7, doi :10.1109/TKDE.2013.19, p. 1575–1590, IEEE. *Cité page XXI*

LIU, HUGO et SINGH, PUSH. 2004, «Conceptnet - a practical commonsense reasoning tool-kit», *BT technology journal*, vol. 22, nº 4, p. 211–226, Springer. *2 citations pages XIV et XIX*

MALAVOLTA, IVANO, MUCCINI, HENRY et REKHA, SMRITHI. 2014, «Enhancing architecture design decisions evolution with group decision making principles», dans *International Workshop on Software Engineering for Resilient Systems*, édité par I. Majzik et M. Vieira, Springer, ISBN 978-3-319-12241-0, p. 9–23, doi :10.1007/978-3-319-12241-0_2. *8 citations pages v, 36, 37, 43, 44, 47, 49 et III*

MANSOOR, USMAN, KESSENTINI, MAROUANE, WIMMER, MANUEL et DEB, KALYANMOY. 2017, «Multi-view refactoring of class and activity diagrams using a multi-objective evolutionary algorithm», *Software Quality Journal*, vol. 25, nº 2, doi :10.1007/s11219-015-9284-4, p. 473–501, ISSN 1573-1367. URL https://doi.org/10.1007/s11219-015-9284-4. *Cité page 16*

MARCA, DAVID A et MCGOWAN, CLEMENT L. 1987, *SADT : structured analysis and design technique*, McGraw-Hill, Inc. *Cité page 99*

MCGUINNESS, DEBORAH L, VAN HARMELEN, FRANK et al.. 2004, «OWL web ontology language overview», *W3C recommendation*, vol. 10, nº 10, p. 2004. *Cité page 38*

MORENTE-MOLINERA, JUAN ANTONIO, WU, XING, MORFEQ, ALI, AL-HMOUZ, RAMI et HERRERA-VIEDMA, ENRIQUE. 2020, «A novel multi-criteria group decision-making method for heterogeneous and dynamic contexts using multi-granular fuzzy linguistic modelling and consensus measures», *Information Fusion*, vol. 53, doi :10.1016/j.inffus.2019.06.028, p. 240–250, Elsevier, ISSN 1566-2535. URL http://www.sciencedirect.com/science/article/pii/S1566253519300636. *2 citations pages 34 et 35*

MUSSBACHER, GUNTER, AMYOT, DANIEL, BREU, RUTH, BRUEL, JEAN-MICHEL, CHENG, BETTY H. C., COLLET, PHILIPPE, COMBEMALE, BENOIT, FRANCE, ROBERT B., HELDAL, ROGARDT, HILL, JAMES, KIENZLE, JÖRG, SCHÖTTLE, MATTHIAS, STEIMANN, FRIEDRICH, STIKKOLORUM, DAVE et WHITTLE, JON. 2014, «The relevance of model-driven engineering thirty years from now», dans *Model-Driven Engineering Languages and Systems*, édité par J. Dingel, W. Schulte, I. Ramos, S. Abrahão et E. Insfran, Springer International Publishing, Cham, ISBN 978-3-319-11653-2, p. 183–200, doi :10.1007/978-3-319-11653-2_12. *Cité page 13*

NASLAVSKY, LEILA, ALSPAUGH, THOMAS A, RICHARDSON, DEBRA J et ZIV, HADAR. 2005, «Using scenarios to support traceability», dans *Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering*, ACM, p. 25–30, doi :10.1145/1107656.1107663. *2 citations pages 6 et 14*

NASSAR, MAHMOUD. 2003, «VUML : a Viewpoint oriented UML Extension», dans *18th IEEE International Conference on Automated Software Engineering, 2003. Proceedings.*, IEEE, ISSN 1938-4300, p. 373–376, doi :10.1109/ASE.2003.1240341. *Cité page 5*

NEUMAYR, BERND, SCHREFL, MICHAEL et THALHEIM, BERNHARD. 2011, «Modeling techniques for multi-level abstraction», dans *The Evolution of Conceptual Modeling : From a Historical Perspective towards the Future of Conceptual Modeling*, édité par R. Kaschek et L. Delcambre, Springer, ISBN 978-3-642-17505-3, p. 68–92, doi :10.1007/978-3-642-17505-3_4. URL https://doi.org/10.1007/978-3-642-17505-3_4. *Cité page 5*

OBJECT MANAGEMENT GROUP (OMG). 2008, «Software & Systems Process Engineering Meta-Model Specification V2.0», cahier de recherche. URL http://www.omg.org/spec/SPEM/2.0. *2 citations pages 58 et 104*

OBJECT MANAGEMENT GROUP (OMG). 2017, «Unified Modeling Language Specification», cahier de recherche. URL https://www.omg.org/spec/UML/2.5.1/PDF. *Cité page 95*

OUSSALAH, MOURAD. 2018, «Evolution in complex objects», *ArXiv*, vol. abs/1802.07974. *Cité page 117*

PAIGE, RICHARD F, MATRAGKAS, NICHOLAS et ROSE, LOUIS M. 2016, «Evolving models in model-driven engineering : State-of-the-art and future challenges», *Journal of Systems and Software*, vol. 111, doi :10.1016/j.jss.2015.08.047, p. 272–280, Elsevier. *Cité page 103*

PEDERSEN, TED, PATWARDHAN, SIDDHARTH et MICHELIZZI, JASON. 2004, «WordNet : :Similarity - Measuring the Relatedness of Concepts», dans *Demonstration papers at HLT-NAACL 2004*, Association for Computational Linguistics, p. 38–41. *2 citations pages XIV et XV*

PFEIFFER, ROLF-HELGE et WĄSOWSKI, ANDRZEJ. 2015, «The design space of multi-language development environments», *Software & Systems Modeling*, vol. 14, nº 1, doi :10.1007/s10270-013-0376-y, p. 383–411, Springer, ISSN 1619-1374. URL https://doi.org/10.1007/s10270-013-0376-y. *2 citations pages 14 et 15*

REINEKE, JAN et TRIPAKIS, STAVROS. 2014, «Basic problems in multi-view modeling», dans *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer, ISBN 978-3-642-54862-8, p. 217–232. *Cité page 5*

REKHA, SMRITHI et MUCCINI, HENRY. 2014, «Suitability of software architecture decision making methods for group decisions», dans *European Conference on Software Architecture*, édité par P. Avgeriou et U. Zdun, Springer, ISBN 978-3-319-09970-5, p. 17–32, doi :10.1007/978-3-319-09970-5_2. *Cité page 36*

RIEBISCH, MATTHIAS. 2003, «Towards a More Precise Definition of Feature Models», *Modelling Variability for Object-Oriented Product Lines*, p. 64–76, Citeseer. *Cité page 14*

ROCKWELL, JUSTIN, GROSSE, IAN R, KRISHNAMURTY, SUNDAR et WILEDEN, JACK C. 2009, «A decision support ontology for collaborative decision making in engineering design», dans *2009 International Symposium on Collaborative Technologies and Systems*, CTS '09, IEEE, IEEE Computer Society, ISBN 9781424445844, p. 1–9, doi :10.1109/CTS.2009.5067456. URL https://doi.org/10.1109/CTS.2009.5067456. *3 citations pages 37, 38 et 47*

ROSE, LOUIS, ETIEN, ANNE, MENDEZ, DAVID, KOLOVOS, DIMITRIOS, PAIGE, RICHARD et POLACK, FIONA. 2010, «Comparing Model-Metamodel and Transformation-Metamodel Coevolution», dans *Model and Evolution Workshop*, Olso, Norway. URL https://hal.inria.fr/inria-00524314. *Cité page 29*

SAATY, THOMAS L. 1988, «What is the Analytic Hierarchy Process?», dans *Mathematical models for decision support*, édité par G. Mitra, H. J. Greenberg, F. A. Lootsma, M. J. Rijkaert et H. J. Zimmermann, Springer, ISBN 978-3-642-83555-1, p. 109–121, doi :10.1007/978-3-642-83555-1_5. *2 citations pages 36 et III*

SAATY, THOMAS L. 1994, «Highlights and critical points in the theory and application of the Analytic Hierarchy Process», *European journal of operational research*, vol. 74, nº 3, p. 426–447, Elsevier. *Cité page IV*

SAATY, THOMAS L. 2008, «Decision making with the analytic hierarchy process», *International journal of services sciences*, vol. 1, nº 1, doi :10.1504/IJSSCI.2008.017590, p. 83–98. *2 citations pages 36 et III*

SAATY, THOMAS L, VARGAS, LUIS G et al.. 2006, *Decision making with the analytic network process*, vol. 282, Springer. *Cité page 36*

SANTOS, RAPHAEL O, OLIVEIRA, FELIPE F, GOMES, ROBERTA L, MARTINELLO, MAGNOS et GUIZZARDI, RENATA S S. 2011, «Lightweight Collaborative Web Browsing», *International Journal of Web Portals (IJWP)*, vol. 3, nº 1, doi :10.4018/jwp.2011010102, p. 17–32, IGI Global, Hershey, PA, USA, ISSN 1938-0194. URL http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/jwp.2011010102. *Cité page 33*

SCHMIDT, ANDRAS. 2011, «EMFCollab», URL http://qgears.com/products/emfcollab/. *Cité page 139*

SCHMIDT, DOUGLAS C. 2006, «Model-driven engineering», *COMPUTER-IEEE COMPUTER SOCIETY-*, vol. 39, nº 2, p. 25–31, Citeseer. *Cité page 5*

SCHÜRR, ANDY. 1995, «Specification of graph translators with triple graph grammars», dans *Graph-Theoretic Concepts in Computer Science*, édité par E. W. Mayr, G. Schmidt et G. Tinhofer, Springer, Berlin, Heidelberg, ISBN 978-3-540-49183-5, p. 151–163, doi :10.1007/3-540-59071-4_45. *Cité page 16*

SELONEN, PETRI et KETTUNEN, MARKUS. 2007, «Metamodel-based inference of inter-model correspondence», dans *11th European Conference on Software Maintenance and Reengineering*, CSMR '07, IEEE, ISBN 0769528023, p. 71–80, doi :10.1109/CSMR.2007.31. URL https://doi.org/10.1109/CSMR.2007.31. *Cité page 11*

SHOSHA, RAMY, DEBRUYNE, CHRISTOPHE et O'SULLIVAN, DECLAN. 2015, «Towards an adaptive tool and method for collaborative ontology mapping», dans *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, édité par I. Ciuciu, H. Panetto, C. Debruyne, A. Aubry, P. Bollen, R. Valencia-García, A. Mishra, A. Fensel et F. Ferri, Springer, ISBN 978-3-319-26138-6, p. 319–328, doi :10.1007/978-3-319-26138-6_35. *6 citations pages v, 17, 25, 26, 27 et 29*

SILVER, BRUCE. 2009, *BPMN method and style*, Cody-Cassidy Press. *Cité page 58*

SNIEZEK, JANET A et BUCKLEY, TIMOTHY. 1995, «Cueing and cognitive conflict in judge-advisor decision making.», *Organizational behavior and human decision processes*, Elsevier Science. *Cité page 79*

SOLEY, RICHARD et al.. 2000, «Model driven architecture», *OMG white paper*, vol. 308, nº 308, p. 5. *Cité page 59*

ŠPERKA, ROMAN. 2011, «Multi-agent systems for business process management-overview», *Economics and Management*, p. 9. *Cité page 152*

STEINBERG, DAVE. 2010, «Eclipsecon 2007 : Effective use of the eclipse modeling framework», URL https://www.slideshare.net/dmsteinberg/eclipsecon-2007-effective-use-of-the-eclipse-modeling-framework. *Cité page 138*

STEPPER, EIKE. 2010, «CDO Model Repository Overview», URL https://www.eclipse.org/cdo/documentation/. *Cité page 140*

TROLLMANN, FRANK et ALBAYRAK, SAHIN. 2016, «Extending model synchronization results from triple graph grammars to multiple models», dans *Theory and Practice of Model Transformations*, édité par P. Van Gorp et G. Engels, Springer, Cham, ISBN 978-3-319-42064-6, p. 91–106, doi : 10.1007/978-3-319-42064-6_7. *Cité page 16*

TROLLMANN, FRANK, BLUMENDORF, MARCO, SCHWARTZE, VEIT et ALBAYRAK, SAHIN. 2011, «Formalizing model consistency based on the abstract syntax», dans *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems*, ACM, p. 79–84, doi :10.1145/1996461.1996498. *2 citations pages 6 et 14*

TÖRNGREN, MARTIN, QAMAR, AHSAN, BIEHL, MATTHIAS, LOIRET, FREDERIC et EL-KHOURY, JAD. 2014, «Integrating viewpoints in the development of mechatronic products», *Mechatronics*, vol. 24, nº 7, doi :10.1016/j.mechatronics.2013.11.013, p. 745 – 762, Elsevier, ISSN 0957-4158. URL http://www.sciencedirect.com/science/article/pii/S095741581300233X. *Cité page 16*

VALLECILLO, ANTONIO. 2010, «On the combination of domain specific modeling languages», dans *European Conference on Modelling Foundations and Applications*, Springer, p. 305–320, doi : 10.1007/978-3-642-13595-8_24. *Cité page 13*

VANHERPEN, KEN, DENIL, JOACHIM, DAVID, ISTVAN, DE MEULENAERE, PAUL, MOSTERMAN, PIETER J, TORNGREN, MARTIN, QAMAR, AHSAN et VANGHELUWE, HANS. 2016, «Ontological reasoning for consistency in the design of cyber-physical systems», dans *2016 1st International Workshop on Cyber-Physical Production Systems (CPPS)*, IEEE, ISBN 978-1-5090-1156-8, p. 1–8, doi :10.1109/CPPS.2016.7483922. *3 citations pages 17, 24 et 29*

VERMOLEN, SANDER D, WACHSMUTH, GUIDO et VISSER, EELCO. 2011, «Reconstructing Complex Metamodel Evolution», dans *Software Language Engineering*, édité par A. Sloane et U. Aßmann, Springer Berlin Heidelberg, Berlin, Heidelberg, ISBN 978-3-642-28830-2, p. 201–221, doi :10.1007/978-3-642-28830-2_11. *Cité page 31*

VIJAYARANI, S, ILAMATHI, MS J et NITHYA, MS. 2015, «Preprocessing techniques for text mining-an overview», *International Journal of Computer Science & Communication Networks*, vol. 5, nº 1, p. 7–16. *Cité page 134*

VON WINTERFELDT, DETLOF et FISCHER, GREGORY W. 1975, «Multi-attribute utility theory : models and assessment procedures», dans *Utility, probability, and human decision making*, Springer, p. 47–85. *Cité page V*

WIMMER, MANUEL, MORENO, NATHALIE et VALLECILLO, ANTONIO. 2012, «Viewpoint co-evolution through coarse-grained changes and coupled transformations», dans *Objects, Models, Components, Patterns*, édité par C. A. Furia et S. Nanz, Springer Berlin Heidelberg, Berlin, Heidelberg, ISBN 978-3-642-30561-0, p. 336–352, doi :10.1007/978-3-642-30561-0_23. *Cité page 31*

WIRTH, NIKLAUS. 1996, «Extended Backus-Naur Form», *Iso/Iec*, vol. 14977, nº 2996. *Cité page 139*

WOHLIN, CLAES, RUNESON, PER, HÖST, MARTIN, OHLSSON, MAGNUS C, REGNELL, BJÖRN et WESSLÉN, ANDERS. 2012, *Experimentation in Software Engineering*, Springer Science & Business Media. *Cité page 164*

WU, ZHIBIAO et PALMER, MARTHA. 1994, «Verbs Semantics and Lexical Selection», dans *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*, ACL '94, Association for Computational Linguistics, USA, p. 133–138, doi :10.3115/981732.981751. URL https://doi.org/10.3115/981732.981751. *Cité page XXI*

XUE, XINGSI et LIU, JIANHUA. 2017, «Collaborative ontology matching based on compact interactive evolutionary algorithm», *Knowledge-Based Systems*, vol. 137, doi :10.1016/j.knosys.2017.09.017,

p. 94–103, Elsevier, ISSN 0950-7051. URL https://doi.org/10.1016/j.knosys.2017.09.017. *4 citations pages 17, 27, 28 et 29*

YIE, ANDRÉS, CASALLAS, RUBBY, DERIDDER, DIRK et WAGELAAR, DENNIS. 2009, «Deriving Correspondence Relationships to Guide a Multi-view Heterogeneous Composition», dans *Proceedings of the 2009 International Conference on Models in Software Engineering*, MODELS'09, Springer-Verlag, Berlin, Heidelberg, p. 225–239, doi :10.1007/978-3-642-12261-3_22. URL https://doi.org/10.1007/978-3-642-12261-3_22. *Cité page 5*

Liste des acronymes

ADMIR Advanced Digital entreprise Modeling and Information Retrieval. 98, 147

AHM Alignment of Heterogeneous Models. 6, 17, 18, 19, 20, 29, 31, 94, 104, 172

AHP Analytic Hierarchy Process. 36, 45, III, IV

AMT Assisted Matching Tool. 134

AMW ATLAS Model Weaver. 18

API Application Programming Interface. 18, 20

ATL ATL Transformation Language. 17

BP Business Process. 95, 96, 98, 115, 146, 147, 154, 160, 162

BPMN Business Process Model and Notation. 58, 96, 149

CAHM Collaborative Alignment of Heterogeneous Models. 7, 8, 82, 94, 99, 120, 123, 144, 146, 147, 164, 165, 167, 172, 173, 174, XIII

CCT Consistency Checker Tool. 134, 135

CDO Connected Data Objects. 21, 138, 139

CDT Change Detector Tool. 134

CIMA Compact Interactive Memetic Algorithm. 17, 27, 29, 31

CMS Conference Management System. vi, 94, 95, 97, 98, 99, 102, 108, 115, 149, 164

CMSPEM Collaborative Model-Based Software & System Process Engineering Metamodel. 33, 58, 64, 75

CMT Consistency Management Tool. 130, 134, 136, 137

CollabT Collaboration Tool. 130, 132, 133, 135, 136, 137

CommT Communication Tool. 130, 133

DAT Decision Aggregator Tool. 130, 131, 132

DIR Domain Independant Relatioship. 19, 100, 101

DMO Decision Making Ontology. 39, 40, 41, 47, 49

DMP Decision Making Policies. 131, 132

DMT Decision Making Tool. 130, 131, 132, 135, 136, 137

DRT Decision-policies Repository Tool. 130, 131

DSL Domain Specific Language. 5, 6, 16, 20, 21, 37, 45, 165

DSO Decision Support Ontology. 37, 38, 39, 43, 47, 49

DSR Domain Specific Relatioship. 19, 82, 100, 101, 104, 105

EBNF Extended Backus-Naur Form. 139

ECF Eclipse Communication Framework. 137, 140

EMF Eclipse Modeling Framework. 20, 21, 22, 23, 137, 138, 139, 140, 141, XIV

EMP Eclipse Modeling Project. 137

ER Emergency Report. 146, 147, 151, 154, 159, 162

GAT Group Awareness Tool. 130, 133

GDM Group Decision Making. 33, 35, 36, 37, 38, 39, 42, 43, 44, 45, 47, 49, 50, 51, 55, 91, 120

GMT Group Management Tool. 130, 133

GPL Generic Purpose Language. 173

HLR High Level Relationship. 19

HMCS Heterogeneous Matching and Consistency-management Suite. 104, 129

HTTP Hypertext Transfer Protocol. 141

IDM Ingénierie Dirigée par les Modèles. 5, 6, 8, 11, 13, 31, 95, 105, 111, 138, 147, 164, 173

IHM Interfaces Homme-machine. 137, 165

IRC Internet Relay Chat. 140

IRT Inconsistency Resolver Tool. 134, 135

JAS Judge-Advisor Systems. 79

JET Java Emitter Templates. 135, 140

JSF JavaServer Faces. 141

JSON JavaScript Object Notation. 62, 64

JSP JavaServer Pages. 140

LC Local Coordinator. 98

LLR Low Level Relationship. 19

LTM Linguistic Type Model. 24

M1C Modèle de Correspondances. 100, 105, 134, 165

M2T Model-to-Text. 135

MADISE MAke Decisions in Information Systems Engineering. 37, 39, 41, 47, 49

MAS Multi Agent System. 147, 159, 162

MAUT Multi-Attribute Utility Theory. V

MCGDM Multi Criteria Group Decision Making. 35, 36

MMC Méta-Modèle de Correspondances. 19, 94, 100, 101, 104, 105, 107, 110, 112, 115, 120, 123, 154, 171, 173, XIII

MMCollab Méta-Modèle de Collaboration. 7, 174, XIII

MOF Meta Object Facility. 59, 165, 173

MT Matching Tool. 130, 134, 136, 137

MVC Modèle-vue-contrôleur. 140

Net4j Net4j. 139

OCL Object Constraint Language. 61, 139, XIII, XIV

OntoGDSS Ontology based Group Decision Support System. 37, 41, 43, 47, 49

ORM Object Relational Mapping. 141

OTM Ontological Type Models. 24

OWL Web Ontology Language. 23, 38

PS PersiStence. 95, 98, 102

QVT Query View Transform. 17

RPC Remote Procedure Call. 140
RT Refining Tool. 134

SADT Structured Analysis and Design Techniques. 99
SD Software Design. 94, 95, 97, 102, 115, 147, 154, 162
SGBD Système de Gestion de Base de Données. 141
SPEM Software & Systems Process Engineering Metamodel. 58, 61, 75, 104
SQL Structured Query Language. 20, 140
SU Service d'urgence. 144, 145, 148, 150, 159, 164
SWOT Strengths - Weaknesses - Opportunities - Threats. IV

T2M Text-to-Model. 135
TT Transformation Tool. 130, 134, 135, 136

UML Unified Modeling Language. 5, 16, 39, 95, 147

XMI XML Metadata Interchange. 138
XML eXtensible Markup Language. 22, 23, 62, 64, 140, 141
XMPP Extensible Messaging and Presence Protocol. 140
XSD XML Schema Definition. 23

SALOUA BENNANI

saloua.bennani91@gmail.com
https://www.linkedin.com/in/saloua-bennani/

FORMATION

Doctorat en informatique *2016 - présent*
Cotutelle entre Université Toulouse Jean Jaurès et Université Mohammed V de Rabat
Laboratoires : IRIT/Toulouse et ADMIR/Rabat
Intitulé : Une approche IDM pour l'alignement collaboratif de modèles hétérogènes.

Cycle d'ingénieur en Informatique *2011 - 2014*
Ecole Nationale Supérieure d'Informatique et d'Analyse des Systèmes - Rabat, Maroc

Classes préparatoires aux grandes écoles *2009 - 2011*
Option MPSI/MP, Centre Ibn Taimiya - Marrakech, Maroc

Baccalauréat Sciences Mathématiques *2009*
Lycée Abou Abass Essebti - Marrakech, Maroc

EXPÉRIENCE

Attaché Temporaire d'Enseignement et Recherche, Université Jean Jaurès *2019 - 2020*
Algorithmiques, Structures de données, Programmation orientée objet, PIX.

Consultant en Informatique, Capgemini, Casablanca, Maroc *2014 - 2016*
Projet : TMA évolutive et corrective sur le CRM SFR Entreprises

ENSEIGNEMENT

Attaché Temporaire d'Enseignement et Recherche, Université Jean Jaurès *2019 - 2020*
Programmation orientée objet : Licence 3 (17h TD)
Algorithmique et structures de données avancées en Python : Licence 2 (17h TD)
Algorithmique et programmation : Licence 1 (57h TD)
PIX : Licence 2 (72h TD)

Vacataire d'enseignement, I.U.T de Blagnac *Jan 2019 - Mars 2019*
Conception orientée objet : 1ère année I.U.T (15h TD)

Vacataire d'enseignement, Université Toulouse Jean Jaurès *2017 - 2019*
Algorithmique et structures de données avancées en Python : Licence 2 (42h TD)
Certificat Informatique et Internet - C2I : Licence 2 (72h TD)

ACTIVITÉS

Membre du comité d'organisation de STAF'18, ENSEEIHT, Toulouse, France
Organisation de la conférence Software Technologies : Applications and Foundations (STAF'18)

Relecteur de papiers scientifiques
Relecteur pour la conférence Evaluation of Novel Approaches to Software Engineering (ENASE'19)

Liste des publications

Revues internationales avec comité de lecture

1. **Saloua Bennani**, Iliass Ait El Kouch, Mahmoud El Hamlaoui, Sophie Ebersold, Bernard Coulette and Mahmoud Nassar. « A Formalization of Group Decision Making in Multi-Viewpoints Design ». Computer and Information Science (CIS). Canadian Center of Science and Education, 2020, vol. 13, n°1, pp. 58-71.
2. Mahmoud El Hamlaoui, **Saloua Bennani**, Sophie Ebersold, Mahmoud Nassar and Bernard Coulette. « AHM : Handling heterogeneous models matching and consistency via MDE ». Extended paper of ENASE 2018, Communications in Computer and Information Science (CCIS), Springer, 2018, Vol. 1023, pp. 288-313.
3. Mahmoud El Hamlaoui, Sophie Ebersold, **Saloua Bennani**, Adil Anwar, Taoufiq Dkaki, Mahmoud Nassar, Bernard Coulette. « A model-driven approach to align heterogeneous models of a complex system ». Journal of Object Technology (JOT). *En cours de révision.*

Conférences internationales avec actes et comité de programme

4. **Saloua Bennani**, Sophie Ebersold, Mahmoud El Hamlaoui, Bernard Coulette and Mahmoud Nassar. « A collaborative decision approach for alignment of heterogeneous models ». Proceedings of the 28th IEEE International Conference on Enabling Technologies : Infrastructure for Collaborative Enterprises (WETICE) 2019, pp. 112-117. IEEE Computer Society.
5. **Saloua Bennani**, Mahmoud El Hamlaoui, Mahmoud Nassar, Sophie Ebersold and Bernard Coulette. « Collaborative model-based matching of heterogeneous models ». Proceedings of the 22nd IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD) 2018, pp. 443-448. IEEE Xplore.
6. Mahmoud El Hamlaoui, **Saloua Bennani**, Mahmoud Nassar, Sophie Ebersold and Bernard Coulette. « A MDE Approach for Heterogeneous Models Consistency ». Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering - Volume 1 : ENASE 2018, pp. 180-191. SciTePress.
7. Mahmoud El Hamlaoui, **Saloua Bennani**, Mahmoud Nassar, Sophie Ebersold and Bernard Coulette. « Heterogeneous design models alignment : from matching to consistency management ». Proceedings of the 33rd ACM Annual Symposium on Applied Computing (SAC). Pau, 2018. pp. 1695-1697. ACM DL.
8. Mahmoud El Hamlaoui, Bernard Coulette, Sophie Ebersold, **Saloua Bennani**, Mahmoud Nassar, Adil Anwar, Antoine Beugnard, Yassine Jamoussi, Hanh-Nhi Tran. « Alignment of viewpoint heterogeneous design models : Emergency Department case study ». Proceedings of the 4th International Workshop On the Globalization of Modeling Languages (GEMOC), CEUR-WS, pp 18-27, co-located with ACM/IEEE MODELS 2016, Saint-Malo.

Poster dans une conférence internationale avec comité de programme

9. **Saloua Bennani**. «Towards a collaborative matching approach to relate sustainable cities design models ». EuroScience Open Forum (ESOF) 2018. Toulouse, France.