Extraction de modèles de processus métiers à partir de journaux d'évènements

THÈSE

présentée et soutenue publiquement le 12 décembre 2007

pour l'obtention du

Doctorat de l'Université des Sciences et Technologies de Lille

(spécialité informatique)

par

Nacim IHADDADENE

Composition du jury

Président :	Sophie TISON, Professeur	Université de Lille I
Rapporteurs :	Dan SIMOVICI, Professeur	Université du Massachusetts
	Mohand-Saïd HACID, Professeur	Université de Lyon 1
Directeur de thèse :	Chabane DJERABA, Professeur	Université de Lille I

UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE
Laboratoire d'Informatique Fondamentale de Lille — UPRESA 8022
U.F.R. d'I.E.E.A. – Bât. M3 – 59655 VILLENEUVE D'ASCQ CEDEX
Tél. : +33 (0)3 28 77 85 41 – Télécopie : +33 (0)3 28 77 85 37 – email : direction@lifl.fr

À mes parents...
À mon épouse et mon fils...
À mes frères et sœurs...
À toute ma famille...
À tous mes amis...

Remerciements

En premier lieu, j'aimerais remercier mon directeur de thèse, Mr Chabane DJERABA, professeur à l'université de Lille 1, de l'attention et du soutien qu'il a porté à mon travail de doctorant.

J'aimerais également remercier Mme Sophie TISON, professeur à l'université de Lille 1, qui m'a fait l'honneur d'exercer la fonction de président du jury.

Je voudrais aussi remercier M. Dan SIMOVICI et M. Mohand Saïd HACID, qui ont bien voulu accepter d'être les rapporteurs de ce travail.

Sans oublier tous les collègues du pôle "Recherche et Développement" de la société SNEDA, dont l'aide et le soutien m'ont été précieux au cours de mon travail, et notamment : M. Alain MORVANT, M. Emmanuël AUGARDE et M. Olivier DUFOURD.

Enfin, merci à toutes les personnes qui ont rendu possible l'achèvement de ce travail.

Résumé

Les approches actuelles de représentation des processus métiers s'appuient principalement sur les systèmes de workflows qui ordonnent les tâches selon une logique métier déterminée. Les systèmes de workflows, présents sur le marché, souffrent en revanche de limites sérieuses dans l'évolution et la normalisation des processus métiers.

En effet, les méthodes traditionnelles de conception de processus se basent sur la perception des concepteurs, certainement expérimentés, mais n'impliquent pas les observations issues des usages réels. De plus, elles souffrent de l'absence de procédures d'évolution permettant d'optimiser la structure du workflow initialement conçue. La conception est globalement statique, alors qu'elle devrait être dynamique. En effet, il n'y a aucun intérêt de garder un modèle de processus qui ne répond pas au besoin des usages.

Par ce travail, nous contribuons à l'évolution et à la normalisation des processus métier, à travers :

- L'exploitation des journaux d'événements contenant le minimum d'informations (identité de l'instance, identité de la tâche, instant de fin).

- L'enrichissement de la description du processus métier par des motifs de construction (diffusion (AND/XOR-Split) et jointure AND/OR-join)) et par des relations exceptionnelles (de continuité et de fin) entre les tâches.

- La prise en compte de procédures de normalisation en cas de défaillance des tâches, afin de fiabiliser au maximum l'exécution du processus.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction générale

1.1 Introduction

À partir des années 70, les progiciels MRP (*Manufacturing Resource Planning*) ont été développés pour mettre en application un concept de gestion élaboré à partir du début des années 60. Ce concept repose sur la planification, la programmation et le suivi de l'exécution des activités des industries d'assemblage. A contrario l'ERP (*Enterprise Resource Planning*) a été implémenté dans les années 90 dans des logiciels de gestion adossés à des bases de données relationnelles, sans que le concept de gestion sous-jacent ne soit véritablement explicité. A l'image des ERP traduit significativement en français par PGI (*Progiciel de Gestion Intégré*), la plupart des grandes solutions informatiques actuelles reposent sur la promesse d'un traitement intégré et synchronisé des données, c'est-à-dire sur des concepts issus du traitement de l'information plutôt que de la gestion des connaissances.

Les outils de travail collaboratif se sont développés pour répondre à ces préoccupations. Ils permettent en effet de travailler en mode asynchrone et de manière distante tout en assurant la structuration, le suivi ainsi que la traçabilité des échanges. On peut définir les principales fonctions d'un outil de travail collaboratif de la manière suivante : Ce sont des outils d'aide à la réalisation de taches communes, et d'aide à la planification et à la synchronisation du travail.

La technologie des Workflows a paru comme une plateforme appropriée pour la gestion

des ressources et des informations pour les entreprises, favorisant l'interopérabilité des outils hétérogènes et des systèmes multi-plateformes et offrant une vue intuitive et globale des modèles de processus métier.

1.2 Cadre général de la thèse

Les systèmes de Workflows permettent de définir, d'exécuter et de gérer les différents processus métiers d'une organisation. Cependant certains problèmes sont rencontrés lors du déploiement de cette technologie. Un des problèmes majeurs est le fait que de tels systèmes requièrent une phase d'étude et de modélisation par des personnes qui doivent avoir des connaissances approfondies sur le processus métier. La conception des processus métiers via un workflow est souvent complexe, car elle exige une maîtrise approfondie des besoins de l'application, elle est souvent statique et suppose des hypothèses à priori souvent articulées autour d'estimations plus ou moins précises des usages réels des processus. Ce qui est fort difficile. Il n'est donc par rare de développer un processus métier qui, sur la base des observations réelles, ne soit pas conforme aux véritables usages, et donc aux véritables besoins des utilisateurs.

Profitant de la capacité de la plupart des systèmes transactionnels à garder la trace des événements qui surviennent tout au long de la vie d'un processus, notre objectif est de permettre, par l'analyse et l'exploitation de ces traces, d'extraire un ensemble de connaissances et ainsi, d'aider les organisations à améliorer la qualité de leurs processus métiers et les services fournis à leurs partenaires.

Les progiciels actuels permettent la préservation des journaux d'événements survenant dans les systèmes d'information. Ces journaux sont rarement exploités pour améliorer les processus métiers qu'ils couvrent. Notre thèse contribue à l'extraction de modèles de processus métiers à partir des journaux d'événements afin de les intégrer dans les systèmes décisionnels et d'améliorer les exécutions futures des processus.

La conception basée sur l'expérience et les usages propose une approche complémentaire aux approches actuelles de conception. Cette complémentarité réduit le risque de subjectivité des modèles de processus. Elle réduit aussi le risque d'anomalies entre les modèles de processus initialement conçus et les processus des utilisateurs. En effet, les utilisateurs peuvent

s'écarter de la conception pré-spécifiée.

1.3 Un peu d'histoire

L'idée de développer des méthodes qui améliorent la conception de processus métiers sur la base de l'expérience et les usages est assez ancienne. Elle date de Hammer [Ham90] et d'autres qui sont parmi les premiers à décrire des approches d'évolution des processus métiers basée sur l'expérience et les usages. Cette approche est désignée par Hammer [Ham90] sous les termes de *Business Process Analysis* ou encore *Business Process Reconfiguration* (BPR). Si l'idée de base est ancienne, il n'en demeure pas moins que très peu d'études théoriques et formelles ont été mises en œuvre dans ce sens. C'est à dire que très peu d'études académiques ont concerné des procédures d'amélioration et de correction conceptuelle sur la base de l'expérience et des usages.

En revanche, cette idée a largement été adoptée par l'industrie. Une compagnie de services australienne a réalisé un sondage auprès de 107 compagnies australiennes et asiatiques et a rapporté que 50% étaient déjà intéressées ou projettent de l'être par la reconfiguration des processus. Dans une étude des compagnies américaines et européennes, 621 entreprises américaines et européennes avec des revenus de 500 millions de dollars au moins par an ont été auditionnées. Plus de 69% de ces entreprises avaient déjà adopté la reconfiguration des processus en tant que moyens d'améliorer leurs opérations commerciales. Aussi bien que 88% des entreprises américaines utilisaient la reconfiguration des processus ou étaient sur le point de lancer des projets de reconfiguration des processus. Une étude britannique semblable a rapporté un pourcentage de 59%. Une autre étude, qui a inclus 93 grandes et moyennes compagnies anglaises, a montré que 41% de ces compagnies ont conduit un ou plusieurs projets de reconfiguration des processus. Ceux-ci et beaucoup d'autres études semblent suggérer que la reconfiguration des processus soit plus populaire parmi de plus grandes compagnies. Même si ces chiffres sont anciennes, nous sommes persuadés que les compagnies sont plus importantes aujourd'hui. Par ailleurs, nous pouvons penser qu'à la différence des entreprises nord américaines, les entreprises dans d'autres régions du monde, comme l'Europe, sont plus frileux à l'idée d'adopter cette approche. La reconfiguration des processus sur la base des observations

des traces d'exécution s'imposent d'elle même. La productivité des entreprises en dépend. Néanmoins, La reconfiguration des processus guidée par l'expérience et les usages suit généralement un cycle de vie qui commence par une action provoquée généralement par l'équipe de direction et inclue plusieurs étapes :

- La phase de diagnostic commence par une analyse de la situation actuelle, et en particulier des problèmes provoqués par les méthodes de travail existantes. Entre autres, elle montre les situations où les méthodes de travail existantes ne produisent pas le résultat désiré.
- Une fois que le diagnostic a été fait, la phase de re-configuration suit. Les méthodes de travail existantes ne seront pas utilisées comme base de travail : une nouvelle description du processus est produite.
- Dans la phase de re-configuration, le nouveau processus est implanté dans un système de workflow.
- Pendant la phase opérationnelle, la performance des processus est mesurée et évaluée en utilisant des critères prédéfinis.

Le point faible des approches courantes de reconfiguration des processus métiers est l'absence d'outils formels permettant d'extraire de façon automatique l'expérience et les usages réels des processus métiers sous une forme exploitable pour la reconfiguration de nouveaux processus métiers plus performants et plus conformes aux besoins. Ça reste un processus très manuel.

1.4 Exemple

Considérons un exemple simplifié de traitement d'une commande de matériels informatique et multimédia par un fournisseur. Le workflow de traitement de commande est représenté sous forme graphique dans la figure 1.1.

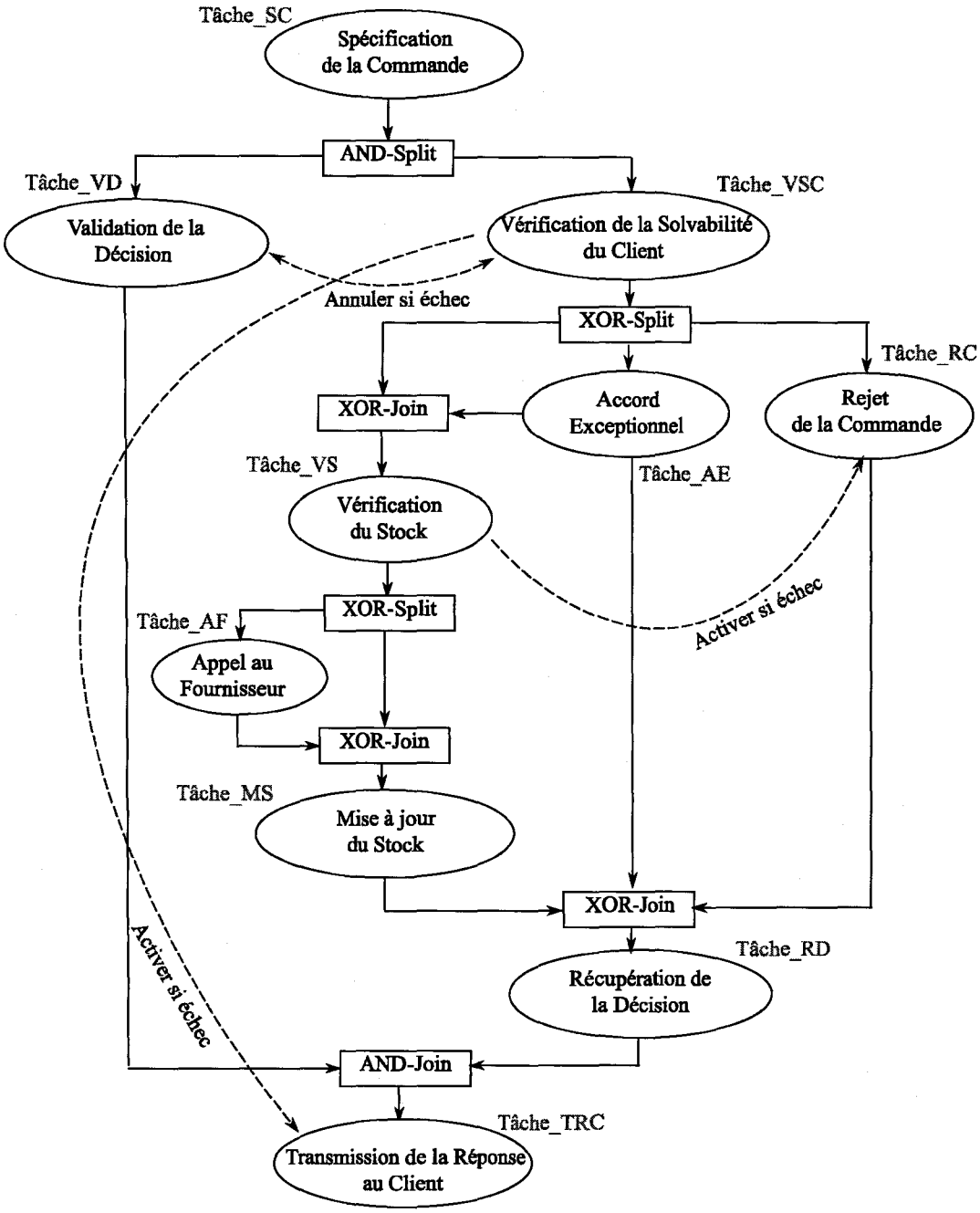

FIG. 1.1 – Exemple d'un processus de traitement de commandes.

D'abord, le client spécifie la commande et les termes de la commande : quantité, référence, date de livraison souhaitée, etc. (*Tâche_SC*). Alors une instance de workflow est lancée pour la collecte d'informations sur le client et évalue son degré de solvabilité par la tâche de Vérification de la Solvabilité du client (*Tâche_VSC*).

Après cela, l'entreprise prend sa décision en choisissant exclusivement entre trois possibilités : Si le client est solvable, la commande est acceptée. Si le client est suffisamment important (chiffre d'affaires important, notoriété sur le marché, taux de dettes acceptables par rapport au chiffre d'affaires, etc.) pour lui accorder la commande, sans évaluation de risque, alors l'entreprise approuve cette commande par la tâche d'accord exceptionnel (*Tâche_AE*). Dans les autres cas de risques, le fournisseur rejette la commande (*Tâche_RC*).

Pour les commandes acceptées, le fournisseur procède à la vérification du stock (*Tâche_VS*) et fait appel au fournisseur (*Tâche_AF*) quand il y'a une rupture de stock. Dans les deux cas, le stock est mis à jour (*Tâche_MS*).

Par la suite, la tâche de récupération de la décision (*Tâche_RD*) rédige et enregistre la décision d'acceptation et de traitement ou de rejet de la commande dans la base de données. La transmission de la réponse au client (livraison de la marchandise au client, accompagnée de la facture ou avis de rejet) est réalisée par la tâche de transmission de la réponse au client (*Tâche_TRC*).

Le traitement de la commande ne peut pas être fait sans approbation d'un contrôleur (direction commerciale de l'entreprise du fournisseur). En effet, une activité humaine de validation de la décision (*Tâche_VD*) est nécessaire tout le long du processus de décision pour contrôler ses activités. Ainsi, il n'est possible de répondre à la commande que si la direction commerciale de l'entreprise la valide. Elle peut rejeter la commande même si une décision positive a été prise. Le contrôleur peut donner librement sa décision à tout moment pendant le processus de traitement de la commande.

Initialement, pour traiter les échecs d'exécution des tâches composantes du workflow, les concepteurs spécifient en plus des tâches, des procédures de normalisation qui décrivent les relations exceptionnelles entre les tâches. Dans notre exemple, il est indiqué que si *Tâche_VD* échoue, on annule aussi l'exécution de *Tâche_VSC* et le traitement de la commande sera re-

exécutée jusqu'au succès. En plus, en cas d'échec de *Tâche_VS*, le workflow continue l'exécution par une décision de rejet de la commande *Tâche_RC*. Finalement, les procédures de traitement d'échecs ne sont pas fournies pour les autres tâches qui sont supposées se dérouler sans échec.

Supposons maintenant que sur la base d'observation d'un nombre suffisant d'instances d'exécution, la durée de la tâche de vérification de la solvabilité du client (*Tâche_VSC*) est variable selon le service (S_A, S_B, S_C) qui assure cette tâche et que cette variation se répercute sur la durée globale de traitement des commandes et donc sur la qualité de service de l'entreprise. En effet, des délais de réponse longs nuisent énormément à l'attractivité de l'entreprise, et réduit considérablement la fidélité des clients. La fidélisation des clients est un enjeux majeur dans la *Gestion de la Relation Client* (GRC) de l'entreprise. Par observation, nous constatons que cette variation dépend du type de commande traitée. Le service S_A a une bonne expérience dans le traitement des commandes de types objets portables et assistants personnels, le service S_B dans les commandes de type ordinateurs personnels, et le service S_C dans les accessoires multimédia et audiovisuel (ex. écrans plats, lecteurs DVD, etc.). Une telle découverte suggère l'amélioration du workflow en guidant la *Tâche_VSC* vers les services S_A, S_B, S_C selon l'une des trois catégories de commande traitée. Le workflow a bien évolué par rapport à sa conception initiale.

Un autre exemple concerne le constat sur la base d'autres observations que la tâche de vérification de la solvabilité du client (*Tâche_VSC*) n'échoue jamais. En même temps, il y'a de nombreuses instances où la tâche de validation de la décision (*Tâche_VD*) échoue. Ce constat nous amène à entreprendre une évolution du modèle de workflow pour répondre à une situation réelle découverte dynamiquement, et donc répondre à des comportements difficilement prédictibles à priori lors de la conception. L'objectif est d'atteindre un modèle de workflow optimal. Nous pouvons conclure à partir de ces observations deux recommandations. La première recommandation concerne la suppression du mécanisme de normalisation pour *Tâche_VSC*. En effet, il n'y a aucun besoin de spécifier un mécanisme de normalisation pour *Tâche_VSC* et ainsi suggérer de supprimer ce mécanisme de normalisation qui peut être cher. La deuxième recommandation concerne l'avortement des tâches concourantes du processus de décision et reprendre l'exécution du workflow comme mécanisme de normalisation à l'échec de *Tâche_VD*.

Le problème majeur, auquel nous nous intéressons, est comment garantir des exécutions d'instances de workflow optimales, c.à.d que des mécanismes de prédiction ou de normalisation des exécutions d'instances pour améliorer la qualité des workflow en prenant en considération des informations difficilement obtenables dans la conception initiale. Cette amélioration permet de pallier à des problèmes de qualité de service ou à tout échec d'exécution que la conception initiale du workflow n'a pas prévue. Ces recommandations répondent aux besoins d'évolution des systèmes en plaçant les utilisateurs au centre du système de workflow. Par exemple, certaines parties du processus sont plus performantes que d'autres ou encore ne jamais être atteintes lors de la phase d'exécution ou des nouvelles contraintes peuvent être observées et omises dans la phase de conception initiale. Ainsi des faiblesses de conception peuvent être traitées.

1.5 Objectif général de la thèse

L'objectif général de la thèse est de développer une méthode capable d'extraire de façon cohérente et flexible le comportement réel des instances du processus métier qui permettrait une conception intelligente et continue. L'idée est de reconfigurer le processus métier sur la base des observations des exécutions des instances. Nous parlons d'extraction de modèles de processus métiers à partir de journaux d'événements, plus généralement connu sous le terme de *workflow mining*.

Les observations sont collectées dans des journaux d'événements (fichiers *logs*) disponibles dans la plupart des workflows commerciaux. Les journaux d'événements ont été conçus pour être utilisés afin d'évaluer et surveiller les processus ainsi qu'à l'analyse de leur impact économique à travers le calcul des indicateurs de performances. Même si leurs destinations initiales sont limitées à la surveillance et à certaines évaluations, elles peuvent servir de bases de données nécessaires à l'extraction des modèles de processus. La destination des journaux d'événements est différente de la destination initiale, ce qui induit une mise à jour des informations sauvegardées (ex. instants de départ et de fin des instances de processus). La nouvelle destination des journaux d'événements est le résultat du besoin d'observer finement les instances de processus pour mieux recommander des évolutions, voire des corrections d'anoma-

lies. L'objectif ultime est de comprendre les actions, voire les comportements des utilisateurs du système pour améliorer leurs performances.

Pour ce faire, notre travail a pour objectif général de répondre aux points suivants :

- identifier les propriétés conceptuelles du modèle de processus et les motifs qui influencent l'évolution du workflow, voire la correction des erreurs.
- identifier les caractéristiques des événements de trace nécessaires à l'extraction de modèles de processus.
- identifier les méthodes de fouille de données les plus adaptées pour l'extraction des modèles de processus.
- prise en compte des informations extraites pour faire évoluer le workflow, corriger les erreurs et mettre en oeuvre des procédures de normalisation.

1.6 Contribution de la thèse

La contribution de la thèse est de proposer une approche automatique qui découvre les usages des processus métiers en développant un modèle de workflow conforme aux usages. L'approche contribue au diagnostic par l'analyse automatique de la situation actuelle, et en particulier la détection des différents motifs et leurs fréquences. Elle montre les incohérences entre les résultats attendus et les résultats réels, par exemple des tâches peu exécutées, alors que la spécification initiale prévoyait l'inverse. Cette phase de diagnostic sera nécessaire à la reconfiguration et à une nouvelle description et réalisation du processus métier.

Plus précisément, notre contribution porte sur une approche d'extraction de modèles de processus, basée sur la notion de *WF-nets* (les réseaux de workflow). Il s'agit d'une sous-classe des réseaux de Petri destinée et optimisée pour la modélisation des processus. Dans le *WF-nets* une seule instance est activée. Notre choix des réseaux de Petri est motivé par la sémantique formelle, la nature graphique, la capacité d'expression et les outils d'analyse, bien souvent libre de droit, qui caractérisent les réseaux de Petri. Le réseau de Petri est d'abord enrichi avec des motifs (structures) de branchement multiple (AND-split), rendez-vous (AND-join), d'aiguillage (XOR-Split) et de jointure (OR-Join). Ces motifs servent à représenter le routage des instances de processus, à savoir routage parallèle, séquentiel, itératif et conditionnel. Le

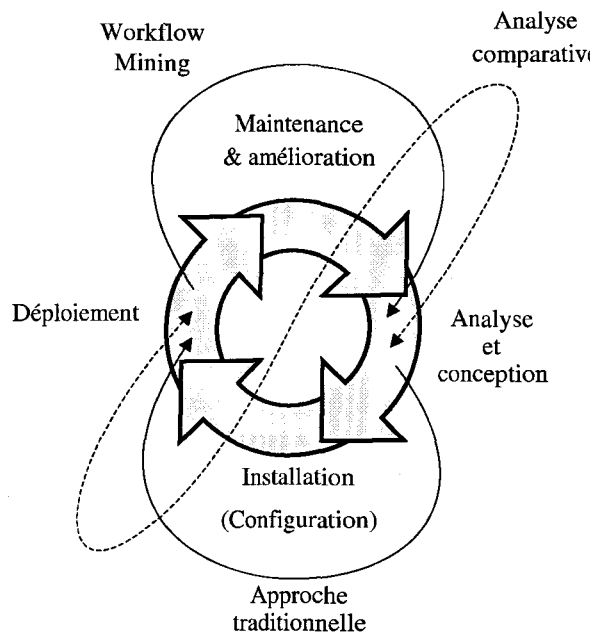

FIG. 1.2 – Cycle de vie de la modélisation d'un processus métier, l'approche traditionnelle et le *Workflow Mining*. [vdAvDH+03]

réseau de Petri est ensuite enrichi de motifs de chevauchement, boucles et tâches dupliquées. Cette enrichissement est rendu possible grâce notamment à l'élaboration d'un algorithme de construction des réseaux de Petri et à l'amélioration des traces des journaux d'événements par les durées minimales et maximales des tâches. L'algorithme de construction se base sur deux étapes importantes, l'extraction des instants de début et de fin des tâches, et l'extraction des relations de causalité entre les tâches. Dans le cadre expérimental, nous avons développé un système composé de plusieurs modules : un module de simulation de l'exécution d'un workflow et de création du journal d'évènements, un module de construction du réseau de Petri à partir des instances, et un module de visualisation des résultats obtenus. Les données du journal d'événements sont représentées sous la forme d'un document XML, selon un format assez proche de celui fourni par les moteurs de workflow disponibles sur le marché.

1.7 Intérêts de notre approche

Pour mieux comprendre l'intérêt de notre approche d'extraction de modèles de processus par rapport à l'approche traditionnelle de conception de Workflows, considérons le cycle de vie d'un Workflow illustré dans la figure 1.2. Ce cycle est constitué de quatre phases principales : l'analyse, la conception, le déploiement et la maintenance ou l'amélioration. Dans l'approche traditionnelle, l'analyse permet de définir et de construire un modèle de processus métier destiné à être exécuté selon les paramètres de la phase de configuration. Après le déploiement, les différentes instances créées suivront le modèle conçu dans les phases précédentes.

Par rapport à cette approche traditionnelle, l'intérêt de notre travail se résume dans les points suivants :

Extraction du modèle du processus métier Reconstruction (ou construction, au cas où un système de Workflow n'est pas déployé) du modèle de processus à partir de l'ensemble des journaux d'événements (fichiers Logs) qui représentent la trace d'exécution de ce processus (figure 1.3). Tous les systèmes transactionnels tels que les Progiciels de Gestion Intégrés (PGI), de Gestion de la Relation Client (GRC), de Business to Business (B2B), de Gestion de la Chaîne Logistique (GCL), sans oublier les Systèmes de Gestion de Workflow, permettent de garder les traces des exécutions des tâches sous une forme ou une autre. Ce qui constitue une source d'information importante pour la construction du modèle du processus métier.

La figure 1.3 représente un exemple d'extraction de relations entre les différentes tâches *Tâche_SC*, *Tâche_VD*, *Tâche_VSC*, *Tâche_AE*, *Tâche_RC*, *Tâche_VS*, *Tâche_AF*, *Tâche_MS*, *Tâche_RD* et *Tâche_TRC* à partir des traces d'exécution. Ainsi, la tâche *Tâche_SC* est toujours suivie par *Tâche_VD* ou *Tâche_VSC* qui s'exécutent en parallèle. Pareil pour *Tâche_MS* qui est toujours suivie par *Tâche_RD*, etc.

Analyse comparative Le modèle de processus reconstruit pouvant être différent de celui établi au cours de l'analyse, les outils développés permettront de réponde à la question : "*est-ce que ce qui est fait est ce qui a été spécifié ?*". Pour cette comparaison, les approches existantes se basent sur deux notions : la notion d'équivalence entre les modèles (ex : équivalences des

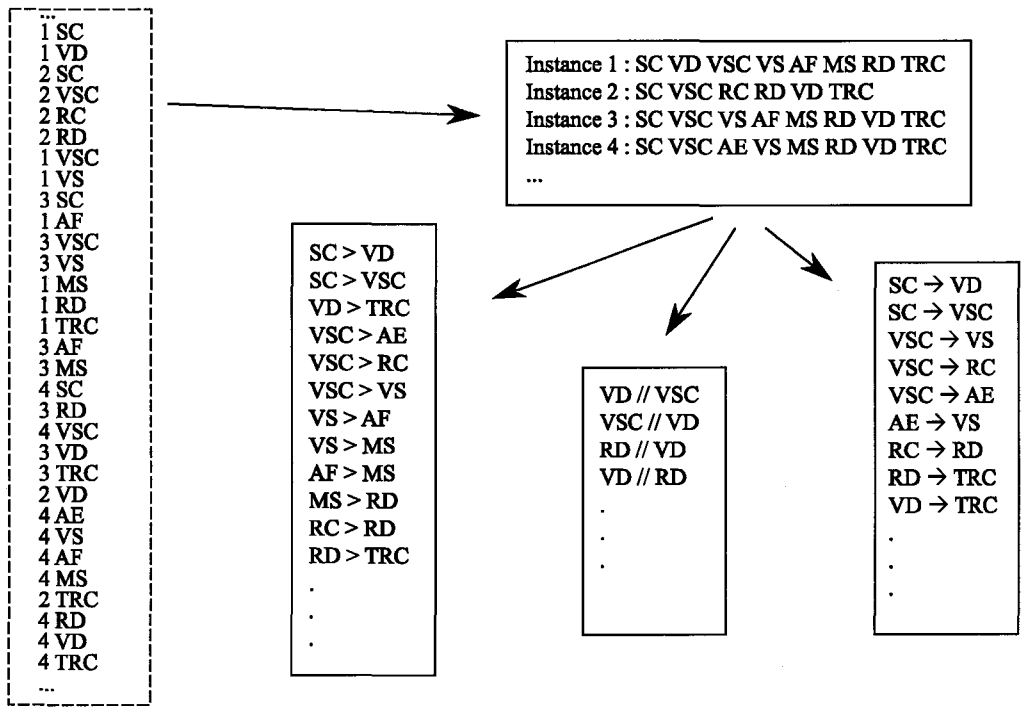

FIG. 1.3 – Exemple d'extraction de relations entre les tâches d'un processus à partir d'un journal d'évènements.

journaux d'évènements) et la notion d'adaptation (un modèle pouvant être obtenu par l'adaptation d'un autre).

Supervision et analyse de performances Evaluation des différentes métriques obtenues à partir d'exécutions précédentes (ex : fréquences des succès et des échecs, rendement des personnes, durée minimale, maximale ou moyenne de l'exécution d'une tâche ou d'un processus, délais d'attente ou de synchronisation). Ces métriques permettent d'optimiser les processus métiers en matière de complexité, de durée d'exécution, de coûts ou de ressources mobilisées. Les différents moniteurs permettront de surveiller l'état du système (services, ressources, processus) ainsi que les instances de processus en cours d'exécution et alerter l'utilisateur de la présence de cas indésirables ou exceptionnels.

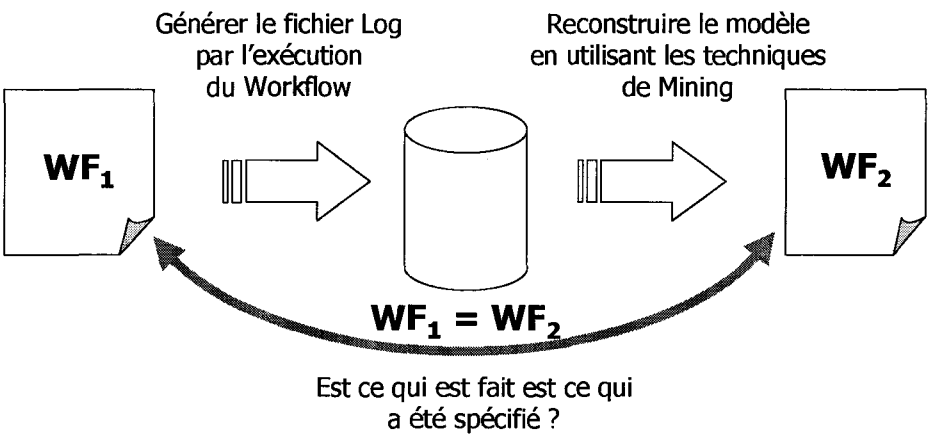

FIG. 1.4 – Analyse comparative.

Analyse prédictive Le but étant de construire des modèles de prédiction qui peuvent être appliqués aux instances de processus qui sont en cours d'exécution afin de détecter des anomalies ou des comportements non désirés et de réduire l'impact des défaillances. Ces modèles permettront aussi d'aider les utilisateurs à prendre les meilleures décisions quand plusieurs choix se présentent.

Le développement de telles solutions nous met devant les défis suivants :

- Identifier l'architecture et les technologies qui peuvent couvrirent de tels besoins, et comprendre comment les appliquer ou les modifier pour atteindre notre but.
- Définir les concepts et les métriques qui permettent une analyse des qualité des processus au niveau métier.
- Développer les techniques qui facilitent l'usage des outils au administrateurs, et de préférence sans écrire de code.
- Comprendre la manière avec laquelle un tel système doit interagir avec le système de workflow et ses utilisateurs pour intervenir, informer ou corriger les situation critiques dans des délais raisonnables.

1.8 Cadre institutionnel de la thèse

La thèse s'est déroulée dans le cadre d'une convention CIFRE entre le *Laboratoire d'Informatique Fondamentale de Lille* (LIFL UMR USTL-CNRS 8022) de l'*Université des Sciences et Technologies de Lille* et l'entreprise privée d'édition de progiciels SNEDA. La problématique de la thèse est un bon équilibre entre les compétences du laboratoire de recherche et la vocation de l'entreprise. Le laboratoire de recherche est spécialisé dans l'analyse des actions des utilisateurs dans les systèmes d'information et dans l'extraction des connaissances utiles à leur exploitation. La vocation de l'entreprise SNEDA est de servir la stratégie des acteurs de l'immobilier, par l'optimisation de leur système d'information, et par son ambition d'innover et de devenir la référence française en informatique immobilière. SNEDA, créée en 1976, est une SSII spécialisée dans la fourniture de solutions progicielles et de services pour la gestion des patrimoines immobiliers et fonciers. Numéro 2 de l'informatique immobilière en France avec un chiffre d'affaires de 10,4 millions d'euros en 2005, SNEDA compte 116 collaborateurs présents sur l'ensemble du territoire français. Le sujet de thèse est le fruit d'une intéraction importante entre la SNEDA et le LIFL. Cette intéraction avait pour seul objectif de répondre aux besoins de l'entreprise en terme d'amélioration de la productivité de ses clients à travers des systèmes de workflows qui tiennent compte de la réalité des usages.

1.9 Structure de la thèse

La thèse est composée de six chapitres. Le premier chapitre introduit le rapport. Le deuxième chapitre présente l'état de l'art scientifique et technique du domaine, et en particulier la notion de workflows et de leur modélisation. Le but étant de montrer la diversité des techniques et la complexité croissante de la gestion des workflows. Le troisième chapitre présente les motifs usuels de workflows.

Le quatrième chapitre traite des problèmes liés à l'exploitation des fichiers logs, ainsi que la méthode que nous proposons pour améliorer la qualité des processus et de leur exécution. Le cinquième chapitre est consacré à la présentation de l'outil développé au sein de la société *Sneda* dans le cadre de cette thèse, ainsi qu'à la présentation de quelques exemples et cas

d'usage. Enfin le dernier chapitre conclut le rapport.

Notre approche est générique et peut s'appliquer à d'autre domaine que ceux des applications classiques de processus métiers (et plus généralement, les systèmes d'information). Nous avons expérimenté notre approche dans le domaine de l'extraction de parcours types et classes de comportements dans les parcours visuels sur un média numérique.

Chapitre 2

État de l'art

2.1 Introduction

Les tâches complexes doivent être structurées et représentées sous forme de modèle pour faciliter leur gestion ainsi que l'automatisation de leur exécution. La technologie des workflows a été développée pour répondre à ce genre de besoins.

Ce chapitre est composé de deux parties. La première partie présente la technologie des workflows ainsi que leur modélisation. La deuxième partie présente les techniques actuelles d'extraction de connaissances à partir de workflows.

2.2 Le travail collaboratif

Le travail collaboratif est devenu une des préoccupations majeures des entreprises. A cela deux raisons essentielles ; la première est le développement du travail en mode projet dont une des caractéristiques est la mobilisation de compétences multiples. La seconde est la tendance à l'éclatement des structures qui rend précisément le mode projet difficile à mettre en œuvre.

Il y a plus de deux siècles, Adam Smith avançait la théorie que la productivité est proportionnelle à la division du travail et soulignait ainsi le rôle du travail collaboratif dans un fameux texte sur la "fabrique d'épingles" : dix ouvriers travaillant chacun de leur côté ne parviennent

pas à produire plus de 20 épingles par jour et par ouvrier. Si chacun d'eux se spécialise dans une étape de la fabrication, les cadences montent à 4 800 épingles par jour et par ouvrier.

Les outils de *Travail Collaboratif Assisté par Ordinateur* (TCAO) se sont développés pour répondre à ces préoccupations. Ils permettent en effet de travailler en mode asynchrone - c'est à dire en temps décalé - et de manière distante tout en assurant la structuration, le suivi ainsi que la traçabilité des échanges. On peut définir les principales fonctions d'un outil de travail collaboratif de la manière suivante : Aide à la réalisation de taches communes, et aide à la planification et à la synchronisation du travail.

Aujourd'hui, la standardisation des protocoles procurées par les technologies Internet contribuent grandement à l'essor de ce type d'applications. Quels sont ces outils ? Citons ceux que l'on rencontre les plus fréquemment dans le milieu des entreprises en nous basant sur la classification suivante :

- Les outils de communication.
- Les outils de partage.
- Les outils de coordination.
- Les outils de gestion des processus.

2.2.1 Les outils de communication

Le plus répandu de ces outils est la messagerie électronique. Il présente toutes les caractéristiques d'un outil de travail collaboratif. Il permet le travail distant et asynchrone, il peut être accompagné d'un document électronique, s'adresse à un seul ou à un groupe de destinataires.

Directement hérité d'Internet, le forum d'entreprise se présente comme une place de discussion publique (accessible à toute l'entreprise) ou privée (un service, un groupe de projet,...), sur laquelle chacun peut initier une discussion sur un thème quelconque ou poser une question. Comme la messagerie, il permet le travail distant et asynchrone à cette différence près qu'il permet de consulter n personnes sans avoir à préjuger de qui aurait une réponse ou un commentaire intéressant. Les échanges sont visibles au travers d'une arborescence et comme cela se pratique sur certains forums Internet, il est fortement recommandé qu'une synthèse des contributions soit produite par la personne qui a initié la discussion. L'usage du forum peut

donc également permettre l'émergence et la localisation de certaines connaissances tacites de l'organisation qui seront ainsi explicitées. On peut encore citer le tableau d'affichage où les utilisateurs peuvent laisser des notes consultables par les autres.

2.2.2 Les outils de partage

C'est l'ensemble d'outils qui permettent aux utilisateurs de partager leurs informations et leur espace de travail ou encore de partager des fichiers, voire des applications. Ils peuvent ainsi interagir entre eux directement et mener des actions ensemble. On peut citer pour l'exemple le "tableau Blanc" de *Net Meeting*©ou encore les annuaires. Généralement on distingue deux types d'annuaires ; les premiers sont les annuaires d'entreprises où l'on retrouve les coordonnées (téléphone, fax, mail, ...) des différents acteurs de l'entreprise. Il peut être à usage exclusivement interne mais également externe. Les seconds, sont les annuaires des différents partenaires de l'entreprise (clients, fournisseurs, sous traitants, ...).

2.2.3 Les outils de coordination

Dans cette catégorie, on peu citer l'agenda partagé. Cet outil encore peu répandu dans les PME/PMI offre pourtant des fonctions tout à fait intéressantes en terme de gestion du temps des acteurs ainsi que des ressources de l'entreprise. Il permet en effet de confier la gestion d'un agenda à plusieurs personnes sans risque d'erreur, d'organiser des réunions en tenant compte des disponibilités des participants et du matériel nécessaire (salles, vidéo projecteur, ...). En tant qu'outil de coordination des ressources de l'entreprise, l'agenda partagé est un véritable outil de travail collaboratif.

2.2.4 Les outils de gestion des processus

La mise en œuvre de ces outils répond à des préoccupations d'automatisation et de gestion des processus d'une entreprise. Ces outils prennent en charge la coordination et la synchronisation des différents acteurs et des ressources impliqués dans le processus et s'assurent de mettre à leur disposition toute l'information nécessaire à la réalisation de leurs tâches. Ils per-

mettent également de visualiser l'état d'avancement d'une procédure et d'identifier les raisons d'un éventuel retard. Pour simplifier, on peut classifier ces outils en deux catégories :

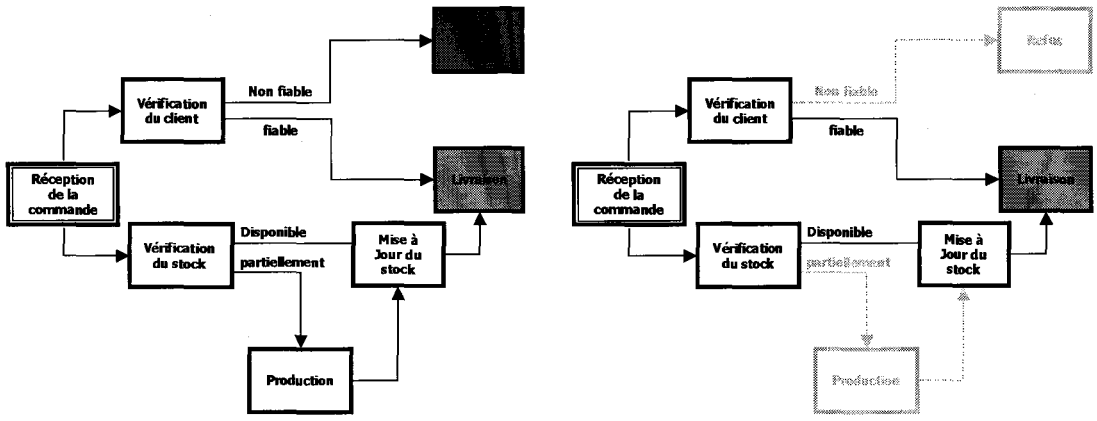

FIG. 2.1 – Exemple d'un processus de gestion de commandes.

Les workflows *Ad-hoc*, *collaboratifs*, *administratifs*, ou *de production* [McC92, LR99, GT98, GHS95]. Ces quatre types de workflows sont classés suivant leur usage et leur fréquence d'utilisation. Les workflow Ad-hoc ainsi que les workflow collaboratif sont utilisés par un groupes de personnes qui travaillent ensemble pour atteindre un but commun. Ces workflow ne sont souvent pas définis à l'avance à cause d'une répétitivité réduite. Les workflows collaboratifs (ex. développement d'un programme par plusieurs personnes) on une valeur ajoutée plus importante que les workflows Ad-hoc (ex. organisation d'une conférence). Les workflows administratifs ainsi que les workflows de production ont par contre une répétitivité plus importante et nécessitent une définition d'un modèle. Les workflows de production (ex. les assurances, services fiscaux), qui représentent le support du métier des organisations, ont une valeur ajoutée plus importante que les workflows administratifs (ex. calcul des salaires, gestion des factures).

Les workflows administratifs : Ainsi que leur dénomination l'indique, ces outils sont orientés vers la gestion des processus administratifs auxquels ils lient l'information et les documents nécessaires à l'accomplissement des tâches de chaque acteur impliqué. Par ailleurs, ils

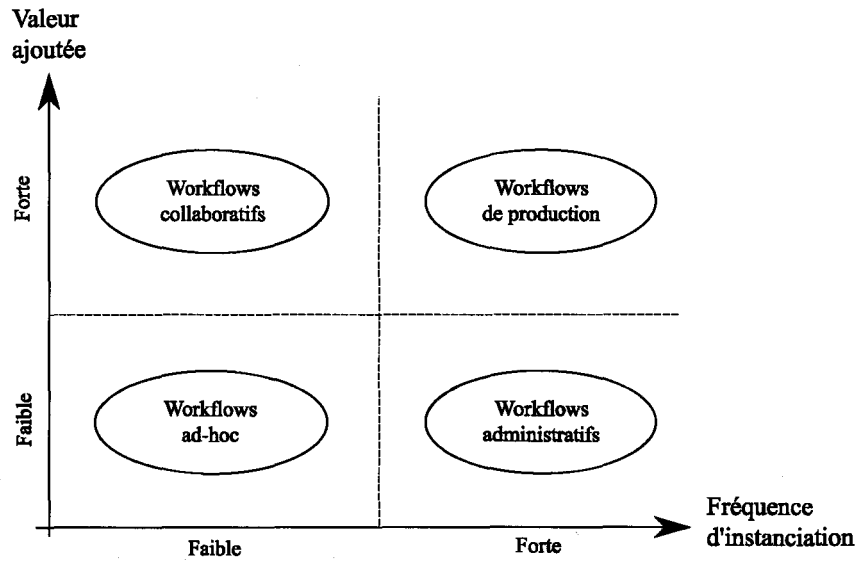

FIG. 2.2 – Classification des workflows suivant leur fréquence d'instanciation.

prennent en charge le routage de formulaires électroniques (via messagerie souvent). Chaque utilisateur ayant accès à un certain nombre de formulaires (demande de congés, demande d'achat,...) qui seront acheminés automatiquement vers le destinataire approprié pour accord ou refus. Ils sont généralement mis en œuvre sur des procédures simples et stabilisées dans le temps.

Les workflows de production : Ce type de workflow gère les processus directement liés aux services que l'organisation propose et dont son efficacité dépend (la gestion des sinistres pour une compagnie d'assurance par exemple). Les procédures concernées sont généralement plus complexes que dans le cas d'un workflow administratif.

Secteur	Applications
Distribution	Gestion logistique
Services bancaires	Approbation de crédit Traitement des prêts hypothécaires
Assurances	Traitement des demandes de règlement Gestion des polices
Santé	Gestion des cas Dossiers médicaux
Services publics	Gestion de la configuration Révision et approbation des documents Traitement des ordres de travail Gestion de la sécurité
Cartes de crédit	Traitement des transactions
Éducation	Traitement des demandes d'admission Demandes de relevés de notes
Fonds communs de placement	Traitement des commandes Demandes de renseignements des clients
Transports	Suivi des lettres de transport Suivi des marchandises
Administration publique	Gestion des publications Demandes de permis de conduire traitement de l'impôt et des déclarations de revenus Renseignements généraux
Administrations municipales	Suivi des contraventions Enregistrement foncier
Fabrication	Suivi des nomenclatures
Commerce de détail	Traitement des commandes
Applications intersectorielles	Traitement des factures (comptes fournisseurs) Service à la clientèle Traitement des déplacements et des frais Gestion du recrutement

TAB. 2.1 – Exemples d'application par secteur d'activité.

Les workflows ad-hoc : Ici, il s'agit de tâches ou activités qui sont plutôt associées à des projets qu'à des traitements intensifs. Si les workflows de production gèrent des tâches

répétitives, les workflows ad-hoc sont soumis à des objectifs dont les étapes et les niveaux d'interaction entre les intervenants sont plus difficiles à définir en détail et à prévoir.

Les outils de gestion de projet sont suffisants pour ce type de tâches. Les workflows ad-hoc ont besoin des fonctionnalités de ces outils pour planifier et contrôler les opérations prévues initialement. Le workflow ad hoc est par nature communicant, alors que le workflow de production est plus orienté "traitement" en fonction de circuits et de règles établies à l'avance : Un dossier de prêt suit toujours le même processus d'approbation alors que l'organisation d'une conférence (workflow ad-hoc) est soumise à un degré d'interaction très fort entre les organisateurs.

Les workflows collaboratifs : Les workflows collaboratifs sont souvent considérés comme du groupware. Ils se concentrent sur le travail d'équipe en vue d'atteindre des objectifs communs. La taille des groupes peut-être très variable. Elle peut aller du petit comité avec une organisation orientée projet, au grand groupe réparti à travers le monde et ayant des intérêts en commun. Les workflows collaboratifs ont pour but de faciliter les communications intergroupes. Par rapport aux workflows de production ou ad-hoc, ils font beaucoup plus souvent appel aux moyens de communication qui permettent l'ajustement mutuel des individus impliqués. Ils sont également caractérisés par un cadre procédural relativement ouvert et plus complexe car moins déterministe dans sa mise en oeuvre.

2.3 Avantages fonctionnels des applications du workflow

Les voies des systèmes de workflow favorisent le transfert de flux de contrôle entre les tâches, en conformité avec les règles en vigueur. Les logiciels de workflow permettent de relier l'ensemble des tâches d'une entreprise selon une séquence logique et font le suivi des états fonctionnels. Les tâches (une demande du service à la clientèle, par exemple), peuvent être déclenchées de différentes manières, y compris par un message de courrier électronique, un appel téléphonique ou la réception d'un document.

Les systèmes automatisés de workflow peuvent offrir de nombreux avantages fonctionnels, dont voici les plus courants :

2.3.1 Accroissement de la productivité

La plupart des organisations se doivent de maximiser l'utilisation de leurs ressources et de veiller à consacrer leur temps aux activités qui contribuent directement à leurs objectifs et leur métier. Grâce aux systèmes de workflow, les tâches sont saisies, mises en ordre de priorité et acheminées vers la personne ou la fonction la mieux placée pour les accomplir. Les règles opérationnelles et les instructions décrivant la tâche à accomplir accompagnent l'acheminement. Le workflow permet également de réduire considérablement le traitement des erreurs, le contrôle de la qualité et les reprises.

2.3.2 Raccourcissement des cycles de travail

Le workflow offre les moyens de contrôler la durée du cycle de vie d'un processus métier, de son déclenchement jusqu'à son achèvement. Il permet de supprimer les délais de traitement superflus, comme le temps passé à transférer des documents et des dossiers de papier. Il n'y a plus de documents perdus ou mal classés. Certaines activités, comme l'envoi de lettres types ou la télécopie de réponses standardisées, peuvent être entièrement automatisées par le système de workflow.

2.3.3 Amélioration du contrôle

Les fluctuations du volume des activités et les engorgements potentiels peuvent être facilement contrôlés, et des mesures correctrices prises pour régler les problèmes à mesure qu'ils surviennent. Les systèmes de workflow peuvent fournir, à tout moment au cours d'une journée, un "instantané" des tâches non accomplies et produire des mises à jour sur la distribution des tâches et les activités en cours. Ils permettent également de compiler de précieuses données de gestion, telles que le type de transactions traitées, les délais de traitement moyens et les modèles d'activités fonctionnelles.

2.3.4 Amélioration du service à la clientèle

La clé d'un bon service à la clientèle est une réponse rapide et efficace aux demandes des clients. Un système de workflow peut permettre de traiter toutes les demandes de façon diligente et de mettre en tête de file les tâches qui remplissent certaines conditions prédéterminées (comme une demande de votre meilleur client ou un prêt hypothécaire dont l'échéance tombe dans deux jours). De plus, le logiciel de workflow réunit tous les éléments et renseignements nécessaires à l'accomplissement de la tâche, y compris des données sur les clients et leurs transactions commerciales.

2.3.5 Avantage concurrentiel

La mondialisation des marchés accroît la concurrence. Grâce à l'avancement des technologies et des communications, n'importe qui peut faire affaire avec un client ou un fournisseur donné, quelle que soit la distance. Pour être concurrentiel, il faut pouvoir réagir rapidement aux changements du marché et aux nouvelles demandes des consommateurs. Un système de workflow permet à l'organisation de réagir plus rapidement. Les règles et procédés opérationnels sont facilement adaptables et peuvent refléter sans délai les changements du marché.

Le rendement du capital investi varie selon le secteur d'activité et l'application ; toutefois, la plupart des organisations enregistrent des améliorations de leur productivité de 20 à 50% et des bénéfices dans les trois années qui suivent.

2.4 Les workflows

Un Workflow est défini comme étant l'automatisation d'un processus métier. On distingue deux aspects : l'aspect statique qui englobe la modélisation et la représentation des processus métier et l'aspect dynamique qui traite des instances d'exécution des processus.

Un modèle de processus métier est une description des activités d'une organisation en terme de tâches, d'agents, de règles et de procédures. Dans les contextes dynamiques et compétitifs actuels, les processus métiers sont sujets à de fréquentes et inévitables évolutions. Il est ainsi nécessaire aux technologies qui assurent l'automatisation de ces processus de permettre

l'adaptation des modèles aux changements requis. L'évolution d'un modèle de processus est une décision stratégique soigneusement planifiée respectant les différentes phases du cycle : proposition, définition, analyse, mise à jour, déploiement et évolution.

La phase d'analyse

C'est la phase de modélisation des procédures, sous la responsabilité des organisateurs de l'entreprise. Elle est issue plutôt des méthodes de BPR (*Business Process Reengineering*), dont l'objectif est de remettre à plat tous les processus de l'entreprise pour en rebâtir de nouveaux, plus efficaces et mieux adaptés. A ces phases de BPR sont toujours associés les outils informatiques. C'est dans ce cadre que ce sont essentiellement développés les outils de workflow, puisque parfaitement optimisés pour le traitement automatisé des procédures de l'entreprise.

La phase de construction

Elle consiste, à partir de la modélisation effectuée lors de la phase d'analyse, à formaliser les procédures résultantes au sein d'un outils informatique, et à définir l'ensemble des conditions nécessaires à son bon fonctionnement, et à son intégration dans l'informatique existante. Les produits de workflow évolués permettent d'utiliser un mode de représentation graphique des procédures.

La phase d'exécution

C'est la phase pendant laquelle les procédures sont exécutées et les tâches traitées, C'est également pendant cette phase que les statistiques, fondamentales pour le suivi de tout processus, sont générées. Des outils d'administration doivent également exister afin de pouvoir intervenir à tout moment sur les procédures elles-mêmes en cas de problème. Bien entendu tous les produits de Workflow intègrent ce module.

Pour mettre en place une terminologie pour les spécifications, ainsi que pour permettre les discussions entre utilisateurs, analystes et chercheurs, les termes de bases liés aux processus et aux workflows doivent être définis. plusieurs papiers proposent une terminologie pour ces concepts et des relations qui les lient [DNR90, FH92, LS97, WfM02]. Les concepts définis par [WfM02] sont les plus utilisés par les industriels dans la définition de leur processus métiers. La liste suivante représente les structures et les concepts de base comme le suggère la WfMC.

2.4.1 Processus

Un autre concept clé du domaine du TCAO est celui de processus. Le travail au sein d'un groupe s'organise, de façon implicite ou explicite, autour d'un processus de coopération qui peut être plus ou moins structuré. Ce processus décrit la manière dont la coopération entre les membres du groupe doit se dérouler, décrivant comment les données doivent circuler à l'intérieur du groupe ou encore quelles activités doivent être exécutées par le groupe afin d'atteindre ses objectifs. David [Dav01] a constaté que les activités professionnelles sont plus ou moins structurées. Les membres du groupe ne sont pas généralement banalisés, leur participation au traitement d'un dossier, par exemple, est définie par des règles précises spécifiant le cheminement du dossier dans l'organisation, pour que celui-ci soit traité, approuvé, etc. Ces règles définissent donc ce que nous appelons ici le processus de coopération. Certains de ces processus sont très structurés, tandis que d'autres ne le sont pas. Dans le cas des processus structurés, tout le processus peut être exprimé dans un logiciel [Man98]. On parle donc de flots de travail ou workflow, et les workflows qui contrôlent ces flots sont appelés systèmes de gestion des flots de travaux ou *Workflow Management Systems* [WfM99]. Ces systèmes sont utilisés pour structurer le travail entre les individus dans une organisation, surtout lorsque le processus qu'ils formalisent est souvent répété et implique de multiples individus [Man98]. Lorsque le processus coopératif est complexe, il devient important d'en produire une représentation explicite pouvant servir de guide et de support au contrôle de son exécution [ECO01]. Cependant, expliciter un processus de coopération n'est pas toujours possible ou souhaitable. Une alternative à cela consiste à définir ce processus de façon implicite, à travers notamment les tâches et les conventions sociales du groupe. Pour Ellis et al. [EGR91], le contrôle du processus de coopération peut être pris en compte par les conventions sociales du groupe, qui sont consenties et connues des membres du groupe, mais non nécessairement contrôlées par le workflow.

2.4.2 Tâche

C'est la description d'une partie du travail à effectuer qui forme une seule entité logique dans le workflow. Elle peut âtre manuelle ou automatique [vdAvH02]. Une tâche manuelle (à l'inverse d'une tâche automatique) nécessite l'intervention d'une personne.

C'est un concept directement lié, par un lien de composition, à celui de processus. Une tâche correspond à une action menée par un ou plusieurs membres du groupe afin d'atteindre l'objectif commun. Il s'agit d'une unité de base du travail en groupe. Souvent, pour atteindre son objectif, le groupe organise le travail en termes d'tâches (et sous-tâches) qui sont réalisées par les membres du groupe. Elles composent, en réalité, le contenu du processus de coopération : si le groupe est capable d'expliciter et de structurer ses tâches, nous avons donc un processus bien défini et structuré. Dans le cas contraire, nous avons un processus implicite, semi- ou non-structuré. Par exemple, la rédaction d'un rapport technique peut s'organiser à l'intérieur soit d'un processus structuré, capable d'utiliser un workflow pour le guider, soit d'un processus semi-structuré. En revanche, un brainstorming va, en général, relever d'un processus non-structuré. Par ailleurs, il est important d'observer la relation directe qui existe entre le concept d'tâche et celui de rôle. Par définition, un rôle définit les responsabilités et les droits des membres du groupe. Il est donc normal qu'un rôle puisse indiquer les droits relatifs à une tâche. Souvent les workflows attribuent directement aux rôles le droit de réaliser ou non une tâche. Par exemple, dans le cas de l'édition coopérative, la tâche d'édition d'un document est habituellement associée au rôle d'auteur, tandis que le rôle de lecteur ne peut que consulter le document. À travers cet exemple, apparaît également le lien qui existe entre les rôles et le contrôle d'accès aux données partagées. Ce contrôle s'effectue à travers l'attribution des droits d'accès, lesquels peuvent être attribués soit individuellement à chaque membre du groupe, soit à des rôles en particulier.

2.4.3 Acteur

Aussi appelé *Agent*, *Utilisateur* ou *Participant*, est l'entité qui effectue une ou plusieurs tâches dans une instance de workflow. Ça peut être une personne comme ça peut être une machine.

Si nous appelons l'ensemble des utilisateurs un groupe, chaque individu dans ce collectif est appelé membre du groupe ou, tout simplement, participant du groupe. Il s'agit, du point de vue de la conception des workflows, des utilisateurs de ces systèmes. Un membre du groupe est donc quelqu'un qui participe aux tâches de ce groupe, partageant ainsi avec les autres membres

(ses collègues) un objectif commun. Cependant, il est important d'observer qu'un individu peut appartenir, simultanément ou non, à plusieurs groupes. Ainsi, de façon plus générale, « chaque participant est un individu personnalisé qui appartient à un ou plusieurs groupes » [TB97].

2.4.4 Instance

C'est la représentation de l'exécution d'un processus une seule fois de son début jusqu'à sa fin. C'est ainsi la suite des actions et opérations qui ont été exécutées pour le traitement d'un cas donné, en respectant le processus dont relève ce cas. Une instance peut être automatique (sans intervention humaine) ou manuelle. dans certaines œuvres, on parle aussi de *cas d'exécution*, ou d'*instanciation de processus*. On peut citer comme exemple, toute demande, litige, note de frais, courrier, appel, etc..., ou tout terme désignant l'objet qui a déclenché et qui est traité par un processus.

2.4.5 État

C'est une notion liée aux workflows et aux tâches dans un intervalle de temps et suivant les conditions interne à l'exécution du workflow. dans le cas des tâches, ça peut être "*active*", "*inactive*", "*suspendue*", "*en cours d'exécution*", "*terminée*". Dans le cas des workflows, ça peut âtre "*initialisé*", "*en cours d'exécution*", "*actif*", "*suspendue*", "*terminée*" ou "*archivé*". On peut trouver d'autres variantes et d'autres termes, surtout lorsqu'il s'agit de produits spécifiques.

2.4.6 Rôle

C'est un groupe d'agents qui ont un ensemble de propriétés requises spécifiques. On dit qu'un agent assume un rôle lorsqu'il a les compétences pour l'ensemble des pré-requis de ce rôle.

Les membres s'organisent au sein du groupe à travers le ou les rôles qu'ils y jouent. Un rôle représente les responsabilités d'un utilisateur (ou d'une classe d'utilisateurs) vis-à-vis du groupe. Le rôle est souvent lié aux droits dont dispose cet utilisateur dans l'environnement

coopératif. Il s'agit d'un ensemble de privilèges et de responsabilités attribué à une personne [EGR91]. Pour Tarpin-Bernard [TB97], un rôle est caractérisé par l'ensemble des droits et des devoirs du participant vis-à-vis de ses partenaires et des données partagées, pouvant évoluer au cours du temps. Nous pouvons ainsi considérer que chaque membre du groupe peut disposer d'un certain nombre de rôles (un nombre supérieur ou égal à un) dans le groupe, mais, à un instant t, chaque membre jouera effectivement un rôle en particulier (parmi ceux dont il dispose). Salcedo [Sal98] parle, dans ce cas, de plusieurs rôles potentiels et d'un rôle effectif pour se référer, respectivement, aux divers rôles que l'utilisateur peut jouer dans l'équipe et celui qu'il joue effectivement au moment présent. Les rôles représentent donc l'organisation interne du groupe et sa structure hiérarchique. Cette organisation détermine les responsabilités de chacun dans le groupe. Elle a aussi une grande influence sur la façon dont se déroulent les interactions entre les membres du groupe. Le concept de rôle, selon David [Dav01], est capital pour comprendre ces interactions. Pour cet auteur, un échange entre les membres ne répond pas du hasard. Pour lui, il n'y a pas d'interaction sans un minimum de réciprocité, chacun doit s'efforcer de réagir de façon adéquate à la conduite de l'autre, et le rôle fonde les attentes réciproques. Il sert à orienter les interactions. Ainsi, on trouve des groupes très hiérarchisés qui comptent plusieurs rôles organisés en divers niveaux, et d'autres groupes avec des structures très aplaties, avec un rôle unique (le participant) ou deux rôles : les participants et le responsable de l'équipe, celui qui joue un rôle de leader, appelé le plus souvent coordinateur. Le choix entre une structure plus ou moins hiérarchique ne dépend pas seulement du groupe et des tâches que le groupe doit exécuter, mais aussi du réseau social (l'organisation, la société, etc.) dans lequel il se trouve. Et selon ce choix, les interactions entre les membres du groupe seront plus ou moins libres. Par ailleurs, à partir du constat qu'un rôle est un indicateur des responsabilités des individus qui le jouent dans le groupe, nous pouvons donc imaginer ce rôle comme un indicateur des informations nécessaires à ces individus pour mener à bien leurs responsabilités. En fait, les membres jouant des rôles distincts ont des besoins aussi distincts, certains rôles nécessitant des informations plus détaillées et plus précises que d'autres [KPLB01].

Nous pouvons faire notamment la différence entre les informations nécessaires au coordinateur du groupe et celles nécessaires aux autres participants du groupe. Un coordinateur a

besoin d'une vue de l'ensemble de la coopération afin de mieux suivre l'avancée des travaux et de prendre les décisions en conséquence. Un participant n'a pas forcément besoin des mêmes informations, puisqu'il n'a pas les mêmes responsabilités vis-à-vis du groupe. Ceci démontre bien toute l'importance que le concept de rôle a pour les collecticiels. Il s'agit d'un concept qui doit être bien maîtrisé par les concepteurs.

2.4.7 Session

La notion de *session* correspond à une période d'interaction assistée par l'outil de travail collaboratif [EGR91]. Il s'agit de la période dans laquelle un ou plusieurs utilisateurs sont connectés au système. Pour Laurillau [Lau02], cette période est celle pendant laquelle les utilisateurs participent aux tâches du groupe. Dans le contexte d'une organisation, la notion de session correspond à l'acte de présence au sein de l'organisation [TB97]. Techniquement, une session peut être vue comme un ensemble d'objets partagés et une série de moyens de communication communs connectant un certain nombre d'instances de l'outil de travail collaboratif [LS03]. Nous considérons dans notre cas qu'une session est principalement la période pendant laquelle les utilisateurs sont connectés au système et participent directement aux tâches du groupe.

2.5 Les systèmes de gestion de workflow

Nous présentons dans ce qui suit, les moyens et les méthodes de gestion des processus métier. Récemment, la puissance des outils à aider à définir et contrôler les différentes tâches associées à un processus métier à considérablement augmenté. Ces outils sont connus sous le nom de "Systèmes de gestion de workflow " (Workflow Management systems, WfMS).

2.5.1 La Workflow Management Coalition (*WfMC*)

La Workflow Management Coalition (*WfMC*) est une organisation internationale qui regroupe plus de 300 membres fournisseurs de système de Workflow à travers le monde. Ils

représentent toutes les facettes du domaine : fournisseurs de logiciels, universités, groupes de recherche et consultants. Sa mission va de la promotion et le développement de l'usage des workflows à l'établissement des standards d'interopérabilité, de connectivité et de terminologie pour les systèmes de workflow.

2.5.2 Le modèle de référence de la WfMC

La WfMC a développé un modèle de référence pour la technologie des workflows [WfM02]. Ce modèle est une description générale de la structure d'un système de gestion de workflow et à comme but de fournir un standard pour l'interopérabilité entre les principaux composants. Cinq principaux composants sont présentés : le service de définition des processus, l'application client, les applications invoqués, le service d'exécution du workflow, et enfin le composant d'administration et de supervision.

2.5.2.1 Le service d'exécution des processus

Le service d'exécution est le composant principal d'un système de gestion de workflow. Il représente l'environnement de création, de gestion et d'exécution des instances du processus métier. Il permet, entre autres, les fonctionnalités suivantes :

- L'authentification des participants.
- La création et la suppression d'une instance.
- Le contrôle des instances créées (suspension, reprise, annulation, ...).
- Gestion du routage (interprétation du modèle du processus métier).
- Gestion des conditions (évaluation et interprétation).
- Attribution des tâches aux participant (suivant la classification et la hiérarchie des participants).
- Lancement d'un application liée à une activité du processus en récupérant les données nécessaires.
- Enregistrement des traces d'exécution des tâches.

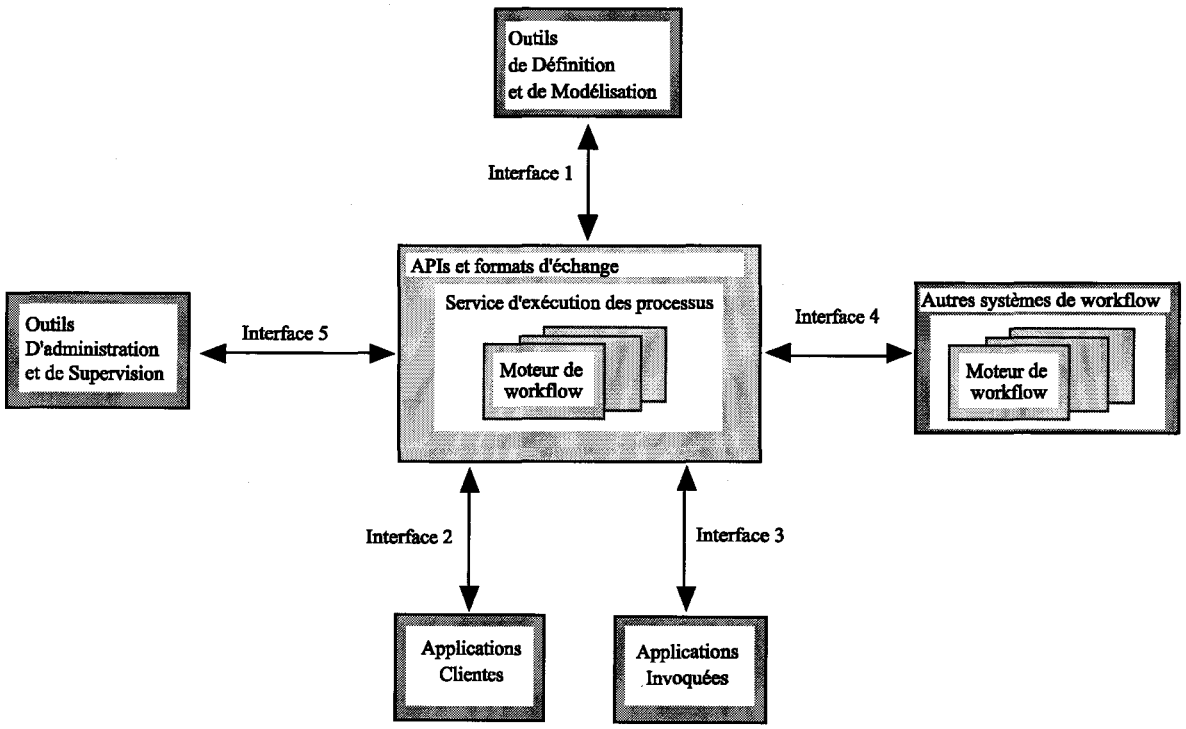

FIG. 2.3 – Modèle de référence des workflows - composants et interfaces.

2.5.2.2 Le service de définition des processus

Pour couvrir les aspects fournis par le service d'exécution, les processus métier doivent être définis, et les resources et le rôles doivent être attribués. C'est le rôle du service de définition des processus. En plus de la modélisation des processus et la classification et l'attribution des rôles, ce service fournit parfois des fonctions d'analyse, de vérification et de simulation. Ces fonctions restent limitées dans les systèmes actuelles et sont l'un des objectifs de cette thèse.

La définition du processus C'est une étape qui permet de décrire le processus métier d'une manière graphique ou textuelle. Certaines caractéristiques doivent être définies pour chaque tâche :

- Le nom et la description de chaque tâche.

- L'utilisateur (le rôle, le groupe) associé ou l'application qui va effectuer la tâche.
- Les conditions d'exécution de la tâche.
- Les attributs mis à jour à la fin d'exécution de la tâche.

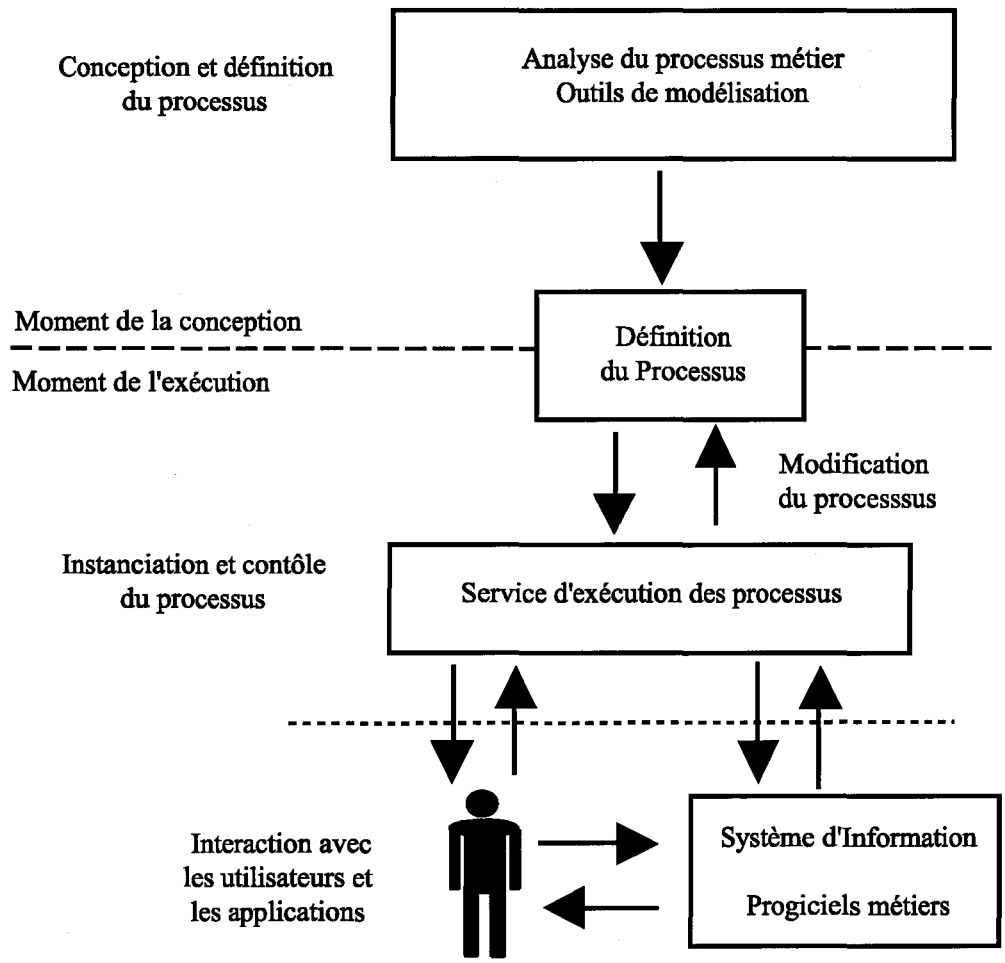

FIG. 2.4 – Caractéristiques d'un système de Workflow.

Classification des ressources Lorsqu'un processus est défini, il est recommandé fortement recommandé d'associer les tâches aux resources, qu'elles soient humaines ou matérielles.

L'outil de classification des ressources, considéré comme composant du service de définition des processus, permet d'établir les liens entre les différentes classes de ressources (les rôles et les hiérarchies organisationnelles) et de leur attribuer des caractéristiques spécifiques. Les rôles sont basés sur la fonction, les qualification et les compétences, tandis que la hiérarchie organisationnelle permet de regrouper les participants en équipes, départements ou filiales.

Analyse L'outil d'analyse, qui peut être intégré dans le service de définition des processus, permet de faire des simulations pour tester les processus avant de les déployer.

2.5.2.3 Les applications clients

Quand un processus métier est lancé (initialisé, instancié) par le service d'exécution, les utilisateurs et les applications concernés sont planifiés et doivent compléter les activités qui leur sont attribuées au fur et à mesure de la progression du processus. La communication entre l'utilisateur et le système de workflow est couverte par l'application client. Une liste de tâches personnelles est générée pour montrer à chaque utilisateur quelles sont les tâches qui doivent être faites.

2.5.2.4 Les applications invoquées

Accomplir une tâche peut nécessiter le lancement d'une ou plusieurs applications externes par le service d'exécution des processus. Une distinction est faite entre les application interactive et les applications entièrement automatiques. Les applications interactives sont initialisées au même temps que la tâche pour interagir avec l'utilisateur (pour remplir un formulaire par exemple). Les applications automatiques quand à elles, ne nécessitent pas l'intervention d'un utilisateur (programmes de calcul ou d'édition en masse par exemple).

2.5.2.5 Service d'administration et de supervision

Ce service est réparti en deux type d'outils : ceux qui servent à la gestion dynamique et le contrôle des processus, et ceux qui permettent de sauvegarder les historiques et générer des rapports.

2.5.3 Quelques produits commerciaux

Dans cette section nous présenterons huit moteurs de workflow et leurs langages disponibles dans le commerce. Ces produits ont été choisis sur la base principale de leur disponibilité et de leur popularité dans le milieu industriel. Nous étudierons les offres des systèmes FileNet, Visual Workflo, Staffware et MQ Series. MQ Series a suscité un grand intérêt dans le monde universitaire

2.5.3.1 IBM WebSphere MQ Workflow

IBM WebSphere MQ Workflow, anciennement appelé *IBM MQ Series Workflow* [IBM04], est un système de gestion de workflow des *IBM Software*. Ces composants, basé sur l'architecture trois tiers (couche présentation, couche métier, couche accès aux données), sont :

- L'éditeur graphique, pour créer les modèles de processus. Cet éditeur permet, entre autres, de définir les aspects organisationnels (personnes impliqués, rôles, etc.) et les aspects liés à l'exécution (structures de données, applications invoquées, etc.).
- L'application client fournit au utilisateurs un espace pour instancier un processus et ainsi de lancer, d'arrêter ou de superviser son exécution. Elle fournit aussi la liste des tâches à accomplir par un collaborateur.
- L'outil d'administration pour gérer le lancement et l'arrêt des moteurs de workflow, ainsi que les ressources qui ont été spécifiées au moment de la modélisation du processus. L'outils vérifie régulièrement l'état des serveurs pour fournir des rapport systèmes à l'administrateur.
- Le serveur de gestion des processus, qui inclut quatre types de serveur : Le serveur d'exécution (qui gère l'exécution des instances des processus), le serveur de planification (pour le lancement différé des instances et des tâches), le serveur de nettoyage (pour la suppression et l'archivage des instances terminées) et Le serveur d'administration (pour gérer les autres serveurs).

IBM WebSphere MQ Workflow s'exécute sur la plate-forme applicative *WebSphere* qui couvre un ensemble de solutions développées par IBM qui permettent de développer, de déployer et d'utiliser des applications d'entreprise faisant appel à des applications et des ma-

tériels hétérogènes. Cette plate-forme fournit une gamme d'outils de développement basés principalement sur le socle de développement Eclipse et le langage java et une gamme de serveurs d'application basés sur J2EE et EJB. IBM WebSphere MQ Workflow utilise des bases de données relationnelles (DB2 or Oracle) en différenciant la base de données qui stocke les informations liées à la modélisation des processus de celle qui stocke les informations liés à leur exécution.

2.5.3.2 BEA WebLogic Integration

BEA WebLogic Integration [BEA04] rassemble tous les composants nécessaires pour assurer une intégration globale et robuste des applications d'entreprise :

- Un middleware pour d'intégrer les applications à travers l'ensemble de l'entreprise, du Web jusqu'au mainframe,
- La livraison et l'intégration des logiciels de gestion de workflow opérationnels.
- Des connecteurs pour les principaux systèmes ERP, les services de données financières et d'autres applications verticales spécialisées,
- La livraison et l'intégration des meilleures offres de répartiteurs d'intégration,

La solution EAI de BEA est composée de produits technologiques clés associés à des logiciels spécialisés incorporés et supportés par BEA, avec des services optionnels de formation et de conseil. BEA offre ainsi une large gamme de logiciels et de services permettant de résoudre les problèmes d'intégration d'applications critiques de production dans un environnement toujours plus distribué et hétérogène. Les entreprises peuvent désormais s'appuyer sur une infrastructure robuste pour s'assurer que des systèmes distincts (d'Internet jusqu'aux mainframes) et des applications différentes (qu'elles soient fortement personnalisées ou qu'il s'agisse de progiciels standards) soient capables de communiquer de façon efficace.

2.5.3.3 SAP NetWeaver

SAP NetWeaver se compose des éléments suivants : Composants

- SAP NetWeaver Application Server : prend en charge les services Web indépendants des plates-formes, les applications de gestion ainsi que le développement basé sur des

normes ouvertes, vous permettant d'exploiter pleinement les ressources technologiques existantes pour des solutions orientées services Web

- SAP NetWeaver Business Intelligence : donne la possibilité d'intégrer toutes sortes de données, indépendamment de leur origine et de transformer les informations en connaissances, les connaissances en actions.
- SAP NetWeaver Exchange Infrastructure : fournit des technologies d'intégration ouvertes qui permettent une collaboration axée sur les processus, à l'intérieur et à l'extérieur de l'entreprise
- SAP NetWeaver Master Data Management : assure la distribution homogène des informations à toutes les applications et à tous les systèmes de votre environnement informatique et vous aide à intégrer vos processus métier sur l'ensemble de votre chaîne de valeur
- SAP NetWeaver Mobile : offre un environnement nomade puisqu'il repose sur des normes technologiques ouvertes et souples ainsi qu'un environnement de développement très performant, vous permettant d'élaborer des solutions nomades intégrées à interfaces natives ou de type navigateur
- SAP NetWeaver Portal : fédère les informations et les applications clés afin de présenter aux utilisateurs une vue unifiée de leurs activités, à l'échelle de l'entreprise, garantissant un rendement maximum de vos investissements informatiques
- SAP Auto-ID Infrastructure : donne la possibilité d'intégrer tous les lecteurs automatiques - y compris les lecteurs et encodeurs RFID, les périphériques Bluetooth, les systèmes embarqués, et les lecteurs de code barre

Nous nous intéressons dans notre thèse au module *SAP NetWeaver BUSINESS INTELLIGENCE* qui permet entre autres les fonctionnalités suivantes :

- Entreposage des données : La gestion des entrepôts de données, la modélisation des informations, l'extraction, la transformation et le chargement des données sont autant de fonctions qui vous aident, d'une part, à bâtir des entrepôts de données et à modéliser une architecture d'informations en adéquation avec la structure de l'entreprise et, d'autre part, à gérer des données émanant de plusieurs sources.
- Plate-forme de Business Intelligence : Grâce à OLAP (online analytical processing), et

à la gestion des alertes, il est possible d'accéder aux données et de les présenter, de faire ressortir des modèles et d'identifier les exceptions.
- Vision stratégique de l'entreprise : Les outils de conception de requêtes, de reporting et d'analyse, et de conception d'applications vous aident à créer des états analytiques, à prendre des décisions (à tous les niveaux de l'entreprise) et à présenter les applications BI sur le Web.
- Mesure et gestion : Grâce à la gestion du contenu métier et des métadonnées, et par le biais de la Business Intelligence, vous pouvez suivre les progrès, créer des modèles de reporting, garantir la cohérence des données et inciter les décideurs à collaborer avec leurs pairs.

2.5.3.4 Xerox DocuShare

Xerox DocuShare est une application de gestion de contenu d'entreprise et de documents qui permet aux utilisateurs d'entreprises petites et grandes et aux petits groupes de travail de stocker, collaborer et partager les informations avec lesquelles ils travaillent au quotidien.

Xerox DocuShare permet de constituer un système d'archivage sécurisé, un système fiable et solide de gestion de contenus, un portail d'entreprise ou encore une application de gestion de projets et de collaboration. Il offre ainsi les fonctionnalités essentielles pour la gestion de contenu pour les entreprises :
- Gestion électronique de documents.
- Numérisation et lecture automatique de documents.
- Module de consultation et de recherche documentaire.
- Système de workflow collaboratif.
- Publication web des contenus.

La plate-forme de gestion de processus de *Xerox DocuShare* s'appuie principalement sur deux modules :

1. *Workflow Studio* qui permet l'édition graphique des modèle de processus (figure 2.5). Cet éditeur fournit un assistant contenant un ensemble de modèles de processus prédéfinis pour un gain de temps en développement et en déploiement et se base sur un format XML

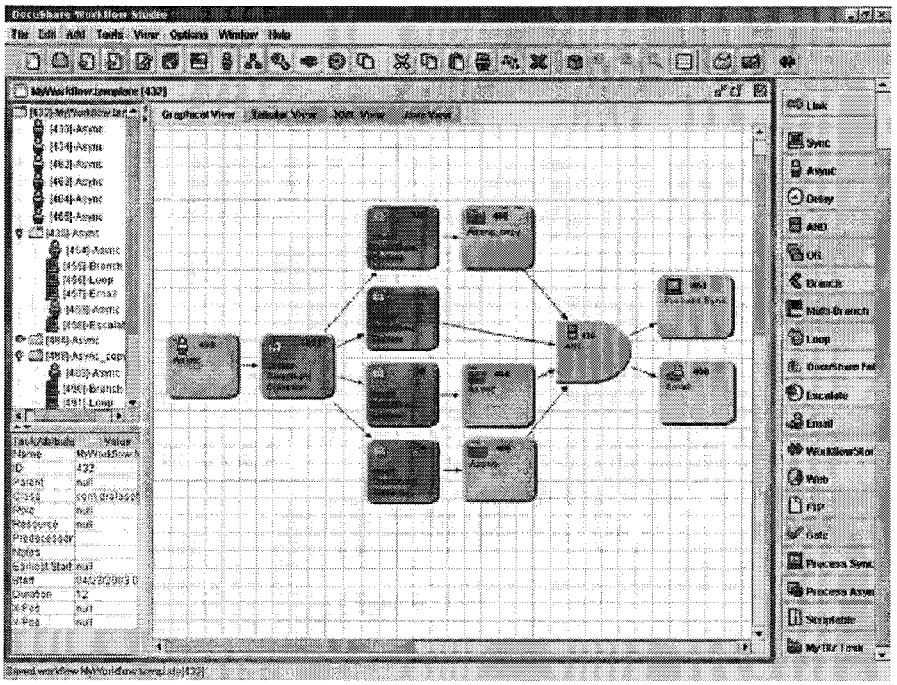

FIG. 2.5 – Interface de modélisation de processus dans Docushare Workflow Studio.

pour un maximum de compatibilité avec les applications tiers. Le moteur de gestion de processus de *DocuShare* supporte la mise en œuvre des motifs suivants :

- le routage séquentiel ou parallèle,
- les branchement multiples et les jonctions,
- le routage conditionnel,
- les boucles,
- la gestion des exceptions,
- les sous-processus,

2. *Workflow Manager* qui permet d'instancier, de lancer, d'arrêter et de superviser l'exécution des processus. C'est aussi ce module qui permet l'enregistrement des évènements survenus lors de l'exécution des processus.

2.5.4 Windows Workflow Foundation

Windows Workflow Foundation (WF) fait partie des nouvelles API de Windows nommées WinFX et disponible sur Windows Server 2003, Windows XP SP2 et bien sûr Windows Vista. Il est constitué d'un ensemble de classes .Net, d'un moteur d'exécution et d'outils (designer Visual Studio 2005, debugger...) qui permettent de construire des applications ayant besoin de workflows.

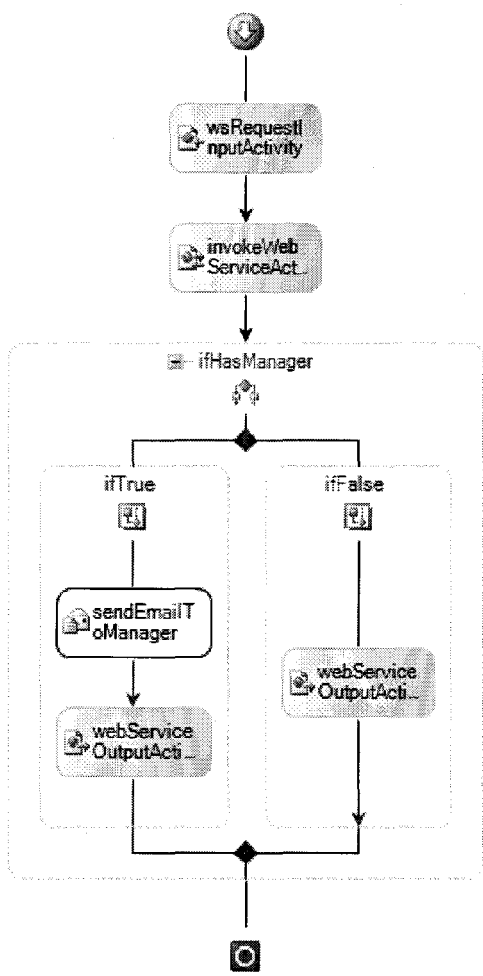

FIG. 2.6 – Exemple de la modélisation d'un processus sous Windows Workflow Foundation.

Sans oublier de citer les prototypes de recherche tels que FlowMake [SO99], ADEPT$_{flex}$ [RD98] et YAWL [vdAtH05]. Il existe aussi des alternatives *open source* aux produits industriels. Cependant, si le nombre d'outils de de gestion de processus métiers issus de l'univers *open source* est important, les solutions véritablement matures ne sont pas nombreuses. Parmi ces solutions, on peut citer :

- Jboss jBPM (http ://www.jbpm.org),
- ObjectWeb Bonita : (http ://bonita.objectweb.org),
- OpenSymphony OSWorkflow (www.opensymphony.com/osworkflow/),
- OpenWFE (http ://openwfe.sourceforge.net),
- PowerFolder (http ://www.powerfolder.org),
- WfMopen (http ://wfmopen.sourceforge.net),
- et l'éditeur Enhydra JaWE (http ://www.enhydra.org/workflow/jawe/).

2.6 Modélisation de workflows

Deux principales méthodologies sont utilisées dans la modélisation des workflow :

- Méthodes orientés "communication" : le but est de modéliser les communications entre les acteurs du workflow.
- Méthodes orientés vers les activités : on traitent l'ensemble des activités dans le workflow.

2.7 Workflows adaptatifs

La capacité des workflows à s'adapter dynamiquement aux situations et aux contextes est un besoin essentiel pour les systèmes de gestion de workflow [RRD04]. Ce besoin est issu de la nécessité de réagir par rapport aux évènements externes et inattendus.

Les experts du métier préfèrent souvent se restreindre à modéliser les chemins et les cas les plus fréquents seulement.

Les workflows adaptatifs sont étudiés par une multitude de chercheurs de la communauté workflow.

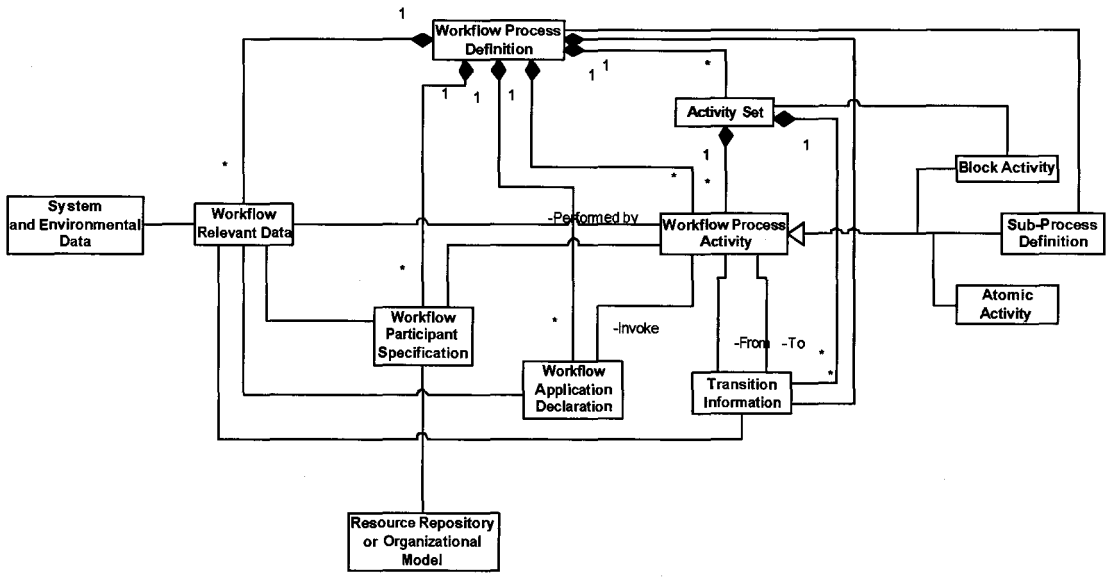

FIG. 2.7 – Méta-modèle de définition de processus.

Il y'a une distinction à faire entre (1) les modifications et adaptations ponctuelles et ad-hoc, qui ne touchent qu'une seule instance du processus, et (2) les évolutions qui concerne le processus tout entier et qui concerneront ainsi toutes les instances futures. Les modifications ponctuelles sont causées souvent par des cas uniques et rares qui nécessitent un traitement exceptionnel, alors que les évolutions des processus sont effectuées, par exemple, lors des modifications des lois ou des règles internes des organisations, ou lorsque la maintenance ou l'optimisation d'un processus devient nécessaire.

2.7.1 Problèmes liés à l'adaptation des workflows

2.7.1.1 L'expressivité du méta-modèle de workflow utilisé

Le méta-modèle de workflow doit fournir les constructions nécessaires à l'adaptation dynamique des processus métier. Cela peut être interprété de deux manières :

– De manière automatique.

– De manière pratique.

2.7.1.2 La complétude par rapport aux opérations de construction

L'ensemble des opérations de construction offertes par le système doit être complet dans le sens où il doit permettre de réaliser tous les processus possibles.

2.7.1.3 La vérification de l'exactitude des modification appliquée

Il s'agit de s'assurer que les modifications apportées ne rentrent pas en conflit ou causent des erreurs dans l'exécution des instances en cours.

2.8 Le travail collaboratif

La conception des processus métiers a fait l'objet de travaux intenses au cours de ces dernières années où de nombreux outils de conception ont vue le jour. Il est certain que la façon avec laquelle des processus métiers sont structurés a une grande influence sur les performances des produits et services sur lesquels ils seront bâtis. Nous présentons, ci-dessous, les différentes approches de conception et les techniques d'aide à la conception de processus métiers dans les systèmes de workflows. Nous présentons aussi des approches nécessaires la re-configuration des processus métiers, et donc à la re-conception de workflows.

2.8.1 Méthodologie de conception classique des processus métiers

La conception traditionnelle des processus métiers est guidée par quatre étapes successives : la planification des besoins, la construction du modèle de workflow, l'intégration et finalement la validation du modèle conçu.

2.8.1.1 Planification, construction, Intégration

La phase d'analyse et de planification des besoins constitue la première étape qui consiste à analyser les besoins et les objectifs du processus métier à concevoir. Il consiste aussi à le

placer au mieux dans son véritable contexte d'usage. Les utilisateurs des workflow produisent ces spécifications, et les transmettent aux concepteurs sous la forme de paramètres d'entrées. Les objectifs peuvent être quantifiés pour être repris par la suite dans la dernière phase comme outils de mesure pour la validation des performances du workflow conçu.

Ainsi, durant la phase de planification des besoins, les directives pour les différentes fonctionnalités et les résultats attendus du processus sont définis. En se basant sur les spécifications de la phase d'analyse et la planification des besoins, la deuxième étape de construction du workflow est lancée. Pendant cette étape de construction, les différentes spécifications du processus sont mises en avant sous la forme d'un modèle de workflow en utilisant l'outil d'édition du système de workflow choisi pour l'implantation du processus métier. Les caractéristiques du workflow conçu sont élaborées interactivement dans des ateliers communs de conception à travers des prototypes qui peuvent être sujets de tests par les concepteurs et des utilisateurs potentiels.

L'étape de construction peut être divisée en plusieurs étapes parallèles. En effet, le processus est souvent trop grand pour être conçu en un seul bloc. Par conséquent, il peut être utile de développer le workflow dans un certain nombre d'étapes séparées. Chaque étape se termine avec la livraison d'un nouveau modèle de workflow qui est une évolution du précédent.

Ces différentes versions sont par la suite intégrées dans la troisième étape d'intégration. Cette étape peut se faire dans le cadre d'une collaboration étroite entre les concepteurs du workflow, d'une part, et ses utilisateurs, d'autre part, sous la forme d'un développement commun. Les utilisateurs peuvent ainsi soumettre des amendements aux propositions des concepteurs pour coller au mieux aux objectifs définis dans la première étape. Mais l'organisation d'une telle coopération reste assez compliquée, ce qui engendre un manque d'efficacité.

2.8.1.2 Validation des workflows

Avant que le workflow conçu soit mis en pratique, il est indispensable de le valider, c'est à dire de vérifier que d'une part, le modèle conçu ne contient pas d'erreurs conceptuelles ou d'erreurs fonctionnelles et de s'assurer, d'autre part, que l'ensemble des objectifs définis initialement est atteint dans la réalisation du workflow. La dernière étape de validation analyse le modèle de workflow conçu et s'assure qu'il répond le plus fidèlement possible et sans erreurs

les spécifications de la première phase.

Cette étape peut inclure des techniques de validation et d'analyse pour parfaire un diagnostic qui peut détecter des problèmes de mise en oeuvre conceptuelle. Ces techniques permettent en particulier d'expérimenter, de mesurer et d'évaluer les performances en utilisant des métrique spécifiques pour s'assurer que la réalisation du workflow converge vers les objectifs définis initialement. De nombreux travaux de la littérature technique, comme ceux de van der Aalst [vdA98a] proposent des approches d'évaluation théorique pour l'amélioration des workflows conçus.

Cette étape comporte aussi la vérification de l'exactitude syntaxique d'un modèle de workflow. Contrairement au contexte des langages de programmation, où la syntaxe se rapporte seulement au langage, elle incorpore dans sa notion l'exactitude syntaxique de la structure et du comportement du schéma de workflow. Particulièrement, quand un modèle de workflow devient grand (parfois des centaines de tâches), il est difficile pour les concepteurs humains de contrôler le modèle complet. Un grand nombre d'outils dédiés, comme par exemple l'outil ExSpect [VBvdA01], ont été développés pour valider le workflow en considérant la masse des tâches.

Cette validation constitue une étape cruciale dans le cycle du vie du workflow. Elle implique l'exactitude syntaxique du modèle. Bien que les spécifications de la phase initiale soient la source appropriée pour dériver ce qui devrait être fait, des fausses interprétations ou des utilisations inexactes peuvent être la cause d'une conception défectueuse de workflow.

D'une manière plus générale, il est important de valider la conception de processus avant la mise en oeuvre et l'exploitation du workflow. Il est bien connu que les erreurs de conception qui sont tardivement détectées dans le cycle de développement sont très coûteuses à corriger. D'après certaines estimations, une erreur de conception détectée le cycle de développement du logiciel, respectivement pendant la programmation, le test, et la maintenance est 3, 10 et 100 fois plus coûteuse que si elle est détectée pendant la conception.

Cependant ces méthodes de validation manquent de mécanismes de correction permettant d'optimiser le modèle du workflow conçu. En plus ces méthodes se caractérisent par une approche statique, du fait qu'elles analysent un modèle conçu figé, et l'évolution du workflow induit un re-développement complet, ce qui induit un coût fort élevé. En fin, il y a très peu

d'outils qui font évoluer le modèle de workflow sur la base des observations constatées dynamiquement (lors de l'exécution).

2.8.2 Méthodes de conception de workflow

Quel intérêt dans une nouvelle méthode de conception basée sur des observations constatées dynamiquement par rapport aux méthodes traditionnelles de conception, à forte connotation statique ? Le point de départ de la réponse à la question est de constater qu'il existe de nombreuses méthodes de conception statique. Ces méthodes, de conception attachent de l'importance à la manière avec laquelle le workflow se présente et interagit avec ses utilisateurs. Les utilisateurs sont impliqués autant que possible dans la conception des processus. Ceci signifie que la conception du processus sera améliorée, par une évaluation et une révision continue jusqu'à ce qu'elle soit satisfaisante.

Nous distinguons principalement deux types de conception :

- la conception nouvelle où le processus est conçu pour la première fois. Il s'agit de la première version du modèle de workflow ;
- la conception évolutive où le processus est conçu en prenant en compte le processus existant comme point de départ. Il s'agit de la n-ième version du modèle de workflow.

La méthodologie traditionnelle, présentée dans la section précédente, adopte le premier type de conception, c'est à dire que nous nous situons dans la phase initiale du processus métier. Ce type de conception, malgré sa popularité, est caractérisé par trois inconvénients majeurs :

- la conception nouvelle est incapable de bénéficier de la connaissance et de l'expérience des conceptions précédentes ni des usages du workflow. Les défauts ne sont pas corrigés, puisqu'elle ne tire pas profit de l'expérience et des usages ;
- les utilisateurs ne sont pas dans la boucle de conception. Le risque d'utiliser un modèle qui ne soit pas conforme à leurs besoins n'est pas faible ;

Rappelons que la WfMC définit le workflow comme : l'automatisation complète ou partielle d'un processus métier selon un ensemble de règles procédurales [WfM99] et définit le système de workflow comme un système qui définit, crée et gère l'exécution des workflows par

l'utilisation d'un logiciel qui est capable d'interpréter la définition d'un processus [WfM99], mettant ainsi l'accent sur la phase d'exécution qui est considérée comme la finalité de la phase de conception. En effet, une conception ne peut être correcte que si sa phase d'exécution fonctionne correctement. Malgré cette évidence de l'importance de la dimension dynamique dans la conception, nous pouvons facilement observé que l'intérêt de la conception traditionnelle pour la prise en compte de la dynamique du processus métier et la prise en compte de la re-conception.

Il semble évident et reconnu par l'ensemble des professionnels du workflow qu'une conception qui prend en compte les usages et l'expérience acquises dynamiquement du processus métier est le préabas indispensable d'une approche complémentaire à la conception traditionnelle, et elle semble être la réponse naturelle pour pallier les erreurs et répondre au mieux à la réalité des exécutions des workflows. Ce constat est d'autant plus important qu'il est le fruit de plusieurs années d'automatisation, de la part beaucoup d'organismes professionnels et qui mettent en avant la nécessité de la complémentarité entre les deux types de conception. Une telle complémentarité exploite les usages et l'expérience acquis. Elle est donc bien placée pour obtenir un plus grand rendement. Cette conception reste cohérente avec les objectifs du BPR (*Business Process Reconfiguration*) ou CBI (*Continuous Business Improvement*) où il est question de processus d'amélioration continue. La conception induit une interaction accrue avec les utilisateurs, contrairement à la conception traditionnelle qui impose une analyse statique. Une fois l'analyse réalisée, elle est suivie de la phase de re-conception. Dans la phase de création, un nouveau modèle est créé pour supporter le processus nouvellement conçu. Pendant la phase d'exécution, une nouvelle phase opérationnelle est lancée, ce qui nous permet de mener à nouveau une nouvelle phase d'analyse et de validation qui peut bien justifier le lancement d'un nouveau cycle de re-conception, comportant très probablement de nouvelles améliorations.

2.8.3 Méthodes de conception basée sur des observations constatées dynamiquement

La nécessité d'extraire des modèles de processus ou des motifs fréquents a été reconnue et traitée de plus en plus sérieusement au cours de ces dernières années. Le terme souvent utilisé

dans la littérature est "process mining". Un nombre croissant d'approches sont proposés avec comme dénominateur commun de prendre pour données en entrée (input data) les journaux d'événements qui gardent les traces d'exécutions des instances de processus. L'approche est donc entièrement automatique sans aucune interaction manuelle. Les journaux d'événements comportent des informations comme les identificateurs des tâches et des instances réalisées.

L'un des premiers travaux dans le domaine concerne l'idée d'utiliser les journaux d'événements pour l'extraction d'informations, il a été réalisé la première fois par Cook et autres (ex. [CW98], [CW99]). L'idée donc d'extraire des modèles de processus n'est donc pas très nouvelle, puisque plusieurs travaux se sont penchés sur le sujet. Le but étant d'extraire des motifs fréquents de comportements à partir des journaux d'événements. L'objectif était d'exploiter au mieux les journaux d'événements pour comprendre l'usage du workflow. Les travaux en question visaient à améliorer le cycle de développement de logiciels, en extractant des motifs de séquences entre les composants logiciels développés. Les motifs sont exprimés sous forme d'automates d'états finis déterministes. Dans [CW98], il est question de décrire trois approches pour la découverte de modèles de processus : une utilisant les réseaux de neurones, une autre purement algorithmique, et une troisième approche utilisant les statistiques. Des extensions à l'extraction des modèles de séquences ont été proposés pour tenir compte de la concurrence entre les tâches [CW98a].

D'autres travaux comme ceux de [SABS02] proposent une modélisation avancée des workflows, en considérant les accès concurrents et la récupération après incidents en généralisant les notions de Transaction, Atomicité et Isolation de processus des Workflows. Nous pouvons citer aussi les modèles basés sur les Réseaux de Petri ([vdA98b]). Les Réseaux de Petri représentent un outil formel très puissant pour la vérification de certaines propriétés des processus.

Les travaux de [AGL98] ont été les pionniers dans le domaine de l'extraction de modèles de processus métiers, afin d'améliorer les systèmes d'information. Les modèles de processus se présentent sous forme de séquences entre les tâches en faisant abstraction des points de synchronisation entre les tâches, comme la jointure et la fusion. Ces travaux sont inspirés des modèles de graphes des produits issus du marché tels que IBM MQSeries ou InConcert... L'algorithme α-algorithm présenté par [vdAvDH$^+$03] est celui qui est le plus proche de notre approche. A partir d'un journal d'événements, cet algorithme permet de construire un réseau

de Petri représentant le modèle de processus équivalent, sauf qu'il se base sur des fichiers logs atomique (où juste l'identificateur d'une tâche est enregistré dans le fichier log). L'exploitation d'autres données extraites du fichier log ainsi que toute autre information concernant les tâches qui composent un processus permettra l'amélioration du modèle reconstruit.

Les travaux de [AGL98] ont été les vrais précurseurs des systèmes d'extraction de modèles de processus, car tous les travaux qui ont suivi se sont basés sur ses principes. Parmi ces travaux, nous pouvons citer ceux de [GP03] qui prend en considération le début et la fin de chaque tâche, ce qui rend possible l'extraction des tâches concurrentes.

Nous pouvons aussi citer les approches de [HK04] et de [Sch04] qui prennent en compte les tâches dupliquées, et génèrent un modèle de processus minimal qui couvre l'ensemble des instances du processus et des motifs fréquents. D'autres approches (ex. [GGP05]) extraient un arbre hiérarchique de modèles de processus qui décrivent les événements du journal à divers niveaux d'abstraction. L'approche commence par extraire un processus racine pour les différents niveaux d'abstraction. Ensuite, elle fusionne les modèles des niveaux d'abstraction inférieurs en commençant des feuilles de la structure hiérarchique.

Des approches, basées sur des structures plus formelles (ex. [dMWvdA05]) ont été proposées. Elles génèrent un modèle de processus basé sur les réseaux de Petri.

Certaines approches comme celle de [JVvdAR06] proposent d'extraire le modèle de processus avec des informations manquantes comme l'identité des tâches réalisées. En fait, elles proposent d'extraire le modèle de processus sur la base de journaux d'événements incomplets. Elles partent du principe que certains systèmes de workflow gardent finement l'identité des traces et instances exécutées. Des motifs fréquents sont certainement extraits et comparés à un modèle de référence. Bien entendu ce type de démarche complique quelque peu la fiabilité du modèle de processus extrait.

Si la présence des journaux d'événements dans les systèmes de workflow a été largement utilisé pur extraire les modèles de processus métiers, il n'en demeure pas moins que d'autres finalités ont été mis en avant, comme l'extraction et l'analyse des réseaux sociaux [vdARS05]. L'objectif étant de bien comprendre la manière avec laquelle les acteurs d'un système interagissent entre eux, et d'optimiser l'organisation de travail en équipe. Le raisonnement porte plus sur les acteurs des instances exécutées que sur les instances elles mêmes.

Nous pouvons par ailleurs signaler l'extrême difficulté de trouver des jeux de tests communs permettant de comparer les performances des différentes approches proposées. La raison étant la confidentialité des données. En fait, c'est un problème récurrent pour toutes données (journaux d'événements) particulièrement sensibles par le fait qu'elles sont révélatrices du fonctionnement interne d'un système organisationnel, et par conséquent révélatrices de ses faiblesses. Les entreprises ne veulent pas voir ce type d'informations au main de la concurrence.

D'autres approches ont été proposées comme ceux de [GGMS03] qui considèrent la probabilité que l'exécution d'une instance du workflow atteigne un état E, et en particulier la probabilité qu'une instance du workflow se termine avec succès ou avec échec. L'expression utilisée est *Successful Termination Prediction* (STP). Des variantes de ce type d'approche est d'extraire les terminaisons d'instances avec succès les plus fréquentes (FSTP). Considérant que l'exécution d'une instance du Workflow a atteint un état E, et qu'à partir de cet état l'administrateur a un choix entre une multitude d'activités. Le problème consiste à déduire d'après les instances passées quel serait le plus fréquent choix pour atteindre une configuration désirée. D'une manière générale, comme indiqué dans la plupart des références ci-dessus, l'extraction des motifs fréquents, connue sous le terme de *Frequent Pattern Mining* (FPM) ont dominé les travaux de l'état de l'art. Le principe le souvent utilisé est de répondre à la question : Quelles sont les séquences d'exécution les plus fréquentes ?

Actuellement, certaines recherches s'orientent vers l'extraction localisée de motifs ou de modèles de workflow à partir de journaux d'événements. L'idée est de se baser sur une fraction de traces d'exécutions, c'est à dire un journal d'événements incomplet, pour extraire des informations d'une partie du processus métier. Ces recherches sont particulièrement utiles dans le cas de présence d'un journal d'événements incomplet.

2.8.4 Synthèse

Notre approche utilise des techniques de calcul de fréquence, via une table de dépendance entre les tâches. Cette table est combinée avec un journal d'événements enrichi pour extraire un modèle de processus métier et des motifs de comportements, en tenant compte du motif de

la diffusion parallèle, du motif de jointure. Ces deux motifs permettent de tenir compte de la concurrence entre les tâches. La concurrence concerne des parties quelconques du modèle. Par ailleurs, notre approche extrait des boucles, grâce notamment à l'enrichissement du journal d'événements par les durées minimales et maximales tâches réalisées. Le tableau ci-dessous résume les particularités de notre approche par rapport aux méthodes de l'état de l'art scientifique. Nous précisons, que les systèmes de workflows présents sur le marché ne disposent pas de méthodes d'extraction de modèles de workflow. La comparaison se base sur plusieurs critères :

La représentation de l'information extraite La représentation concerne des formalismes comme les réseau de Petri, les graphes etc. Certaines représentations se limitent à décrire les relations de dépendance entre les tâches, d'autres à extraire l'ensemble du modèle de processus, d'autres à associer la sémantique aux points de synchronisation (diffusion, jointure), et d'autres à extraire des parties du modèle de processus métier.

La globalité/localité du traitement Elle indique si la méthode extrait l'information sur l'ensemble du workflow ou sur une partie du workflow. Le principe de la globalité/localité fait l'objet actuellement de recherche scientifique considérable. Dans le cas de la globalité des traitements, il est exigé qu'un journal d'événements soit complet. Dans le cas de la localité du traitement, la présence complète du journal d'événements n'est pas obligatoire. Ce qui donne une certaine flexibilité au traitement.

Complétude des traces Elle indique la complétude des traces d'exécution via l'identification des instants de départ et de fin de chaque tâche. Une telle complétude facilite l'identification de motifs de comportement, difficilement extractable sans leur présence.

Parallélisme Elle indique si la technique supporte des flux de contrôle concurrentiels.

Reconstruction des boucles Elle indique si l'approche extrait les transitions cycliques. Des boucles particulières peuvent être composées d'une transition cyclique de deux activités.

Bruit Elle indique la robustesse à la présence d'erreurs dans les traces d'exécutions.

Temps Elle calcule des indicateurs temporels, comme le temps d'exécution des instances d'exécutions, sur la base d'un journal d'événements enrichi. Le journal d'événements est enrichi d'informations sur les temps d'exécution des tâches, les temps d'attente/synchronisation, les taux de chargements, etc.

En conclusion, notre approche, concrétisée via un outil appelé Workflow Miner, répond d'une manière compétitive à la majorité des critères de comparaison. Notons l'approche de l'état de l'art la plus proche de la notre est celle de la découverte de workflows, la différence se situe dans le modèle de représentation. Dans notre approche, le modèle de représentation du workflow est basé sur les réseaux de Petri, ce qui est un avantage certain en terme de puissance du formalisme, vérifications, etc. L'approche de découverte de workflows se base sur un formalisme formel, plus spécifique. Par ailleurs, les critères de localités et de prise en compte des bruits sont encore peu étudiés ou étudiées avec peu de validation,dans l'ensemble des approches de l'état de l'art.

2.9 Conclusion

Dans ce chapitre, nous avons présenté les concepts liés à la modélisation et la gestion des processus métiers, ainsi qu'une grande partie de la terminologie qui sera utilisée dans la suite de ce mémoire. La gestion des processus métier implique leur modélisation et leur exécution. La modélisation du processus consiste à décrire ses différents aspects : les tâches à accomplir, l'attribution de ces tâches à des personnes ou à des applications informatiques, les conditions et le flux de données qui circulent entre les tâches.

Nous avons présenté en premier lieu l'approche traditionnelle de conception de processus métiers. Nous montrons que l'approche traditionnelle est caractérisée par la difficulté de prendre en considération les interactions des utilisateurs du système d'une manière automatique et efficace. C'est une faiblesse majeure de l'approche traditionnelle dans le sens où elle n'a pas de moyen systématique de valider et de faire évoluer le système en fonction de l'usage réel du workflow, en l'occurrence par l'exploitation des journaux d'événements pour extraire

Outils	EmiT	Little-Thumb
Référence	[vdAvD02]	[WvdA02]
Représentation	réseaux de Petri	Graphe
Globalité	oui	oui
Complétude des traces	oui	oui
Parallélisme	Oui	Oui
Boucles	Oui	Oui
Bruit	Non	Oui
Temps	Oui	Non

Outils	InWoLvE	Process Miner
Référence	[HK04]	[Sch02]
Représentation	relations de dépendances	blocs
Globalité	oui	oui
Complétude des traces	oui	non
Parallélisme	Oui	Oui
Boucles	Non	Non
Bruit	Oui	Non
Temps	Non	Non

TAB. 2.2 – Synthèse des systèmes de reconstruction.

des connaissances utiles à leur amélioration. Ce qui induit des découvertes tardives des erreurs et dans des conditions hasardeuses, et donc un retour vers la conception forts coûteux, à cause de la distance temporelle qui sépare la conception et la découverte des erreurs.

La conséquence naturelle de cette lacune majeure est l'absence de méthodes de correction permettant de faire évoluer et améliorer le modèle du processus métier. Nous nous situons dans une approche statique dans le sens où l'approche traditionnelle analyse et conçoit un modèle de processus, mais le fait pas évoluer dynamiquement sur la base de l'expérience et les usages du workflow. L'expérience a montré que la conception a priori dans l'approche traditionnelle est rarement parfaite, car on se base sur ce qui devra être réalisé, sans se soucier de la manière avec laquelle les tâches seront réalisées.

Nous avons présenté en second lieu l'approche dynamique basée qui extrait dynamique-

ment des modèles de processus, nécessaire à l'évolution du workflow. L'approche permet de considérer et d'inclure les utilisateurs via les instances d'exécutions dans la boucle dynamique de la vérification, la correction et plus généralement de l'évolution de la conception du workflow. Des recherches sont actives dans le domaine avec cependant très peu de produits commercialisés.

Chapitre 3

Motifs de Workflows

3.1 introduction

Nous présentons ici les motifs des processus métier mises en oeuvre dans notre approche.

3.2 Motifs de base

Dans cette section, nous parlerons des motifs permettant de représenter les aspects élémentaires de la modélisation et l'exécution d'un processus métier. Ces motifs reprennent les définitions fournies par la *WfMC* et complétées pour les besoins de notre travail. La plupart des systèmes de gestion de workflows considèrent ces motifs de base.

3.2.1 Séquence

Description : Une activité dans un workflow est enclenchée après la fin d'une autre activité d'un même processus.

Synonymes : On parle aussi de routine séquentielle et de routine en série.

Exemples :

- L'activité "enclencher le service après vente" est exécutée après l'exécution de l'activité "vendre les biens".

- La demande de remboursement auprès de l'assurance est évaluée après que le fichier client soit recherché.

- L'activité "ajouter les miles" est exécutée après l'exécution de l'activité "réserver le vol".

Implantation : Le motif de séquence est utilisé pour modéliser des étapes consécutives dans un processus de workflow et supporté directement par le système de workflow courant. La réalisation type induit deux activités.

3.2.2 Division parallèle

Description : C'est un point dans le processus de workflow où le processus de contrôle se divise en plusieurs processus de contrôle qui peuvent être exécutés en parallèle. Ce qui permet les activités d'être exécutées simultanément ou dans n'importe quel ordre.

AND-Split est un nœud dans le workflow qui permet de couper une activité en plusieurs activités parallèles qui seront exécutées simultanément.

WfMC constate que dans certains systèmes de workflow, toutes les tâches créées par un AND-Split doivent converger et se rassembler dans un AND-Join commun (structure en blocs). Dans d'autres systèmes de workflow, les tâches engendrées par une division parallèle peuvent se grouper en sous-ensembles pour se rassembler en plusieurs AND-Join (structure en graphe libre).

Il y a *branchement multiple* lorsqu'un itinéraire unique se sépare en deux ou plusieurs itinéraires différents dans le but de réaliser deux ou plusieurs activités en parallèle.

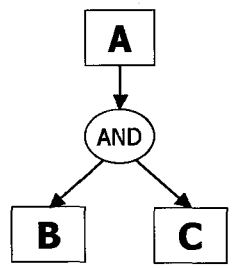

FIG. 3.1 – Représentation du AND-Split

Synonymes : AND-Split, fork, routine parallèle.

Exemples :

- L'exécution de l'activité de paiement enclenche l'exécution les activités "marchandises" et "informer le client".

- Après l'appel à l'assurance, deux tâches parallèles sont enclenchés : la première pour vérifier la police d'assurance du client et la seconde pour enregistrer les dommages présents.

Réalisation : Tous les moteurs de workflow connus supportent ces motifs, en considérant généralement deux approches élémentaires : AND-Split explicite et AND-Split implicite. Les moteurs de Workflow qui supportent le AND-Split explicite, comme le Visual WorkFlow, définissent un point de contrôle en liaison avec plusieurs transitions de sorties. Les transitions de sortie seront enclenchées dès que le point de contrôle est enclenché.

Les moteurs de Workflow qui supportent AND-split implicite, comme le workflow MQ-Series, ne produisent pas des routines spéciales de construction. Chaque activité peut avoir plusieurs transition et chaque transition lui est associée des conditions. Pour permettre une execution parallèle, le concepteur du workflow doit s'assurer que les conditions multiples associées aux transitions de sorties du noeud soient évaluées à "vrai".

3.2.3 Synchronisation

Description : C'est un point dans le processus du workflow où plusieurs activités parallèles convergent vers un point de contrôle, permettant ainsi la synchronisation de plusieurs activités. Nous supposons que chaque branche entrante au point de synchronisation est exécutée une seule fois. Si ce n'est pas le cas, nous parlerons du motif : instances multiples avec synchronisation.

Il y a *rendez-vous* (AND-Join) lorsque deux ou plusieurs activités parallèles convergent vers un itinéraire unique et que l'on assure la synchronisation des itinéraires, c'est-à-dire qu'on ne passera à l'activité suivante que lorsque toutes les activités parallèles seront achevées.

Le *Rendez-vous* de deux tâches parallèles B et C pour lancer une tâches A et est globalement schématise par le graphe présenté dans la figure 3.2

Synonymes : AND-Join, rendez-vous, synchronizer.

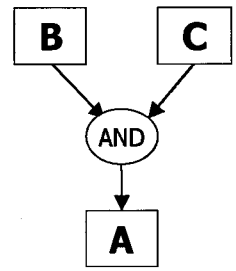

FIG. 3.2 – Représentation du AND-Split

Exemples :

- L'activité d'archivage est enclenchée après l'achèvement des tâches envoi des tickets et réception des paiements.

- Les appels à l'assurance sont évalués après que la police d'assurance soit vérifiée et que le dommage soit enregistré.

Réalisation : La majorité des moteurs de workflow disponibles supportent la construction de ce motif. Comme pour le motif "division parallèle", nous distinguons deux approches élémentaires : le AND-join explicite, réalisé dans des systèmes comme Visual WorkFlow et Synchronizer in Verve, où le contrôle est explicite, via des routines ; le AND-join implicite, réalisé dans des systèmes comme MQSeries et Forté Conductor, qui joint dans une seule activité plusieurs transitions d'entrées.

3.2.4 Choix exclusif

Il s'agit de la description d'un point de contrôle dans le processus de workflow à partir duquel plusieurs branches sont possibles.

On parle d'*aiguillage* (XOR-Split) lorsqu'un itinéraire s'ouvre sur plusieurs itinéraires possibles et que le cas d'exécution suit l'une ou l'autre de ces itinéraires, selon les conditions de transition.

L'*aiguillage* vers une des deux tâches B et C après l'achèvement de l'exécution d'une

tâches *A* et est globalement schématise par le graphe présenté dans la figure 3.3

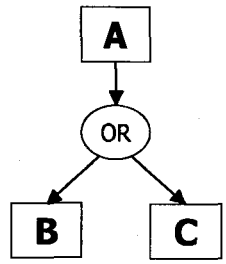

FIG. 3.3 – Représentation du OR-Split

Synonymes : aiguillage, XOR-Split, routine conditionnelle, decision switch, .

Exemples :

- Une activité d'évaluation d'une réclamation est suivie par le paiement du dommage ou (exclusif) par le contact du client.

- Une déclaration de taxe est soit vérifiée (checked) en utilisant une simple procédure administrative, soit rigoureusement vérifiée par un employé expérimenté (senior).

Réalisation : De même que la "division parallèle", il y plusieurs stratégies élémentaires. Certains moteurs produisent une construction explicite pour la réalisation du motif "choix exclusive" (ex. Staffware, Visual WorkFlow). D'autres moteurs de workflow (ex. MQSeries Workflow, Verve) permettent au concepteur démuler l'exclusivité du choix en spécifiant les conditions de transition exclusive.

3.2.5 Fusion simple - Simple Merge

Description : Il s'agit d'un point dans le processus de workflow où deux ou plusieurs branches alternatives viennent ensemble sans synchronisation. Nous partons de l'hypothèse que le motif en question ne supporte pas des exécutions en parallèle. Si c'est le cas, alors on parle de motif "fusion multiple" ou motif "descriminateur".

Il y a *jonction*lorsque deux ou plusieurs itinéraires convergent vers une même activité. Il ne s'agit pas de la synchronisation de plusieurs itinéraires, mais plutôt de la jonction de plusieurs

itinéraires alternatifs.

La *jonction* de deux tâches *B* et *C* pour lancer une tâches *A* et est globalement schématise par le graphe présenté dans la figure 3.4

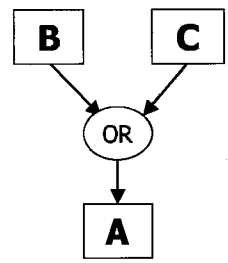

FIG. 3.4 – Représentation du OR-Join

Synonyms : OR-joint, Join asynchronous, fusion.

Exemples :

- L'accès aux archives suite à une réclamation du client est enclenchée après le paiement du dommage ou après avoir contacté le client.

- Après la réception du paiement ou que le crédit soint contractualisé, la voiture est délivrée au client.

Réalisation : Nous supposons qu'il n'y a pas d'exécutions parallèles de processus alternatives. La majorité des moteurs de workflow réalisent des constructeurs qui permettent d'implanter la fusion simple. Il existe certains langages qui imposent un certain niveau de structuration pour garantir automatiquement l'absence de processus alternatives en exécution parallèle. Visual WorkFlow, par exemple, exige le constructeur de fusion d'être toujours précédé par un constructeur de choix exclusif, combiné éventuellement avec d'autres besoins pour aboutir au comportement désiré. Dans d'autres langages de workflow, les concepteurs eux mêmes sont responsables de la non présence d'exécution parallèle de processus alternatives.

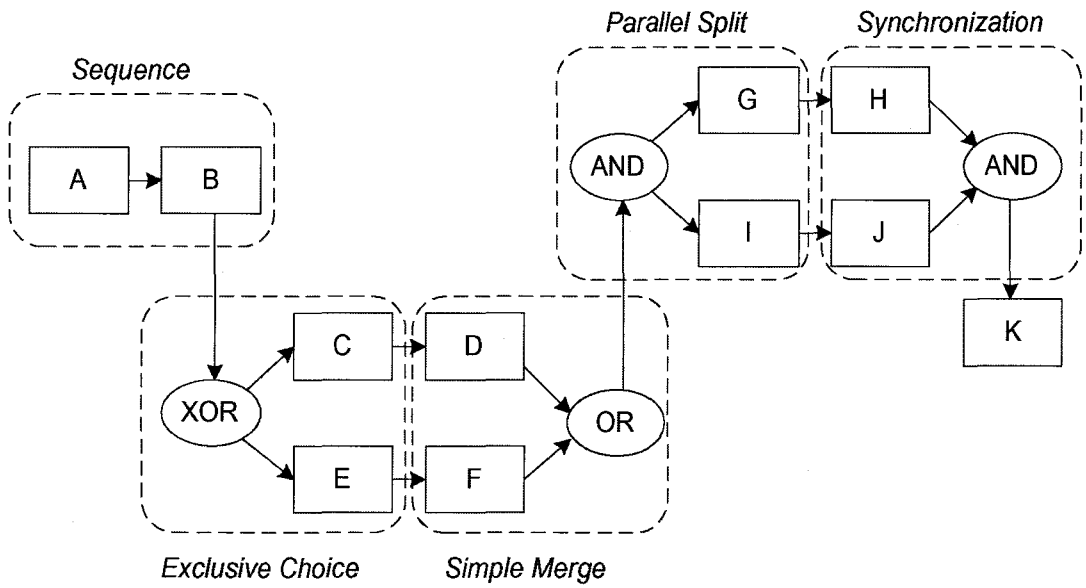

FIG. 3.5 – Exemple.

3.3 Motifs de branchement et de synchronisation

Dans cette section, nous nous focalisons sur des motifs plus sophistiqués pour le branchement et la synchronisation. Contrairement aux motifs présentés dans la sections précédente, la majorité des moteurs de workflow ne les réalisent pas. Ces motifs sont, pourtant, très présents dans les applications finales et plus précisément dans les scénarios affaires du monde réel.

Le motif "choix exclusif" suppose qu'une seule alternative parmi plusieurs est sélectionnée et exécutée. Elle correspond au ou-exclusif (OR-exclusive). quelques fois, il est utile de déployer un constructeur qui peut choisir des alternatives multiples à partir d'un ensemble donné d'alternatives. En conséquence, nous introduisons le choix-multiple.

3.3.1 Multi-choice

Description : Il s'agit d'un point dans le processus du workflow où plusieurs branches sont choisies, sur la base d'une décision ou sur la base de données de contrôle de workflow.

Synonymes : Routine conditionnelle, sélection, OR-Split.

Exemples :

- Après l'exécution de l'activité "évaluation du dommage", l'activité "contacter le département des pompiers" ou l'activité "contacter la compagnie d'assurance" est exécutée.

Réalisation :

Au moins une de ces deux activités est exécutée. Cependant, il est aussi possible que les deux activités soient exécutées.

Le problème dans beaucoup de systèmes de gestion de workflow peuvent spécifier des conditions dans les transitions. Dans ces systèmes, le motif du choix multiple peut être implanté directement. Cependant, il existe des systèmes de gestion de workflow qui n'offrent pas la possibilité de spécifier des conditions sur les transitions et qui offrent seulement des pures constructeurs AND-split et XOR-split (ex. Staffware).

- Comme mentionné précédement, pour les langages de workflow qui assignent des conditions de transition (ex. VErve, MQSeries Workflow, Forté Conductor), l'implantation du choix multiple est simple. Le concepteur du workflow spécifie simplement les conditions désirées pour chaque transition. Nous pouvons noter que le motif du choix multiple généralise la division parallèle (parallal split) et le choix exclusif (exclusive choice).

- Pour les langages qui implantent la division parallèle (parallel split) et le choix exclusif (exclusive choice), l'implantation du choix multiple est faite à travers la combinaison des deux motifs. Chaque branche possible est précédée par XOR-Split qui décide, sur la base des données de contrôle, soit pour activer la branche ou pour pa contourner. Tous les XOR-Splits sont activés par un AND-split.

- Une solution similaire au cas précédent est obtenue en inversant l'ordre du motif de division parallèle et le motif du choix exclusif. Pour chaque ensemble de branches qui peuvent être activées en parallèle, un AND-split est ajouté. Tous les AND-splits sont précédés par un XOR-split qui active le AND-split approprié. Notons que cette solution peut induire la

spécification de workflow plus compacte. Les deux solutions sont mises en avant dans la figure ci-dessous.

3.3.2 Synchronisation de la fusion

Description :

Il s'agit d'un point dans le processus de workflow où plusieurs chemins convergent vers un seul flot de contrôle. Si plusieurs chemins sont pris en compte, la synchronisation des flots de contrôle actifs doit être prise. Si seulement un chemin est pris en compte, les branches alternatives doivent converger sans synchronisation. L'hypthèse du motif est de considérer qu'une branche déjà activée ne peut pas être activée une seconde fois tant que la fusion est en attente de complétude d'autres branchements. Une trace est considérée comme une séquence d'activités complètes pendant une instance d'un processus d'exécution.

Synonymes : Synchronisation de la jointure.

Exemples :

- Après que l'une ou les deux activités "contacter le département des pompiers" et "contacter la compagnie d'assurance" soient complètes (dépendant du comment elles ont été exécutées), l'activité soumission du rapport d'activités est enclenchée.

Réalisation :

Le problème principal de ce motif est de décider quand synchroniser et quand fusionner. Généralement, ce type de fusion a besoin d'une certaine capacité de déterminer le moment d'activation à partir de certaines de ses branches.

- Le seul workflow, cité dans cette thèse, qui implante le motif est MQSeries. Comme noté précédemment, si la synchronisation de la fusion suit OR-split et plusieurs transitions de sortie du OR-plit peuvent être enclenchées, nous ne pouvons dire si le synchronisation est mise en oeuvre, que lors de l'excécution réelle (run time). Le workflow MQSeries contourne ce problème en mettant un token flase à chaque transition qui évalue à faux et à true token pour chaque transition qui évalue vraie. La fusion va attendre jusqu'à la réception des tokens à partir de chaque transition en entrée.

- Dans d'autres moteurs de workflow, la réalisation de la synchronisation de fusion n'est

pas triviale. La seule solution possible serait d'éviter l'usage explicite du OR-split qui enclenche plusieurs transitions de sorties et de l'implanter comme une combinaison de AND-splits et XOR-splits (voir motif choix multiple). De cette manière, nous pouvons facilement synchroniser des branches correspondantes en utilisant AND-join et des constructeurs de fusion standards.

3.3.3 Multi-fusion / Multi-merge

Description : Il s'agit d'un point dans un processus de workflow où deux ou plusieurs branches reconvergent sans aucune synchronisation. Plusieurs branches sont activées, peut être même en concurrence. L'activité qui suit la multi-fusion débute chaque activation et chaque branchement en entrée.

Synonymes : Synchronisation de la jointure.

Exemples :

Quelques fois deux ou plus branches parallèles partagent la même fin. Pour réaliser un tel motif, potentiellement compliqué, alors dans chaque branche l'opérateur multi-fusion peut être utilisé. C'est le cas par exemple de deux activités, à savoir une application d'Audit et une application (cible de l'Audit) exécutée en parallèle. Les deux activités doivent être suivies par une activité de fermeture. L'application d'audit et l'application concernée partagent la même fin. L'utilisation d'un constructeur de fusion standard, comme fournit par certains produits de workflow, pour implanter ce motif entraîne souvent des résultats indésirables. Certains produits (ex. Staffware, i-Flow) ne génèrent pas une seconde instance d'une activité si une autre instance de la même activité est en cours d'exécution. En fin, dans certains produits de workflow (ex. Visual WorkFlo, SAP R/3) il n'est pas possible d'utiliser le constructeur de fusion en conjonction avec une division parallèle (parallel split) comme dans le workflow de la figure.

Réalisation :

- Le constructeur de fusion du workflow Verve et Forté peuvent être utilisés pour implanter ce motif.

- Si la multi-fusion n'est pas spécifiée explicitement, la conception du motif pour les langages qui ne sont pas capables de créer plus d'une instance active à la fois est répliquer l'ac-

tivité dans le modèle de workflow comme présenté dans la figure ci-dessous. Avec la multi-fusion, le nombre d'instances d'une activité n'est pas connu lors de la conception, mais plutôt lors de l'exécution. Même si nous n'avons pas spécifié une notation graphique explicite pour la multi-fusion, cela ne devra pas causer des des ambiguïtés avec les notations graphiques simple-fusion ou or-jointure qui sont utilisés dans un contexte différent.

Boucles Une ou plusieurs tâches peuvent être répétées. Nous utilisons aussi le terme de cycle ou itération

Synonyms : Loop, iteration, cycle.

Exemple : La tâche de vérification du stock est répétée jusqu'à ce que toutes les mises à jour soient terminées. Avant le traitement d'une commande, la saisie de la commande est répétée jusqu'à la validation de la saisie.

Implementation :

Certains moteurs de workflow ne permettent pas les boucles arbitraires. Ils permettent cependant des cycles structurées (e.g. MQSeries, Visual WorkFlo et SAP R/3 Workflow). Les boucles sont converties en boucles structurées, lorsque les instanciations multiples sont permises.

Beaucoup de produits de workflow ne l'implantent pas, car on estime qu'une boucle n'envisage pas la fin d'un processus après un nombre d'étapes fini. Or, nous estimons dans les processus métier que toute tâche avoir une fin.

3.4 Conclusion

Nous venons de présenter deux catégories de motifs de workflow. La première est relativement bien prise en compte par la plupart des systèmes de gestion de workflow présents sur le marché (séquence, choix exclusif, division parallèle, synchronisation, jointure). La deuxième catégorie de motifs est moins prise en compte par les systèmes présents sur le marché, comme le choix multiple (OR-Split) ou la multifusion.

Dans notre approche nous avons pris en compte la première catégorie de motifs car elle correspond à la réalité des processus actuels. Rappelons que notre objectif est d'extraire les modèles de processus, donc des modèles conformes aux motifs les plus utilisés et les plus

fréquents dans les processus métiers actuels.

LA figure illustre la corrélation entre les motifs. Le choix exclusif fonctionne généralement avec la fusion simple. Le parallélisme est suivie généralement d'une synchronisation. Cela ne signifie pas qu'il n'y a pas d'autres motifs, mais juste que ce sont les corrélations les plus fréquentes et les plus connues.

Chapitre 4

Extraction et adaptation des modèles de processus

4.1 Introduction

Dans ce chapitre, nous décrirons notre approche d'extraction de modèles de processus ainsi que sa réalisation. Nous montrerons comment elle améliore le comportement initial du workflow.

L'approche est ainsi résumé par les étapes suivantes :

- Collecte des traces d'exécutions du processus : Cette étape consiste à collecter les traces d'exécutions des instances du processus. La structure des traces d'exécutions doit être suffisamment riche pour supporter les méthodes d'extraction des motifs fréquents.
- Extraction du modèle processus : Le but de cette phase est d'extraire le modèle de processus, à partir du journal d'événements qui garde la trace des instances d'exécutions. Les motifs de processus et les dépendances entre les tâches extraits contribuent à générer un modèle de processus.
- Evolution du processus : À partir du modèle de processus extrait, lors de l'étape précédente, l'approche utilise au mieux des recommendations définies par les concepteurs du processus en relation avec le processus métier. Ces recommendations servent à faire

évoluer le processus initial en fonction du modèle de workflow généré.

Instance	Tâche	Fin de la tâche
1	A	09 : 20
1	B	10 : 30
2	A	10 : 40
1	C	11 : 00
1	D	14 : 00
2	B	14 : 10
2	C	14 : 20
3	E	14 : 40
4	E	15 : 30
2	D	15 : 40
3	E	16 : 00
4	F	16 : 30
3	F	17 : 00

Tâche	Durée Min	Durée Max
A	00 : 30	01 : 00
B	00 : 45	01 : 00
C	00 : 30	03 : 00
D	00 : 15	02 : 00
E	00 : 30	01 : 30
F	00 : 45	01 : 30

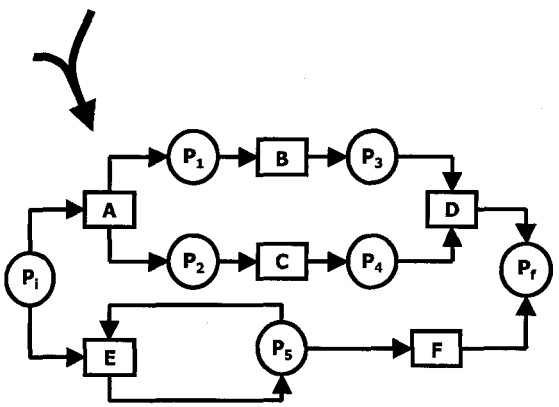

FIG. 4.1 – Extraction d'un modèle de processus à partir des traces d'instances.

Pour extraire le modèle de processus (motifs, dépendances), à partir du journal des événements, notre approche opère en deux étapes : La première consiste à analyser statistiquement les événements du journal qui correspondent aux traces des instances d'exécutions. Cette analyse détermine la table des tables de dépendances statistiques. La deuxième étape consiste à spécifier statistiquement les propriétés du modèle de workflow sous forme de règles à associer à la table des dépendances statistiques. Le schéma de la figure 4.2 illustre avec plus de détails notre approche. Nous avons en entrée un journal d'évènements composé d'un flux de tâches, chaque tâche est caractérisé par son instance et l'instant de sa fin d'exécution. Le journal d'évènements est enrichi par le tableau qui définit, pour chaque tâche, la durée minimale et la durée maximale pour son exécution. Le réseau de Petri présenté dans le schéma est extrait à partir de ces données. Il représente le modèle de processus équivalent au journal d'évène-

ments. Les tâches sont représentées par des transitions dans le réseau de Petri. Par exemple, la présence deux transitions A et E en sortie de la place initiale P_i signifie que toutes les instances du journal d'évènements commencent par une des tâches A ou E. La présence de deux place P_1 et P_2 en sortie de la transition A, et qui vont respectivement vers les transitions B et C, signifie que la fin d'exécution de la tâche A lance l'exécution de B et C en parallèle. L'algorithme de construction est détaillé par la suite.

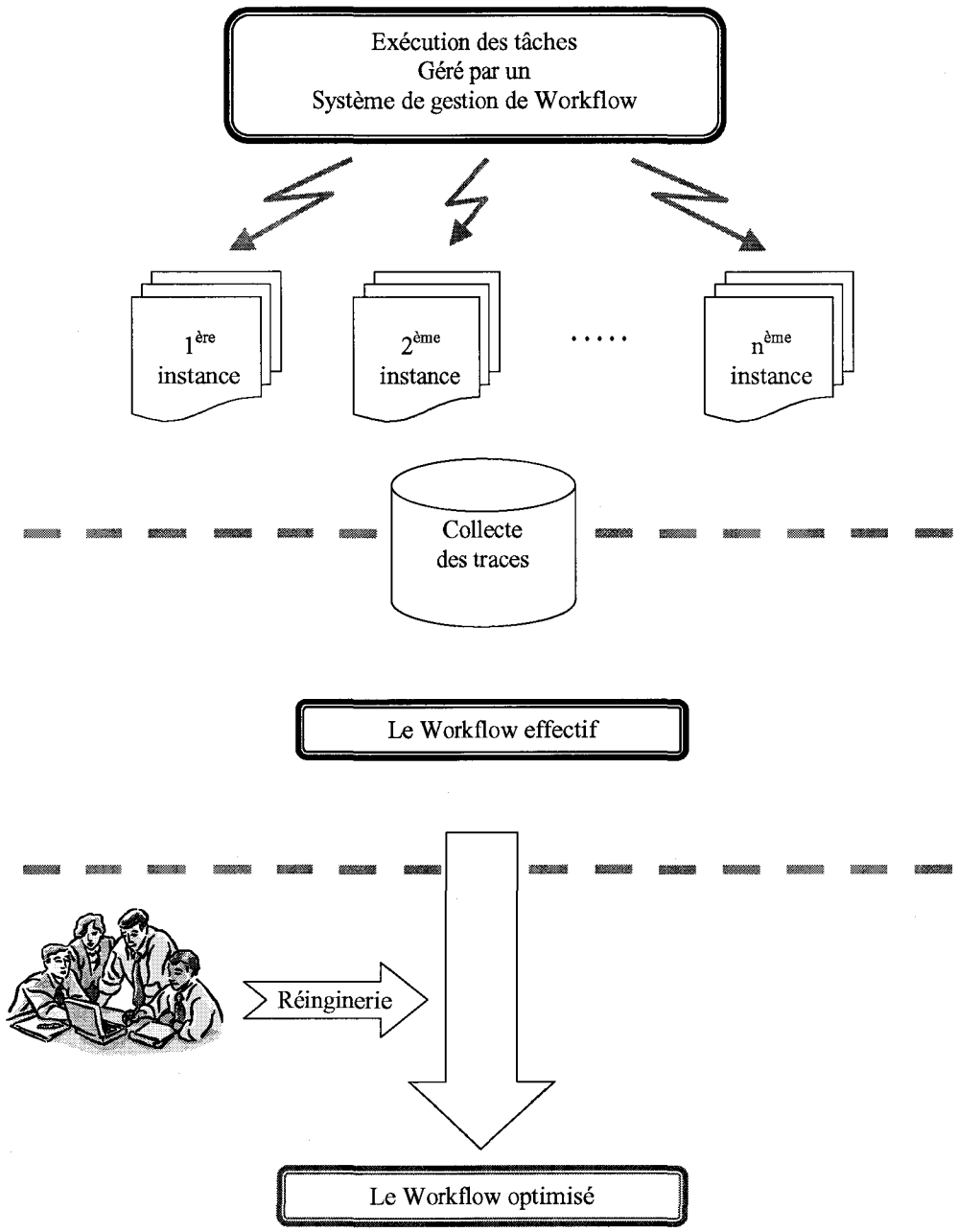

FIG. 4.2 – Schéma général

Ce chapitre est structuré comme suit. D'abord, nous présentons les concepts de notre approche, à savoir les réseaux de Petri et les réseaux WF-nets (réseaux de workflow). Ensuite, nous mettons l'accent sur le journal des événements sur lequel se basera notre approche pour extraire les modèles de processus. Nous verrons les règles de construction, à savoir la méthode avec laquelle les relations temporelles de type succession, croisement et boucles peuvent être déduites à partir du journal des événements. Nous présentons par la suite l'algorithme de construction qui se basera à la fois sur des analyses statistiques et sur les règles de construction. Sur la base des résultats obtenus par le modèle de processus, nous proposons des recommendations d'évolution du workflow initial. En fin, nous terminons notre chapitre par une conclusion qui reprend les points importants de notre approche.

Avant de présenter l'approche d'extraction de modèles de workflow, nous introduirons la classe des WF-nets (pour Workflow Nets) qui est une sous-classe des réseaux de Petri classiques et qui est optimisée pour la modélisation des processus. À noter que ces réseaux sont conçus pour la représentation du comportement dynamique d'une seule instance à la fois.

La figure 4.3 reprend l'exemple de la modélisation d'un processus de gestion de commandes, de la réception de la commande de la part d'un client à la livraison ou le refus. Rappelons les grandes lignes de ce workflow :

D'abord, le client spécifie la commande et les termes de la commande : quantité, référence, date de livraison souhaitée, etc. (*Tâche_SC*). Alors une instance de workflow est lancée pour la collecte d'information sur le client et évalue son degré de solvabilité par la tâche de vérification de la solvabilité du client (*Tâche_VSC*).

Après cela, l'entreprise prend sa décision en choisissant exclusivement entre trois possibilités : Si le client est solvable, la commande est acceptée. Si le client est suffisamment important (chiffre d'affaires important, notoriété sur le marché, taux de dettes acceptables par rapport au chiffre d'affaires, etc.) pour lui accorder la commande, sans évaluation de risque, alors l'entreprise approuve cette commande par la tâche d'accord exceptionnel (*Tâche_AE*). Dans les autres cas de risques, le fournisseur rejette la commande (*Tâche_RC*).

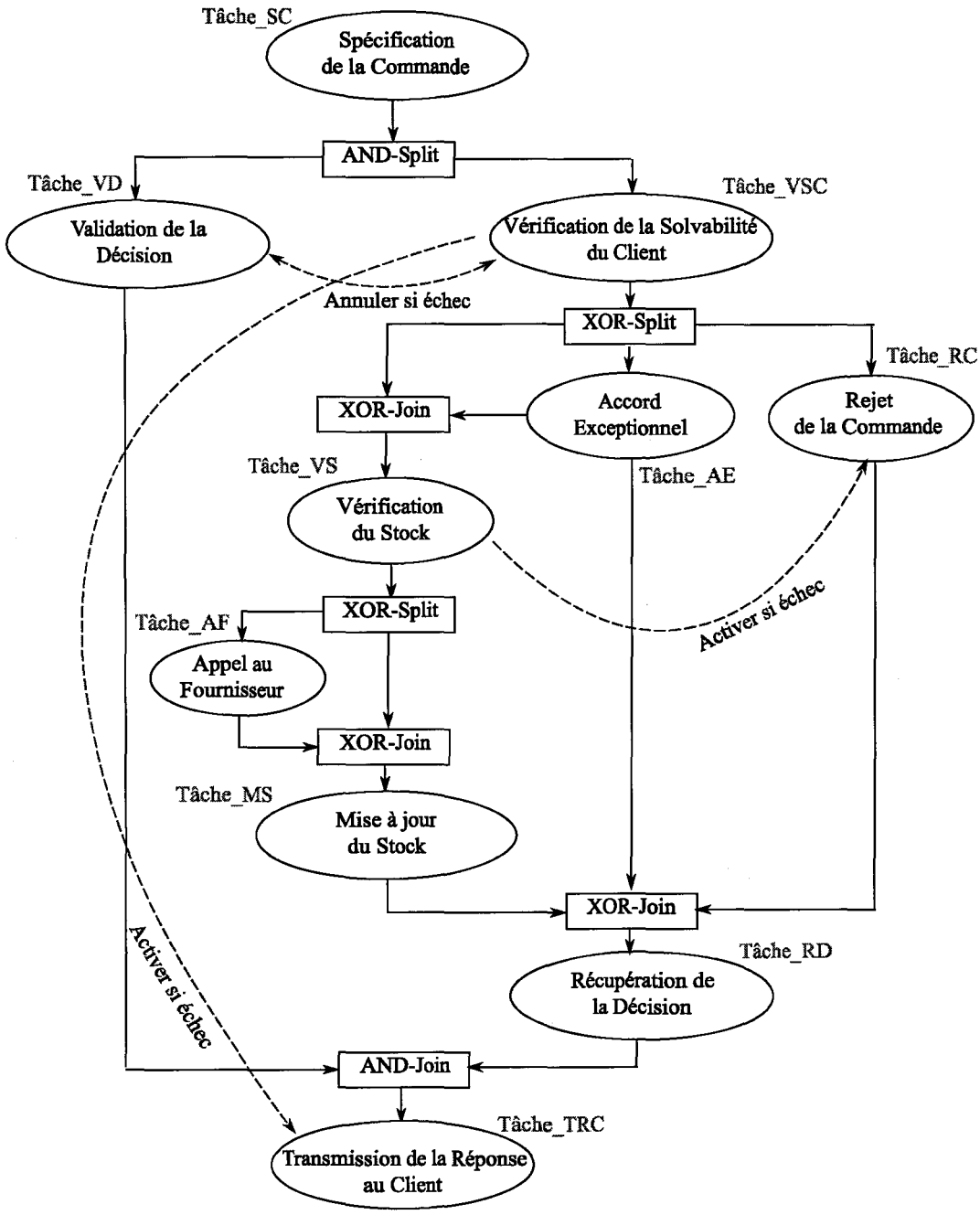

FIG. 4.3 – Exemple de la modélisation d'un processus de gestion de commandes.

Pour les commandes acceptées, le fournisseur procède à la vérification du stock (*Tâche_VS*) et fait appel au fournisseur (*Tâche_AF*) quand il y'a une rupture de stock. dans les deux cas le stock est mis à jour (*Tâche_MS*).

Par la suite, la tâche de récupération de la décision (*Tâche_RD*) rédige et enregistre la décision d'acceptation et de traitement ou de rejet de la commande dans la base de données. La transmission de la réponse au client (livraison de la marchandise au client, accompagnée de la facture ou avis de rejet) est réalisée par la tâche de transmission de la réponse au client (*Tâche_TRC*).

Le traitement de la commande ne peut pas être fait sans approbation d'un contrôleur (direction commerciale de l'entreprise). En effet, une activité humaine de validation de la décision (*Tâche_VD*) est nécessaire au processus de décision. Ainsi, il n'est possible de répondre à la commande que si la direction commerciale de l'entreprise la valide. Elle peut rejeter la commande même si une décision positive a été prise. Le contrôleur peut donner librement sa décision à tout moment pendant le processus de traitement de la commande.

Initialement, pour traiter les échecs d'exécution des tâches composantes du workflow, les concepteurs spécifient en plus des tâches, des procédures de normalisation qui décrivent les relations exceptionnelles entre les tâches. Dans notre exemple, il est indiqué que si *Tâche_VD* échoue, on annule aussi l'exécution de *Tâche_VSC* et le traitement de la commande sera re-exécutée jusqu'au succès. En plus, en cas d'échec de *Tâche_VS*, le workflow continue l'exécution par une décision de rejet de la commande *Tâche_RC*. Finalement, les procédures de traitement d'échecs ne sont pas fournies pour les autres tâches qui sont supposées se dérouler sans échec.

4.2 Les raisons d'usage des réseaux de petri

Sémantique Formelle Un processus spécifié sous forme de réseau de Pétri a une définition claire et précise car les réseaux de Pétri classiques, ainsi que des améliorations (couleurs, temps, etc.) sont bien définis formellement.

De nature graphique Les réseaux de Pétri sont de nature graphique, ils sont donc intuitifs et faciles à apprendre. Cette nature graphique permet aussi une meilleure communication

avec les utilisateurs finaux.

Capacité d'expression Les réseaux de Pétri permettent de représenter toutes les primitives de base nécessaires à la modélisation d'un processus, ainsi que les systèmes de routage présents dans les Systèmes de Workflow actuels.

Propriétés Durant les trois dernières décennies, beaucoup de chercheurs ont étudié les propriétés de base des réseaux de Pétri. Les mathématiques de base tiennent compte des réflexions autour de ces propriétés. En conséquence, il y a beaucoup connaissances et de savoir-faire partagés, sous forme de livres ou d'articles, concernant cette technique de modélisation.

Outils d'analyse Les réseaux de Pétri bénéficient d'une large collections d'outils et de techniques qui sont en leur faveur pour leur usage dans la modélisation et l'analyse des processus. Ces techniques peuvent être utilisées pour montrer certaines propriétés (terminaison, boucles infinies, etc.) ou calculer certaines métriques (temps de réponse, délais d'attente, pourcentage de ressources utilisées, etc.). Cela permet aussi d'évaluer les autres systèmes de modélisation en utilisant les systèmes basés sur les Réseaux de Pétri Standards.

Libre de droits Les réseaux de Pétri représentent un environnement libre et indépendant pour la modélisation et l'analyse des processus. Ils ne dépendent ni d'éditeurs spécifiques ni du système de versions.

4.3 Notions de base

4.3.1 Définition d'un workflow

Un workflow est "l'automatisation d'un processus, totale ou partielle, au cours duquel des documents, des informations et des tâches passent d'un participant à un autre, au sein d'un groupe de travail, en conformité avec un ensemble de règles prédéfinies" [WfM99]. Un système de workflow définit, crée et gère l'exécution des processus.

Un schéma de workflow WS est un n-uplet :

$< A; E; a_0; F; IN; OUT_{min}; OUT_{max} >$:

- A : Un ensemble fini d'activités.
- $a0$: Activité initiale.
- F : Ensemble des activités finales.
- $E \subseteq (A-F) \times (A-\{a_0\})$: Une relation non cyclique de précédence entre les activités.
- IN ; OUT_{min} et $OUTmax$: sont trois fonctions assignant une valeur naturelle à chaque nœud :
  - $\forall a \in A-\{a_0\} : 0 < IN(a) < InDegree(a)$.
  - $\forall a \in A-F : 0 < OUT_{min}(a) < OUT_{max}(a) < OutDegree(a)$.
  - $IN(a_0) = 0$ et $\forall a \in F : OUTmin(a) = OUTmax(a) = 0$.

En résumé, une activité a ne peut commencer son exécution que si au moins $IN(a)$ des taches qui la précèdent soient terminées.

4.3.2 Réseaux de Petri

Un réseau de Petri (*Petri net*) est un graphe biparti constitué de places, de transitions, et d'arcs orientés qui relient les places aux transitions et les transitions aux places, il est représenté par un triplet $R = (P;T;F)$:

P : Ensemble fini de places $P = \{P_1, P_2, P_3, ..., P_m\}$.
T : Ensemble fini de transitions $T = \{T_1, T_2, T_3, ..., T_n\}$.
$F = Pré \cup Post$: ensemble d'arcs orientés, avec :
$Pré \subseteq P \times T$: ensemble d'arcs directs liant les places aux transitions.
$Post \subseteq T \times P$: ensemble d'arcs directs liant les transitions au places.

TAB. 4.1 – Définition formelle d'un Réseau de Petri.

On notera $\bullet t$ (respectivement $t\bullet$) l'ensemble $\{p \in P/Pre(p,t)\}$ (respectivement $\{p \in P/Post(p,t)\}$) correspondant à l'ensemble des places d'entrée (respectivement de sortie) de la transition t.

De même, on notera $\bullet p$ (respectivement $p\bullet$) l'ensemble $\{t \in T/Pre(p,t)\}$ (respectivement $\{t \in T/Post(p,t)\}$) correspondant à l'ensemble des transitions d'entrée (respectivement de sortie) de la place p.

4.3.3 Réseau de Petri marqué

Un réseau de Petri marqué est un couple $< R, M >$ où :

- R : un réseau de Petri
- M : une application qui associe à chaque place du réseau de Petri un nombre de marques :
  - $M : P \longrightarrow N$
  - $p \longrightarrow M(p)$

Une transition est dite *franchissable* (*validée* ou *tirable*) si :

$$\forall p \in {}^\bullet t : M(p) \geq Pré(p,t)$$

Le *tir* (ou le *franchissement*) d'une transition t à pour conséquences :

- de retirer $Pré(p,t)$ jetons de chaque place d'entrée p de la transition t,
- d'ajouter $Post(p,t)$ jetons à chaque place de sortie p de la transition t.

Le franchissement d'une transition t provoque ainsi le passage d'un marquage M à un marquage M'

$$\forall p \in P : M'(p) = M(p) + Post(p,t) - Pré(p,t)$$

4.3.4 Les Réseaux WF-nets (Workflow Nets)

Nous introduisons dans notre cas la notion de *WF-nets* qui est une sous-classe des réseaux de Petri. Un processus métier est défini par un ensemble de tâches ainsi que les conditions de leur exécution. En utilisant les Réseaux de Petri, ce processus sera représenté en transformant son entrée (initialisation) en une place sans arcs entrants, et sa fin (condition de terminaison) en une place sans arcs sortants. Les conditions seront représentées par des places et les tâches par des transitions. Quand un processus est modélisé sous forme de Réseau de Petri, il doit satisfaire deux conditions :

1. À tout moment et depuis chaque état du réseau, on doit pouvoir atteindre l'état où la place finale est marquée.
2. Quand la place finale est marquée, les autres places ne doivent pas l'être.

Soit $N = (P, T, F)$ un réseau de Petri, et t un élément qui n'appartient pas à T. N est un *WF-net* si et seulement si :

- P contient une place initiale $\bullet i = \varnothing$.
- P contient une place finale $f\bullet = \varnothing$.
- Le réseau $N' = (P, T \cup \{t\}, F \cup \{(f,t),(t,i)\}$ est fortement connexe.

Le réseau présenté dans la figure 4.10 est un *WF-net*. Ce réseau n'est pas fortement connexe, mais on le court-circuitant avec une transition supplémentaire t qui viendra lier PF à PI, il devient fortement connexe. Cependant même si un réseau remplit les conditions présentées dans la définition précédente, le processus correspondant peut être confronté à des situations d'erreurs ou de blocage ou contenir des transitions par lesquelles le passage est impossible. Une grande partie des langages de modélisation de processus fournissent des structures telles que le *AND-Split* (Branchement multiple), le *AND-Join* (Rendez-vous), le *XOR-Split* (Aiguillage), le *OR-Join* (Jonction). Ces structures sont utilisées pour modéliser les routages parallèles, séquentiels, itératifs ou conditionnels. Les réseaux *WF-nets* permettent le routage des différentes instances d'un processus. Ainsi, les tâches sont représentées par les transitions et les dépendances et conditions par les places et les arcs.

4.4 Les fichiers logs

Dans cette section, nous présenterons des notions liées au concept des workflows et tout particulièrement les traces sauvegardées suite de leur exécution.

4.4.1 Instance d'exécution de processus

L'instance d'exécution d'un processus, ou d'un workflow, est une occurrence de l'exécution de ce processus de son début jusqu'à sa fin. Les différentes instances d'un même processus peuvent être composées de différents sous-ensembles de tâches (elles suivent des chemins différents dans le modèle de ce processus).

Soit T un ensemble de tâches, une instance $\sigma \in T^*$ est une séquence de tâches représentant la chaîne de traitement sur une seule entité.

Dans notre exemple de processus de traitement de commandes, la séquence { *Tâche_SC*, *Tâche_VSC*, *Tâche_VS*, *Tâche_AF*, *Tâche_MS*, *Tâche_RD*, *Tâche_VD*, *Tâche_TRC* } représente une seule instance.

4.4.2 Classe d'instances

Un modèle de processus pouvant être représenté par un graphe où les nœuds sont les tâches et les arcs représentent l'ordre d'exécution. Une classe d'instances est un ensemble d'instances représenté par le même graphe partiel. On peut ainsi séparer les classes suivant différents critères tels que : la nature du chemin emprunté, les applications concernées, les durées d'exécution des tâches, etc. L'identification de ces classes d'instances aura comme but :

- Analyser en temps réel des flux d'instances d'exécution des processus afin d'identifier les différentes classes ou les exceptions si c'est le cas.

- Visualiser et exploiter les connaissances acquises sur les différentes classes identifiées et les intégrer dans le processus d'aide à la décision.

4.4.3 Journal d'événements

Un journal d'événements $W \subset T*$ est un ensemble d'instances. Le tableau gauche dans la figure 4.3 est un exemple de journal d'événements.

4.4.3.1 Format des journaux d'événements

Nous présentons ci-dessous le format général des journaux d'événements. Il s'agit d'un format cohérent avec la majorité des systèmes de gestion de workflow. En principe, n'importe quel système qui collecte des événements liés à l'exécution de ses tâches peut utiliser ce format pour stocker et échanger les traces d'exécutions. Ce format est le seul paramètre en entrée des outils d'analyse et d'extraction de modèles de processus que nous présentons dans les sections suivantes. L'intérêt d'utiliser un format est de simplifier, voire réduire l'effort de traitement et de permettre l'usage de méthodes variées d'extraction des modèles de processus.

La trace d'exécution d'un processus de workflow est l'ensemble des traces d'exécutions de ses instances d'exécution. Chaque instance d'exécution capture des événements atomiques représentant des changements d'états de ses activités suite à l'exécution d'une action. Chaque événement est décrit par l'identifiant unique de la tâche, l'instant (la date et l'heure) de production de l'événement. L'ordre des événements dans une même instance d'exécution est important tandis que l'ordre des différentes instances d'exécution est sans importance.

Le Tableau 4.2 présente une petite partie d'un fichier de traces d'événements au sein de la société *Sneda.* chaque ligne représente l'achèvement de l'exécution d'une tâche au sein de cette entreprise. La première ligne, par exemple, signifie qu'une mise à jour de groupe "MAJGRP" a été effectuée par Françoise O. "fro" le "23/05/2006". Les autres colonnes sont des paramètres liés à la tâche et les services qui la gèrent.

10 0 835936 0 23/05/06 58440 0 "fro" "MAJGRP" 0 "E2P2" "0x021400513631000000000000"
10 0 836000 0 23/05/06 52237 0 "nai" "CRTPIE" 0 "E2P2" "0x021400fa383536320000000"
10 0 836001 0 23/05/06 52238 0 "np8" "MAJLOC" 0 "E2P2" "0x021400283239313731390000"
10 0 836002 0 23/05/06 58467 0 "brn" "MAJLOC" 0 "E2P2" "0x021400273239363730370000"
10 0 836003 0 23/05/06 58471 0 "brn" "MAJLOC" 0 "E2P2" "0x021400273239363730370000"
10 0 837852 0 23/05/06 58620 0 "fro" "MAJGRP" 0 "E2P2" "0x021400513631000000000000"
10 0 837854 0 23/05/06 48443 0 "cad" "MAJGRP" 0 "E2P2" "0x021400523137393835320000"
10 0 842570 0 23/05/06 54421 0 "sne" "MAJGRP" 0 "E2P2" "0x021400513200000000000000"
10 0 842573 0 23/05/06 45410 0 "frr" "MAJLOC" 0 "E2P2" "0x021400273232353931350000"
10 0 842574 0 23/05/06 64463 0 "beo" "MAJLOC" 0 "E2P2" "0x021400273333310000000000"
10 0 842575 0 23/05/06 51283 0 "frr" "MAJLOC" 0 "E2P2" "0x021400273232353931350000"
10 0 858892 0 23/05/06 62444 0 "brn" "MAJLOC" 0 "E2P2" "0x021400273100000000000000"
10 0 858909 0 23/05/06 62445 0 "brn" "MAJLOC" 0 "E2P2" "0x021400273100000000000000"
10 0 873376 0 23/05/06 63393 0 "fro" "MAJLOC" 0 "E2P2" "0x021400273239313939310000"
10 0 875501 0 23/05/06 63395 0 "fro" "MAJLOC" 0 "E2P2" "0x021400273239313939310000"
10 0 880889 0 23/05/06 41455 0 "anm" "MAJGRP" 0 "E2P2" "0x021400523200000000000000"
10 0 893673 0 23/05/06 61777 0 "gag" "MAJLOC" 0 "E2P2" "0x021400273239323033330000"
10 0 893674 0 23/05/06 62186 0 "gag" "MAJLOC" 0 "E2P2" "0x021400273239323033340000"

TAB. 4.2 – Exemple de trace d'exécution d'un processus.

Définition d'un événement : Soit T l'ensemble des tâches appartenant à un workflow. Un

événement est défini comme un couple *Event* = (identité de l'instance, identité de la tâche, instant de fin la tâche) où : l'identité de l'instance représente un identificateur de l'entité traitée tout au long du processus, l'identité de la tâche concerne l'identifiant de la tâche concernée par l'événement ; l'instant de la tâche concerne l'instant de fin d'exécution de la tâche.

Certains systèmes de collecte d'événements proposent un niveau de granularité plus fin, comme les états possibles d'une activité au sein d'une instance de workflow : *initial*, *suspendu*, *activé*, *terminé*, *échoué* et *annulé*. Ces états couvrent l'essentiel des états présents dans ces systèmes de collecte de traces d'instances d'exécutions. Précisons qu'ils ne sont pas tous disponibles dans ces systèmes. D'autres ne collectent qu'une partie des événements, par exemple le début d'une tâche d'une instance de processus. En plus, la nomenclature est généralement différente d'un système à un autre. Il est donc évident que pour que le domaine puisse progresser, il faut normaliser la représentation des journaux d'événements pour permettre à un plus grand nombre d'outils d'extraction de modèles de s'appliquer à n'importe quel système de workflow. Un tel besoin dépasse largement le cadre de ce travail.

Les traces d'exécutions d'instances de processus de workflow peuvent contenir des centaines, voire des milliers de traces d'exécutions de ces instances. Nous partons de l'hypothèse générale que les instances d'exécution sont indépendantes. Cela dit, nous pensons qu'à terme, les relations de dépendance ou de causalité entre les instances seraient un atout majeur dans la richesse des modèles de processus. Ce point dépasse le cadre actuel de notre travail. Nous supposons dans le cadre de notre travail qu'il n'y a pas dépendances entre les différentes instances d'exécution. Ainsi, nous pouvons représenter les traces d'instances d'exécutions d'un workflow comme un ensemble de séquences distinctes, tel que chaque séquence représente l'exécution d'une instance de workflow.

Le schéma ci-dessous montre qu'un journal d'événements d'un workflow se compose d'un ensemble de flux d'événements. Chaque flux d'événements trace l'exécution d'une instance et se compose d'événements qui capturent le cycle de vie des tâches exécutées dans une instance.

Il est conseillé de s'assurer que le journal des événements et les flux des événements comprennent des événements d'identifiants unique. Le nom de la tâche doit être unique dans le workflow. S'il y a deux tâches ou plus dans un workflow ayant le même nom, nous pouvons penser qu'elles se rapportent à la même activité. Bien que nous supposons qu'une tâche ait un

nom unique, elle peut être présente plusieurs fois dans la même instance (par exemple dans une situation de boucle).

L'expérience montre qu'il est également assez simple d'extraire l'information à partir des systèmes d'information d'entreprise spécifiques et de le traduire au format XML. Le format que nous avons présenté est transcrit au format XML. Il peut être utilisé par les autres outils d'extraction de modèle de processus.

```
<!ELEMENT WorkFlow_log (source?, process+)>
<!ELEMENT source EMPTY>
<!ATTLIST source
          program(staffware|inconcert|pnet|IBM_MQ|other) #REQUIRED>
<!ELEMENT process (case*)>
<!ATTLIST process
          id ID #REQUIRED
          description CDATA 00none00>
<!ELEMENT case (log_line*)>
<!ATTLIST case
          id ID #REQUIRED
          description CDATA 00none00>
<!ELEMENT log_line (task_name, task_instance?, event?, date?, time?)>
<!ELEMENT task_name (#PCDATA)>
<!ELEMENT task_instance (#PCDATA)>
<!ELEMENT event EMPTY>
<!ATTLIST event
          kind (normal|schedule|start|withdraw|suspend|resume|abort|complete) #REQUIRED>
<!ELEMENT date (#PCDATA)>
<!ELEMENT time (#PCDATA)>
```

TAB. 4.3 – Schéma du format XML recommandé pour les fichiers logs

Un flux d'événements représente l'historique d'exécution, en termes d'événements, d'une instance de workflow comme un tuple flux d'événements = (begin, end, séquence, instances) où : "begin" représente l'instant du début de la trace d'exécution de cette instance ; "end" représente l'instant de la fin de la trace d'exécution de cette instance ; "séquence" représente un ensemble ordonné d'événements collectés à l'exécution de cette instance de processus ; "instances" représentent le numéro de la trace d'exécution de cette instance. Précisons que le

"end" représente la fin d'une instance et qu'il faut distinguer de la fin d'une tâche.

Un journal d'événements est un ensemble de flux d'événements. Le journal d'événements = (identité du workflow, flux d'évènements i, $0 < i <$ nombre d'instances du processus) où le flux d'événements représente la trace d'exécution de la i-ème instance du workflow.

4.4.4 Contraintes de mises en oeuvre

Généralement, les systèmes de workflow sauvegardent les traces d'exécutions des instances de processus. Ces traces contiennent souvent des informations de nature temporelle (par exemple, la fin d'une tâche) au niveau de chaque événement. Ces informations sont particulièrement utiles pour déduire des informations additionnelles, comme l'état de l'instance (activé, terminé) et pour détecter explicitement le parallélisme. Les instants de début, la durée maximale, la durée minimale des tâches peuvent également être utilisés pour estimer la durée d'exécution d'une tâche. En plus, un certain nombre de traces d'exécutions contiennent également des informations sur les états de l'activité (e.g. activé, terminé) offrant la possibilité d'analyser le comportement du workflow.

Il existe cependant d'autres systèmes de workflows qui collectent des événements et proposent des traces moins détaillées. Par exemple, les instants de début des tâches ne sont pas sauvegardés ou encore les durées minimales et maximales des tâches ne sont pas spécifiées. Il est clair, que la solution idéale est de sauvegarder un minimum d'informations nécessaire tout en permettant la déduction de toutes les informations utiles à l'extraction des modèles de processus. Cette solution idéale est valable dans la conception d'un nouveau système de workflow. En revanche, s'il est question de réutiliser un système existant, alors il est indispensable de réutiliser au mieux les informations sauvegardées pour extraire le modèle du processus.

Pour la construction du modèle de processus, nous nous intéressons aux instances exécutées avec succès, sans échec ou exception. Ainsi, les traces d'exécutions des instances utilisées pour l'extraction du modèle présentent toutes les tâches qui ont atteint l'état final sans exception ou échec. En conséquence, nous pouvons présenter, pour la découverte du modèle de processus, un format de traces d'exécution d'instances de processus simplifié, où les événements sont atomiques et totalement ordonnés ne contenant que le nom de la tâche et l'identifiant de

son instance. Nous utiliserons aussi les flux d'événements relatifs aux échecs d'exécutions. En fait, ces cas concernent seulement le comportement et les dépendances et seront utilisés par la suite pour la découverte du comportement et des dépendances qui désignent les mécanismes pour le traitement des échecs.

Le but de cette section est de mettre en avant un défi majeur dans la réalisation des journaux d'événements : Pour développer un moteur d'extraction de modèles de processus qui s'applique à un large nombre de moteurs de workflow disponibles sur le marché, il est indispensable de développer un moteur d'extraction qui prend en compte un minimum d'information du journal des événements. Ce minimum d'information, comme décrit précédemment, est composé de l'identité de la tâche et son instant de fin.

Les traces d'exécutions initialement collectées, dans les workflows cibles, contiennent des informations diverses, dont des informations sur les instances d'exécution exécutées avec échec. Pour l'extraction du modèle de processus, nous considérons uniquement les informations concernant les instances du workflow exécutées avec succès. L'unique condition pour l'extraction du modèle de processus est d'avoir des traces d'exécutions qui rapportent l'état "terminé" de leurs tâches. Cette propriété nous permet d'exploiter des traces d'exécutions qui ne contiennent que l'information concernant la succession des tâches exécutées avec succès sans rapporter par les états d'exécution intermédiaires des tâches ou les temps d'exécution.

Cependant pour extraire le comportement parallèle, voire des boucles, nous enrichissons les traces par les durées minimales et maximales des tâches. Nous partons de l'hypothèse que les instants de fin des tâches sont enregistrés en temps-réels, et qu'ils illustrent l'ordre dans lequel les tâches se sont réellement réalisées. Ceci implique aussi que notre approche ne fonctionne qu'à la condition que les traces d'exécutions respectent cette contrainte.

Cette exigence s'inscrit dans le cadre d'une hypothèse forte spécifiant que les caractéristiques décrivant les propriétés des traces d'exécutions sont conformes à leurs apparitions. C'est à dire que les événements sont enregistrés à l'heure de leur occurrence. Cette hypothèse concerne l'ensemble des traces d'exécutions d'un système de workflow. En d'autres termes, les traces d'exécutions de workflow doivent remplir la condition : si une tâche précède une autre dans un processus, alors il doit y avoir une trace d'exécution d'une instance de workflow rapportant deux événements relatifs à ces deux tâches qui se suivent.

En particulier, une hypothèse importante de notre approche spécifie que si l'exécution d'une tâche dépend directement de la terminaison d'une autre tâche, alors deux événements relatifs à ces deux tâches doivent se suivre directement (immédiatement, sans événement intercalé entre elles) au moins une fois dans une trace d'exécution d'une instance. En d'autre termes, pour être complètes, les traces d'exécutions des workflows doivent couvrir tous les cas possibles (c.à.d si un élément spécifique décrivant un élément de routage au niveau du modèle de processus, les traces d'exécutions doivent contenir ce comportement au moins une fois dans la trace d'exécution d'une instance).

Nous pouvons déduire pour l'analyse minimale des traces, c'est à dire une analyse des traces qui couvre l'ensemble du processus métier, le nombre d'instances nécessaires à la construction du modèle de processus métier.

Une approche qui se base uniquement sur un ensemble d'instances qui ne couvre pas l'ensemble des tâches des processus, ne peut avoir qu'un impact local. Cette faculté définit une spécification locale sur le nombre d'instances pour des traces d'exécutions qui couvrent l'ensemble du processus. Une telle spécification locale permettrait la non obligation de présence de toutes les instances d'exécution possibles pour un workflow. Par exemple, dans un workflow qui contient n tâches concurrentes qui se suivent par m tâches concurrentes, la couverture serait de $n! \times m!$ instances de processus. Si nous partons de l'hypothèse de la localité, le nombre d'instances suffisant pour des traces d'exécutions est évalué à $n! + m!$.

D'une manière générale, le nombre minimal d'instances qui garantit une couverture maximale du processus est défini comme suit :

1. Ce nombre est égal à 1 pour les workflows ne contenant qu'une suite séquentielle de tâches ;
2. Un comportement conditionnel entre n tâches avant un point de jointure ou après un point de diffusion nécessite n traces d'exécutions d'instances différentes reprenant à chaque fois un choix.
3. Un comportement concurrentiel entre n tâches nécessite $n!$ traces d'exécutions d'instances différentes reprenant toutes les combinaisons possibles.

Le Tableau 4.4 représente l'exécution sans échec de six instances de notre exemple de

Instance 1	SC	VD	VSC	VS	AF	MS	RD	TRC
Instance 2	SC	VSC	VS	AF	MS	RD	VD	TRC
Instance 3	SC	VSC	RC	RD	VD	TRC		
Instance 4	SC	VD	VSC	AE	VS	AF	MS	RD	TRC
Instance 5	SC	VD	VSC	VS	MS	RD	TRC	
Instance 6	SC	VD	VSC	AE	VS	MS	RD	TRC

TAB. 4.4 – Flux de quelques instances d'un processus de workflow de traitement d'une commande.

workflow de gestion de commande. Ce tableau contient l'information suffisante que nous supposons être présente pour l'extraction du modèle de processus. En effet, notre workflow exemple contient d'une part un comportement conditionnel entre trois tâches (*Tâche_VS*, *Tâche_AE* et *Tâche_RC*). D'autre part, nous avons flux de contrôles concurrents contenant respectivement *Tâche_VD*, et le bloc de *Tâche_VSC* à *Tâche_RD*. Ces différentes instances nous permettent à la fois de décrire les différents choix possibles du traitement du prêt ainsi que les différents scénarios possibles d'exécution des tâches concurrentes dans ces choix.

Dans cette section, nous avons présenté des aspects importants de la procédure et le format des traces d'exécutions de workflow. Nous avons mis en avant une format unique qui est partagé et compatible avec la plupart des systèmes d'enregistrement des traces d'exécutions de workflows actuels. Ces traces d'exécutions seront l'unique entrée pour nos techniques d'extraction de modèles de processus.

Nous venons de présenter aussi les conditions minimales pour que ces traces d'exécutions assurent l'extraction des modèles de processus. Nous avons défini, d'une part, une structure minimale décrivant la nature de l'information, et d'autre part, le nombre d'instances de workflow nécessaire à l'extraction de modèles. cette section nous a permis de mieux voir la structure et la richesse des données de traces d'exécutions nécessaires au processus d'extraction de modèles. Dans la section suivante, nous allons présenter les principes nécessaires à la construction du réseau de Petri, ainsi que l'approche complète.

4.5 Principe de construction

4.5.1 Succession

Soit $W \subset T^*$ un journal d'événements, et soit $a, b \in T$ deux tâches. On dit que a est "directement succédé" par b dans W, noté $a >_w b$, si et seulement si : il existe une instance $\sigma \subset T$ tel que :

$$\sigma = t_1 t_2 ... t_n \; et \; a = t_i \; et \; b = t_{i+1} \; avec \; 1 \leq i < n \; et : E(a) \leq E(b) - D_{Max}(b)$$

où E est la fonction qui renvoie le temps enregistré dans le journal d'événement représentant la fin d'une tâche t. D_{max} est les durée Maximum de l'exécution d'une tâche. Cette durée est connue à priori.

4.5.2 Croisement

Soit $W \subset T^*$ un journal d'événements, et soit $a, b \in T$ deux tâches. On dit que a "se croise" avec b dans W, noté $a \times_w b$, si et seulement si : il existe une instance $\sigma \subset T$ tel que :

$$\sigma = t_1 t_2 ... t_n \; et \; a = t_i \; et \; b = t_{i+1} \; avec \; 1 \leq i < n \; et \; E(b) - D_{Min(b)} \leq E(a) \leq E(b)$$

où E est la fonction qui renvoie le temps enregistré dans le journal d'événement représentant la fin d'une tâche t. D_{min} est les durée Minimum de l'exécution d'une tâche. Cette durée est connue à priori.

4.5.3 Boucles et Tâches dupliquées

Comme la montre la figure 4.5, les boucles peuvent être déduites par l'absence de chevauchement entre la tâche et elle même. Dans le cas contraire, cela signifie que la tâche A est dupliquée : elle est définie dans deux chemins différents du modèle et elle peut être exécutée en parallèle avec elle même.

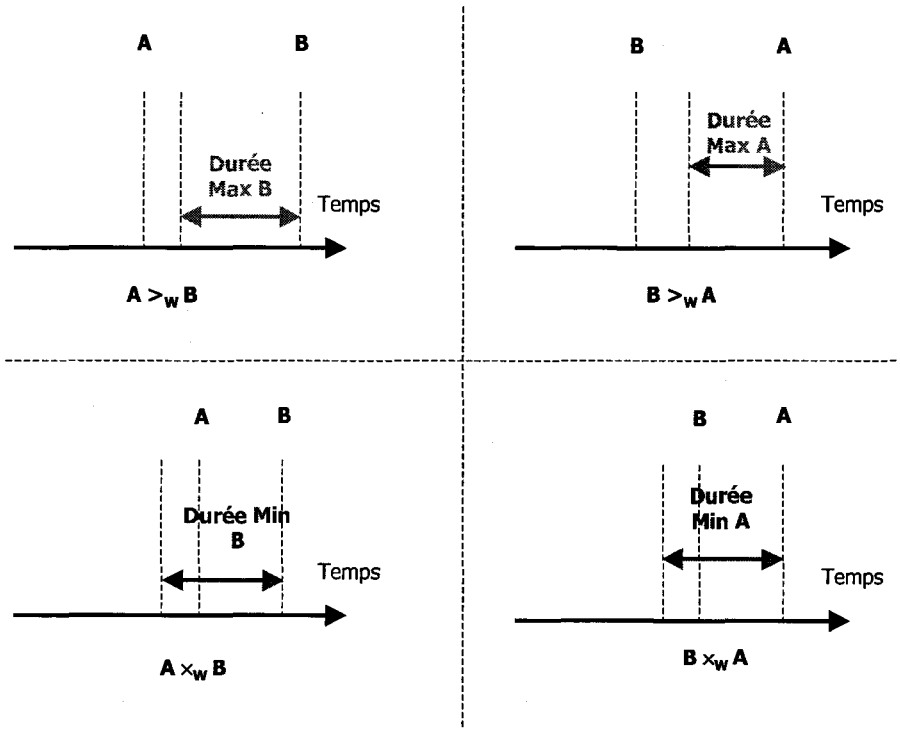

FIG. 4.4 – Illustration des relations "Succession" et "Croisement" de deux tâches.

4.5.4 Théorèmes

Théorème 1 : Pour tout $a, b \in T$:

$$(a \rightarrow_W b) \Longrightarrow (a\bullet \cap \bullet b \neq \emptyset)$$

Théorème 2 : Pour tout $a, b \in T$:

$$(a\bullet \cap \bullet b \neq \emptyset) \Longrightarrow (a >_W b)$$

Théorème 3 : Pour tout $a, b \in T$:

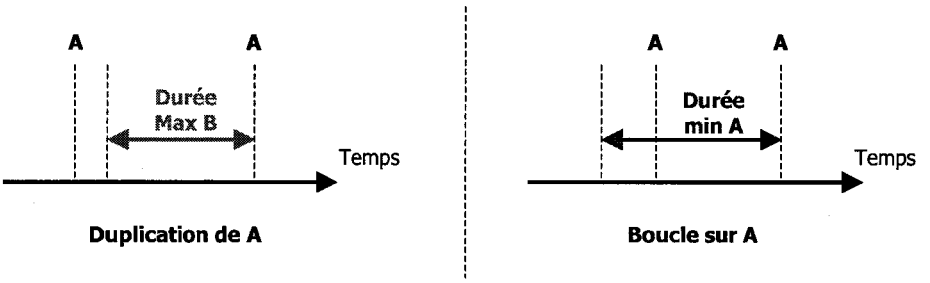

FIG. 4.5 – Exemple d'une "Duplication" et d'une "Boucle" sur une tâche A.

$$(a\bullet \cap b\bullet = \emptyset) \Longrightarrow (a \nparallel_W b)$$

$$(\bullet a \cap \bullet b = \emptyset) \Longrightarrow (a \nparallel_W b)$$

Théorème 4 : Pour tout $a, b, c \in T$:

$$(a \rightarrow_W c) \wedge (b \rightarrow_W c) \wedge (a \nparallel_W b) \Longrightarrow (a\bullet \cap b\bullet \cap \bullet c \neq \emptyset)$$

$$(c \rightarrow_W a) \wedge (c \rightarrow_W b) \wedge (a \nparallel_W b) \Longrightarrow (\bullet a \cap \bullet b \cap c\bullet \neq \emptyset)$$

à partir de ces théorèmes, nous pourrons déduire les constructions équivalentes aux structures de bases

4.6 L'approche complète

4.6.1 Vue générale

L'algorithme commence par extraire des statistiques du journal d'événements. Ces statistiques sont nécessaires à l'extraction de dépendances entre les tâches en analysant les traces

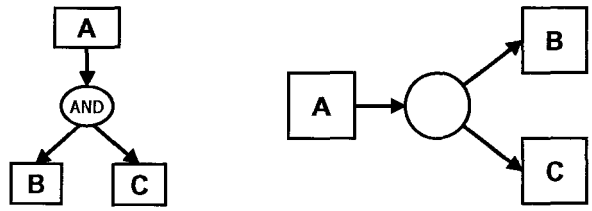

FIG. 4.6 – Construction en Réseau de Petri équivalentes au AND-Split.

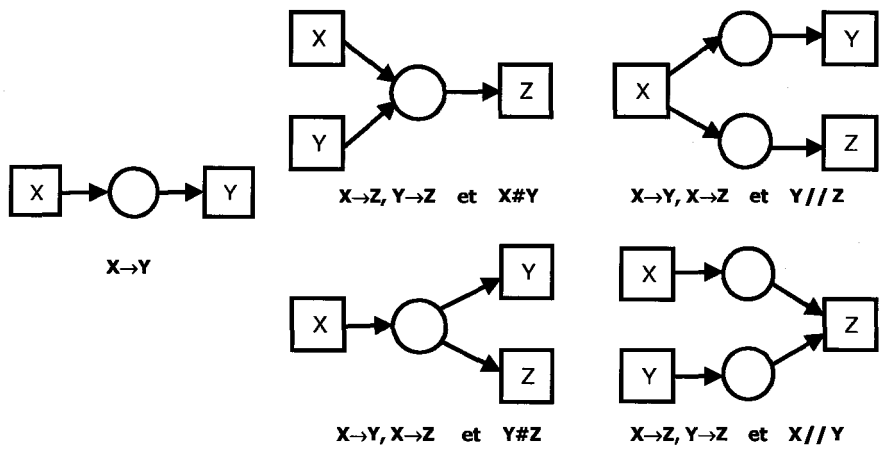

FIG. 4.7 – Construction en Réseau de Petri équivalentes aux structures basiques.

d'exécutions. Ces dépendances sont raffinées par la suite sous la forme de propriétés statistiques qui capturent les principaux comportements des instances d'exécutions. Ces propriétés statistiques seront exploitées pour extraire des motifs de workflow. En effet, un motif de workflow est un ensemble de dépendances qui définit une structure avancée exprimant un comportement spécifique, en termes d'instances d'exécution, caractérisé par les propriétés statistiques. Globalement, nous pouvons regrouper les différentes étapes de l'algorithme de construction en trois grands ensembles de fonctionnalités :

1. L'extraction des dépendances entre les tâches : D'abord, l'algorithme spécifie les dépendances liant les tâches du workflow pendant l'exécution. Le terme utilisé est dépen-

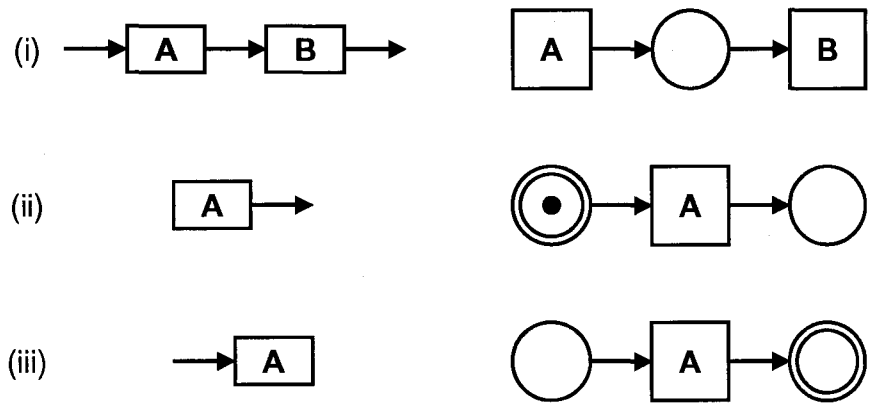

FIG. 4.8 – Construction en Réseau de Petri équivalentes à (i) la séquence, (ii) la tâche initiale et (iii) la tâche finale.

dance "causale" où la fin d'une tâche provoque le début d'une autre tâche. Les relations de dépendances sont calculées sur la base du calcul des propriétés statistiques des comportements des tâches. Les propriétés statistiques contribuent aussi à extraire les motifs de comportements. Nous définissons quatre types de motifs : *AND-Split* (Branchement multiple), *AND-Join* (Rendez-vous), *XOR-Split* (Aiguillage), *OR-Join* (Jonction).

Les motifs "séquentiels" et "conditionnels" héritent de la dépendance causale. Le premier exprime une dépendance causale exclusive entre deux tâches. Tandis que le deuxième définit un ensemble de dépendances causales entre, d'une part, une tâche, et d'autre part, une ou plusieurs tâches d'un ensemble de tâches permettant de décrire et de spécifier un choix entre ces tâches. Le motif "parallèle" caractérise les comportements concurrents entre un ensemble de tâches.

2. L'extraction des motifs de workflow. L'algorithme utilise un ensemble de règles pour extraire les motifs de workflow. Ces règles s'expriment à l'aide de propriétés statistiques qui estiment le motif extrait. Même si nous nous limitons à quelques motifs les plus communs et les plus utilisés, nous considérons dans l'approche la possibilité d'enrichir cet ensemble de motifs en indiquant de nouvelles dépendances, de nouvelles propriétés statistiques ou tout simplement en combinant les propriétés existantes, pour extraire de

nouvelles catégories de motifs.

3. La construction du réseau de Petri sur la base des propriétés statistiques, des dépendances et des motifs de comportement. Cette étape se décline naturellement des étapes précédentes.

4.6.2 Calcul des fréquences

Cela nous permettra de reconnaître les événements exceptionnels ainsi que les erreurs.

Pour commencer, nous construisons une table ou nous calculons pour chaque tâche A, les valeurs suivantes :

- Le nombre total des apparitions de la tâche A, noté $\sharp A$.
- Le nombre de fois où la tâche A est suivie directement par la tâche B dans la même instance, noté $A \rightarrow B$.
- Le nombre de fois où la tâche A est précédée directement par la tâche B dans la même instance, noté $A \leftarrow B$.
- Le nombre de fois où la tâche A est suivie directement ou indirectement par la tâche B dans la même instance, noté $A \dashrightarrow B$.
- Le nombre de fois où la tâche A est précédée directement ou indirectement par la tâche B dans la même instance, noté $A \dashleftarrow B$.
- Métrique calculée en fonctions des variables précédentes. Cette métrique représente une estimation de la causalité entre deux tâches.

Le tableau suivant montre les valeurs des fonctions définies précédemment pour *Tâche_VD* par rapport aux autres tâches qui composent le fichier journal.

Rappelons la signification des tâches :

- Spécification de la commande : *Tâche_SC*,
- Validation de la Décision : *Tâche_VD*,
- Vérification de la Solvabilité du Client : *Tâche_VSC*,
- Accord Exceptionnel : *Tâche_AE*,
- Rejet de la Commande : *Tâche_RC*,
- Vérification du Stock : *Tâche_VS*,

- Appel au Fournisseur : *Tâche_AF*,
- Mise à jour du Stock : *Tâche_MS*,
- Récupération de la Décision : *Tâche_RD*,
- Transmission de la Réponse au Client : *Tâche_TRC*.

T	$\sharp B$	$A \rightarrow B$	$A \leftarrow B$	$A \dashrightarrow B$	$A \dashleftarrow B$	M
SC	213	107	0	213	0	0,531
VD	213	0	0	0	0	0,000
VSC	213	0	107	106	107	0,445
RC	72	0	0	36	36	0,000
AE	70	0	0	35	35	0,219
VS	141	0	0	70	71	0,128
AF	135	0	0	70	65	0,040
MS	141	0	0	70	71	0,196
RD	213	106	0	106	107	0,438
TRC	213	0	106	0	213	0,527

TAB. 4.5 – Exemple.

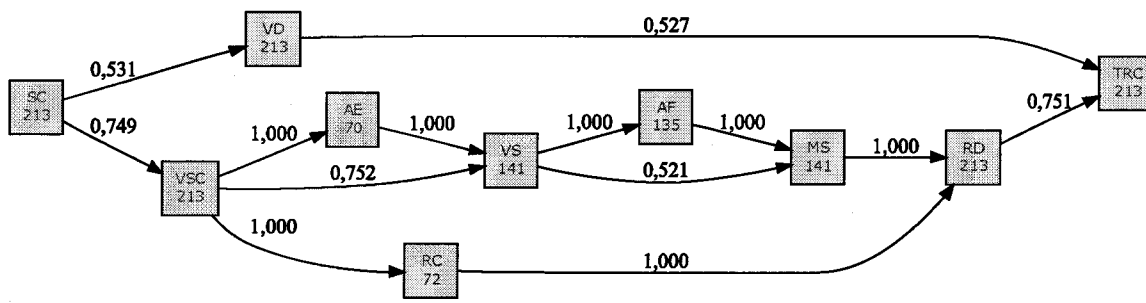

FIG. 4.9 – Graphe des fréquences et dépendances entre les tâches.

4.6.3 Extraction des relations élémentaires directes

Le but de cette section est de décrire la partie de notre algorithme qui extrait les relations élémentaires directes à partir de la table des relations, issue elle même du journal des événe-

ments.

Une relation élémentaire directe relie deux tâches dans le sens où l'instant de fin de la première précède directement l'instant de début de l'autre dans le flux des événements. Pour extraire les relations élémentaires directes, nous calculons une table de relations statistiques, calculée à partir du journal des événements.

Comme les motifs de workflow sont décrits uniquement par des relations élémentaires directes, cette table capture les relations directes entre les tâches, réellement réalisées, c'est à dire caractérisées par des instants de fin dans le journal d'événements.

La table des relations statistiques construit les relations élémentaires entre les tâches d'un journal des événements. Pour chaque tâche T, nous extrayons à partir du journal des événements les valeurs suivantes : (i) le nombre d'occurrences global de cette tâche (dénotée $\sharp A$) (ii) les relations directes aux relations précédentes B_i (dénoté $P(A/B_i)$).

La taille de la table est $n \times n$, où n est le nombre de tâches du workflow. L'entrée (i, j) de la table (notation : $P(T_0 < i < n/T_0 < j < n)$) est la fréquence de la j-ième tâche qui précède immédiatement la i-ième tâche. par exemple, dans cette table $P(Tâche\_MS/Tâche\_VS) = 0,52$ exprime le fait que l'événement correspondant à l'exécution de *Tâche_VS* à 52% de chance de se produire immédiatement avant l'événement correspondant *Tâche_MS* dans les traces d'exécutions du workflow.

Il y a une dépendance entre les dépendances normales entre les tâches et les statistiques des traces d'exécutions exprimées dans la table des relations statistiques. Ainsi chaque relation entre deux tâches du workflow s'exprime par une valeur positive dans l'entrée correspondante dans la table des statistiques. Nous notons respectivement R_D et R_I les relations directes et indirecte induites de la table. Nous pouvons utiliser aussi le terme de relation de dépendance pour exprimer la relation entre les tâches.

La table des relations, ainsi calculée, comprend quelques faiblesses pour exprimer complètement les relations entre les tâches pour le comportement parallèle et conditionnel des tâches. En effet, la table ne permet pas d'identifier pertinemment ces motifs. Dans la suite, nous détaillons ces faiblesses et nous proposons des solutions pour compléter ces statistiques, afin de pouvoir les exploiter pour extraire les motifs de workflow.

4.6.4 Élimination des relations erronées

Si nous supposons que chaque flux d'événements du journal des événements vient d'un workflow séquentiel (c.à.d, ne possédant pas un comportement concurrentiel), une entrée égale à zéro dans la table des relations représente une indépendance entre les tâches, et symétriquement une entrée différente de zéro signifie une relation (c.à.d, une relation séquentielle ou conditionnelle).

Dans les comportements concurrents, comme on peut le voir dans les motifs de workflow (par exemple, diffusion parallèle, synchronisation, etc.), des flux d'événements peuvent contenir des séquences d'événements entrelacées venant de flux concurrents.

Cela signifie que les valeurs non nulles dans la table initiale des relations peuvent ne pas correspondre à des relations, mais à des exceptions. Par exemple, le flux d'événement *Tâche_SC, Tâche_VSC, Tâche_RC, Tâche_RD, Tâche_VD, Tâche_TRC* suggère une relations incorrecte entre *Tâche_RD* et *Tâche_VD*. Ces données ne sont pas tout a fait vraies, parce que la relation entre ces tâches sont faibles. L'importance de la valeur dans la table de relations indique s'il s'agit de relations ou d'exceptionnelles relations entre les tâches.

Nous pouvons déduire que deux tâches A et B sont concurrentes si et seulement si les entrées P(A/B) et P(B/A) dans la table de relations sont non nulles. En se basant sur cette déduction, nous proposons un algorithme pour découvrir le parallélisme et marquer les entrées incorrectes dans la table de relations. Par ce marquage, nous pouvons éliminer la confusion provoquée par le comportement concurrentiel produisant ces entrées non nulles et erronées. L'algorithme parcourt (scanne) la table initiale et marque les dépendances d'activités concurrentes en changeant leurs valeurs par 0. Par exemple, nous pouvons déduire de cette table que *Tâche_VSC* et *Tâche_VD* sont en concurrence (c.à.d, P(*Tâche_VSC/Tâche_VD*) <> 0 et P(*Tâche_VD/Tâche_VSC*) <> 0)), ainsi après application de notre algorithme P(*Tâche_VD/Tâche_VSC*) et P(*Tâche_VSC/Tâche_VD*) seront égales à 0.

4.6.5 Découverte des relations indirectes

À cause du comportement concurrentiel, l'exécution d'une tâche pourrait ne pas dépendre de son prédécesseur immédiat dans le flux des événements, mais elle pourrait dépendre d'autres

tâches qui la précèdent indirectement. Supposons que dans les flux des événements, la fin de la tâche *Tâche_VD* arrive entre la tâche *Tâche_RD* et la tâche *Tâche_TRC*. En conséquence, la tâche *Tâche_RD* ne se termine pas toujours juste avant *Tâche_TRC* dans les traces d'exécutions du workflow. Si, nous avons P(*Tâche_TRC*/*Tâche_RD*) = 0.75 qui présente une relation sous estimée. En fait, la bonne valeur doit être égale à 1 parce que l'exécution de la transmission de la réponse au client dépend exclusivement de la récupération de la décision. D'autres cas similaires peuvent être constatés.

Pour tenir compte de la concurrence, nous introduisons la notion d'intervalle de tâches concurrentes, nous utilisons aussi le terme d'intervalle de concurrence. L'intervalle de concurrence définit un intervalle d'événements dans un flux d'événement. Elle est caractérisée par le fichier des événements, le début de la fenêtre et la fin de la fenêtre,

Les relations directes sont extraites dans un intervalle de concurrence. L'intervalle de concurrence couvre l'ensemble des tâches concurrentes. Au début, la largeur de l'intervalle est égale à 1. Chaque fois qu'une tâche est en concurrence avec une autre tâche, nous ajoutons 1 à la taille de l'intervalle. Si cette tâche n'est pas en concurrence avec d'autres tâches et a des tâches concurrentes qui la précédent, alors nous ajoutons leur nombre à la taille de son intervalle. Par exemple, la tâche refus de la commande est en concurrence avec la tâche de validation de la commande alors la largeur de son intervalle est égale à 2. Se Basant sur ceci, l'algorithme calcule pour chaque tâche la taille de l'intervalle qui les regroupe dans une table de concurrence qui indique pour chaque tâche la taille de son intervalle de concurrence. Cet algorithme scanne la table, calculée dans la dernière section, et met à jour la table de concurrence en fonction du comportement concurrentiel. Ensuite, nous procédons par une partition de chaque flux d'événements qui construit un ensemble de fenêtres se chevauchant partiellement en se basant sur la table de concurrence.

Nous considérons que chaque intervalle partage l'ensemble de ses éléments avec l'intervalle qui le précède sauf le dernier événement qui contient la tâche de référence de la fenêtre. Les intervalles se chevauchent à une tâche près.

En appliquant la partition au-dessus du flux des événements, la taille de l'intervalle de la tâche de validation de la demande est égale à 4 parce que cette tâche est en concurrence avec trois tâches précédentes en concurrence accord exceptionnel, vérification de la solvabilité

du client et refus de la demande. Et, la taille de l'intervalle de la concurrence est égale à 2 parce que cette tâche est en concurrence seulement avec une seule tâche (e.g. validation de la demande).

Notons que pour chaque tâche dans ce flux des événements, son intervalle de concurrence lui permet de couvrir toutes, et seulement toutes, les tâches qui sont dans ses pré-conditions d'exécution.

Finalement, l'algorithme calcule la table des relations finales. Pour chaque intervalle, il calcule pour sa dernière tâche les fréquences des tâches qui la précèdent. La table des relations est calculée en divisant les entrées de chaque rangée par la fréquence de la tâche qui lui est relative.

En appliquant l'ensemble des algorithmes, nous déduisons la table finale des relations qui sera utilisée pour extraire les motifs de workflow. Notons que notre approche s'adapte dynamiquement au comportement concurrentiel, à travers la taille de l'intervalle de concurrence. En effet, cette taille est sensible au comportement concurrentiel : elle augmente en cas de concurrence et elle est égale à 1 en cas d'absence de comportement concurrentiel.

Ainsi, notre algorithme adapte son comportement au contexte de la concurrence, nous utiliserons aussi le terme de parallélisme. Cette stratégie permet d'avoir une approche dynamique d'extraction du parallélisme par rapport à d'autres approches similaires de l'état de l'art qui extraient des motifs fréquents de comportement et qui utilisent une taille fixe de la fenêtre de parallélisme.

4.6.6 Extraction des motifs de workflow

Comme annoncé précédemment, nous considérons 4 opérateurs de base qui construisent les motifs de tâches : *AND-Split*, *AND-Join*, *XOR-Split* et *OR-Join*. Les opérateurs correspondent respectivement aux branchement multiple, rendez-vous, aiguillage et jonction. Les motifs correspondants contribuent à la représentation des routages parallèles, séquentiels, itératifs ou conditionnels des flots de contrôle des instances de processus.

Nous identifions plusieurs types de parcours possibles de tâches d'une instance de workflow : D'une part, un parcours séquentiel exclusif entre deux tâches qui sont exclusivement en

relation de dépendance. D'autre part, on peut se trouver devant des opérateurs de jointure ou de diffusion où nous devons faire un choix parmi l'ensemble des tâches qui précèdent l'opérateur de jointure ou après le point de diffusion. Nous avons identifié un parcours à choix exclusif où l'exécution choisit une seule tâche, un parcours à choix libre qui ne pose pas de restriction sur le choix du nombre de tâches à activer et un parcours sans choix où l'exécution ne fait pas de choix et exécute toutes les tâches.

Nous avons décrit ces comportements en utilisant les propriétés statistiques extraites de la table finale des symboles. Ces propriétés vont être utilisées pour identifier séparément les motifs de workflow à partir des traces d'exécutions.

Nous commençons par le comportement de relation de dépendance exclusive qui décrit un flux séquentiel unique. Le comportement de dépendance mutuelle exclusive entre deux tâches spécifie que l'exécution d'une des deux tâches dépend seulement de la fin de l'exécution de l'autre et que la fin d'exécution de la première tâche déclenche seulement l'exécution de la deuxième. Le comportement de dépendance mutuelle exclusive est défini statistiquement de la manière suivante :

Soit T l'ensemble des tâches d'un workflow W. Soient T_i et T_j deux tâches de W. T_i et T_j décrivent un comportement de relation de dépendance exclusive entre T_i et T_j (notée $\rightarrow$) signifie :

- Les fréquences des tâches sont égales. $\sharp T_i = \sharp T_j$

- la relation de dépendance entre les tâches : P(Ti/Tj) = 1 et quelque soit k(0 < k ; l < n ; k <> j ; l <> i, P (Ti/Tk) = 0 et P(Tl/Tj) = 0).

Le comportement parallèle lie un groupe de tâches en relation non-causale. Il spécifie comment, en termes de concurrence, l'exécution de ces tâches est effectuée. Ce groupe de tâches se trouve dans la structure de contrôle après un opérateur de diffusion ou avant un opérateur de jointure.

Nous notons plusieurs types de comportement parallèle. Ces types de comportement contribuent à la bonne lisibilité sémantique des comportements, et indirectement, ils contribuent à la bonne interprétation du modèle de workflow généré :

- Parallélisme généralisé où l'ensemble des tâches est exécuté simultanément,

- Parallélisme partiel où une partie des tâches est exécutée simultanément,

- Absence de parallélisme où il n'y a pas de parallélisme entre les tâches.

Toutes ces informations sont déduites de la table des relations, de la manière suivante : Soit Ti, 0<i<n, un groupe de tâches formant l'opérande qui se trouve après un opérateur de diffusion ou avant un opérateur de jointure. Cet ensemble correspond à :

- Parallélisme total signifie : quelque soit $i,j,i <> j$, $0 < i,j < n$, $\sharp T_i = \sharp T_j$ et $P(T_i/T_j) = 0$
- Parallélisme partiel signifie : il existe i,j $(i <> j)$, $0 < i$, $j < n$, $P(T_i/T_j) = 0$
- Absence de parallélisme signifie : quelque soit $i,j,i <> j$, $0 < i$, $j < n$, $P(T_i/T_j) \neq 0$

Le comportement conditionnel spécifie, dans l'instance d'exécution du workflow, comment s'effectue le choix d'activation parmi l'ensemble des tâches qui se trouve après un opérateur de diffusion ou avant un opérateur de jointure. Il définit une relation de dépendance entre une tâche et un groupe de tâches formant les opérandes des opérateurs de diffusion et de jointure. Nous avons plusieurs types de comportement conditionnel :

- choix libre où une partie du groupe de tâches est exécutée en fonction des contraintes et paramètres de chaque instantiation, - choix unique où seulement une tâche est exécutée. Le choix de cette activité dépend des contraintes et paramètres de l'instance de workflow exécutée, - pas de choix où toutes les tâches sont exécutées pour chaque instantiation.

Le comportement conditionnel, sous ses trois formes, est défini statistiquement dans la table des relations de la manière suivante :

Soit T une tâche et T_i ; $0 < i < n$ un groupe de tâches d'un opérateur de diffusion ou de jointure. T et T_i ; $0 < i < n$ décrivent un :

- choix libre signifie en termes de fréquences de tâches ($\sharp T <$ Sommei=0 à i=n-1($\sharp Ti$)) ET ($\sharp Ti < \sharp T$) et en termes de relations de dépendance entre les tâches : – Dans un opérateur de diffusion, quelque soit i, 0<i<n, P(Ti/T)=1 (Ti s'exécute certainement après T)
- Dans un opérateur de jointure, 1 < Somme i=0 à n-1, P(T/Ti) <n ; (T s'exécute certainement après la fin de quelques tâches, mais pas forcément après la fin de toutes les tâches)
- choix unique signifie en termes de fréquences de tâches ($\sharp T$ = Sommei=0 à i=n-1($\sharp Ti$)) et en termes de relations de dépendance entre les tâches :
- Dans un opérateur de diffusion, quelque soit i, 0<i<n, P(Ti/T)=1 (Ti s'exécute certainement après T) – Dans un opérateur de jointure, 1 < Somme i=0 à n-1, P(T/Ti) =1 ; (T s'exécute certainement après la tâche Ti)

- pas de choix signifie en termes de fréquences de tâches ($\sharp T = \sharp T_i$) quelque soit i, 0<i<n, et en termes de relations de dépendance entre les tâches : – Dans un opérateur de diffusion, quelque soit i, 0<i<n, P(Ti/T)=1 (Ti s'exécute certainement après T)

- Dans un opérateur de jointure, quelque soit i, 1 < i< n ; P(T/Ti) =1 ; (T s'exécute certainement après la tâche Ti)

Nous divisons les motifs de workflow en trois catégories :

– les motifs de séquence,

– les motifs de jointure

– et les motifs de diffusion.

Dans la suite, nous présentons les règles qui nous ont permis d'extraire les motifs les plus intéressants, par rapport aux applications, appartenant à ces trois catégories.

4.6.6.1 Motifs de séquences

Le motif est composé de deux tâches T_1 et T_2, l'exécution de la tâche T_2 dépend seulement de la fin de la tâche T_1. De ce fait, nous avons employé la propriété statistique de relation de dépendance exclusive pour extraire cette relation liant T_2 à T_1.

Règles

Motifs de workflow ($\sharp T_2 = \sharp T_1$)

Motif de séquence ($P(T_2/T_1) = 1$)

Par exemple, en appliquant les règles de ce motif sur la table des relations, nous en déduisons le motif de séquence liant *Tâche_VD* et *Tâche_TRC*. En effet, ($\sharp VD = \sharp TRC$) et (P(*Tâche_TRC/Tâche_VD* = 1) et quelque soit T_i, 0<i<n, T_i<>*Tâche_VSC* ; P(*Tâche_VD*/T_i) = 0 et quelque soit T_i, $0 < i < n$, T_i<>*Tâche_VD*, P(T_i/*Tâche_VD*) = 0.

4.6.6.2 Motifs de diffusion

Dans cette catégorie de motifs, un opérateur de diffusion est utilisé. Une tâche T se divise en plusieurs sous-tâches $T_i; 0 \leq i < n$ qui peuvent être, selon le motif utilisé, exécutées ou pas. La relation de dépendance entre les tâches T et $T_i; 0 \leq i < n$ avant et après l'opérateur de diffusion diffère dans les trois motifs de cette catégorie : choix exclusif, diffusion parallèle.

Ces relations de dépendance sont décrites à travers les propriétés statistiques conditionnelles, vues précédemment. Le motif du choix exclusif sélectionne parmi plusieurs tâches, une seule tâche, via l'opérateur de diffusion *XOR-split*.

Quant aux motifs de diffusion parallèle, ils se différencient par la propriété statistique du parallèle total. À l'inverse de la diffusion multiple ou seulement une partie tâches est exécutée, toutes les tâches de T_i sont exécutées dans le motif diffusion parallèle.

L'absence de parallélisme entre les tâches B_i, dans le motif choix exclusif, est assuré par la propriété statistique "choix exclusif", vue précédemment. Par exemple, la table des relations de dépendance indique la présence de motifs de diffusion parallèle liant les tâches *Tâche_VSC* et *Tâche_VD*.

Ci-dessous les règles de construction des motifs :

($\sharp T$ = Sommei=0 à i=n-1($\sharp Ti$))

Motif du choix exclusif (XOR-split) : quelque soit i, $0 < i < n$, $P(T_i/T) = 1$ quelque soit i, j, $0 < i, j < n$, $P(T_i/T_j) = 0$

($\sharp T = \sharp Ti$, quelque soit $0 < i < n$)

Motif de la diffusion parallèle (AND-split) : quelque soit i, $0 < i < n$, $P(T_i/T) = 1$ quelque soit i, j, $0 < i, j < n$, $P(T_i/T_j) = 0$

($\sharp T <$ Sommei=0 à i=n-1($\sharp T_i$)) ET ($\sharp T > \sharp Ti$, quelque soit 0<i<n)

Motif de choix multiple (OR-split) : quelque soit i, $0 < i < n$, $P(Ti/T) = 1$ il existe i,j, $0 < i, j < n$, $P(T_i/T_j) = 0$

4.6.6.3 Motifs de jointure

Cette catégorie de motifs est caractérisée par un opérateur de jointure où plusieurs tâches se rejoignent en un même point du flux de contrôle. Le nombre de chemins nécessaires pour l'activation de la tâche B après un opérateur de jointure dépend du motif utilisé.

Nous identifions deux motifs de cette catégorie : les motifs de jointure simple (OR-Join) et la synchronisation (AND-Join), nous avons analysé les relations de dépendance entre les tâches T_i et T avant et après l'opérateur de jointure. Ainsi, les propriétés du choix unique et de l'absence de parallélisme sont utilisées pour identifier le motif de jointure simple où les flux de contrôle portant les activités T_i se rejoignent sans synchronisation, ni concurrence. Les

propriétés du non-choix et de parallélisme total sont toutes les deux utilisées pour identifier le motif de synchronisation où toutes les tâches T_i convergent ensemble en se synchronisant au point de jointure.

4.6.7 Algorithme de construction

Soit W un workflow sur un ensemble d'activité T, l'algorithme de construction du réseau des Petri est définit comme suit :

Algorithme 1 Algorithme de construction

ENTRÉES: un fichier log d'événements
SORTIES: WF-net $A(W)$

1. $T_W = \{t \in T | \exists \sigma \in W : t \in \sigma\}$,
2. $T_I = \{t \in T | \exists \sigma \in W : t = first(\sigma)\}$,
3. $T_F = \{t \in T | \exists \sigma \in W : t = last(\sigma)\}$,
4. $X_W = \{(A,B) | A \subseteq T_W \wedge B \subseteq T_W$
   $\wedge \forall a \in A, \forall b \in B : a \rightarrow_W b$
   $\wedge \forall a1, a2 \in A : a_1 \sharp_W a_2$
   $\wedge \forall b_1, b_2 \in B : b_1 \sharp_W b_2\}$,
5. $Y_W = \{(A,B) \in X_W | \forall (A',B') \in X_W : A \subseteq A' \wedge B \subseteq B' \Rightarrow (A,B) = (A',B')\}$,
6. $P_W = \{p_{(A,B)} | (A,B) \in Y_W\} \cup \{i_W, o_W\}$,
7. $F_W = \{(a, p_{(A,B)}) | (A,B) \in Y_W \wedge a \in A\}$
   $\cup \{(p_{(A,B)}, b) | (A,B) \in Y_W \wedge b \in B\}$
   $\cup \{(i_W, t) | t \in T_I\}$
   $\cup \{(t, o_W) | t \in T_F\}$,
8. $A(W) = (P_W, T_W, F_W)$.

Retourner $A(W)$.

Cet algorithme construit un réseau de Petri (P_W, T_W, F_W) à partir d'un journal d'événements W. L'ensemble des tâches T_W, des tâches initiales T_I (les tâches avec lesquelles peut commencer toute instance du processus) et des tâches finales T_F (les tâches qui déterminent la fin de l'exécution d'une instance) sont obtenus facilement. Les étapes 4 et 5 permettent de retrouver

les pairs qui satisfont les motifs définis précédemment. Ces deux étapes sont les plus coûteuses en temps de calcul. Enfin Les étapes 6, 7 et 8 permettent la construction du réseau de Petri représentant le processus équivalant à notre journal d'événements. A noter que la complexité du calcul de l'ensemble Y_W est exponentielle par rapport au nombre de taches. En pratique, ce nombre ne dépasse pas la centaine dans les processus existants, ce qui fait que la complexité n'est pas un grand obstacle pour les applications de cette échelle.

Pour illustrer l'algorithme, on prend le fichier log exemple suivant : $W = \{ABCD, ACBD, AED\}$, l'exécution de l'algorithme donnera :

1. $T_W = \{A, B, C, D, E\}$,

2. $T_I = \{A\}$,

3. $T_O = \{D\}$,

4. $X_W = \{(\{A\}, \{B\}), (\{A\}, \{C\}), (\{A\}, \{E\}), (\{B\}, \{D\}), (\{C\}, \{D\}),$
   $(\{E\}, \{D\}), (\{A\}, \{B, E\}), (\{A\}, \{C, E\}), (\{B, E\}, \{D\}), (\{C, E\}, \{D\})\}$,

5. $Y_W = \{(\{A\}, \{B, E\}), (\{A\}, \{C, E\}), (\{B, E\}, \{D\}), (\{C, E\}, \{D\})\}$,

6. $P_W = \{iW, oW, p(\{A\}, \{B, E\}), p(\{A\}, \{C, E\}), p(\{B, E\}, \{D\}), p(\{C, E\}, \{D\})\}$,

7. $F_W = \{(iW, A), (A, p(\{A\}, \{B, E\})), (p(\{A\}, \{B, E\}), B), ..., (D, oW)\}$,

8. $a(W) = (P_W, T_W, F_W)$.

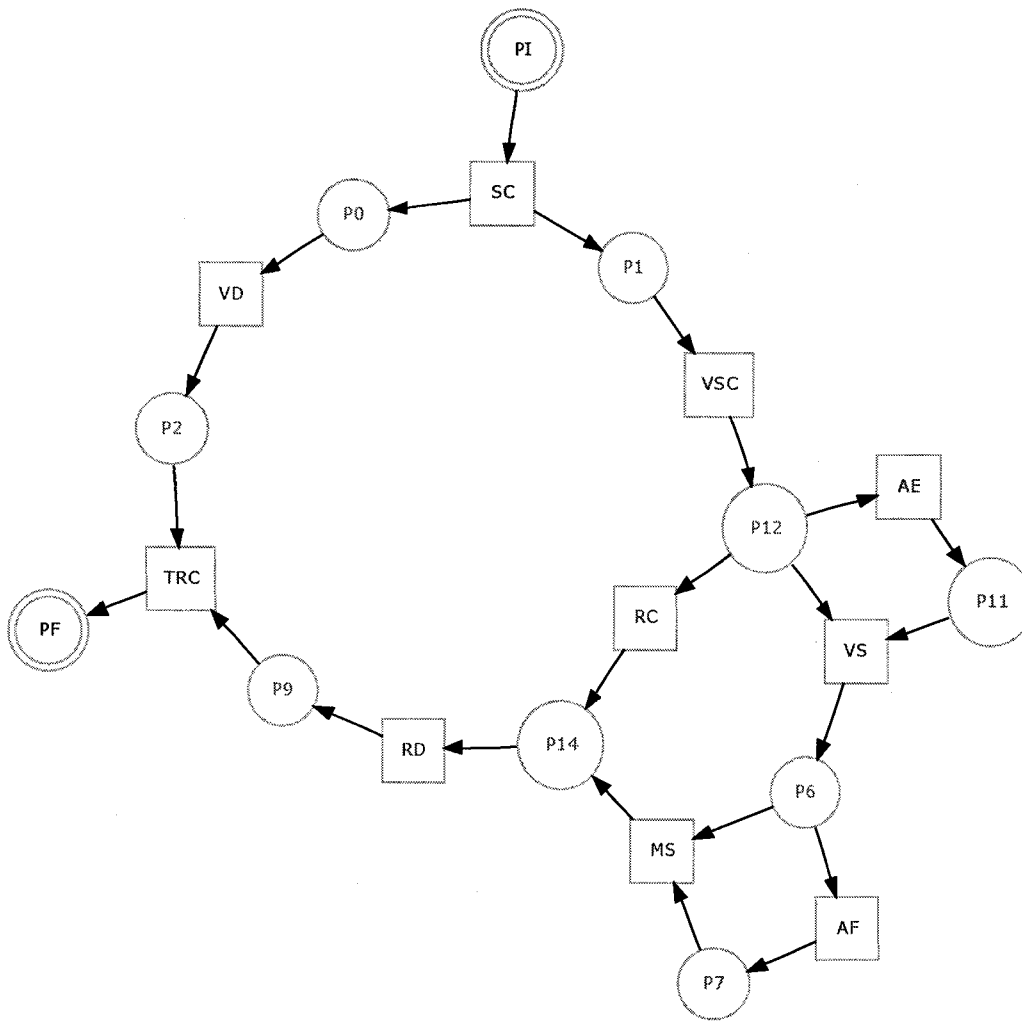

FIG. 4.10 – Réseau de Petri équivalent au processus de gestion des commandes.

4.6.8 Boucles

Le problème de la reconstruction de la boucle de tâche peut être résolu par l'ajout de deux étapes, l'une avant le lancement de l'algorithme de construction et l'autre après. le rôle de la première étape et de nettoyer le fichier log et supprimant la tâche sur laquelle on suppose

l'existence de la boucle. La dernière étape permettra d'ajouter la boucle sur la place créé entre les deux transitions correspondant aux tâches qui précèdent et suivent la tâche sur laquelle on boucle.

Soit l'ensemble de tâches $T = \{A, X, Y\}$, et W un fichier log sur T tel que $W = \{XY, XAY, XAAY\}$.

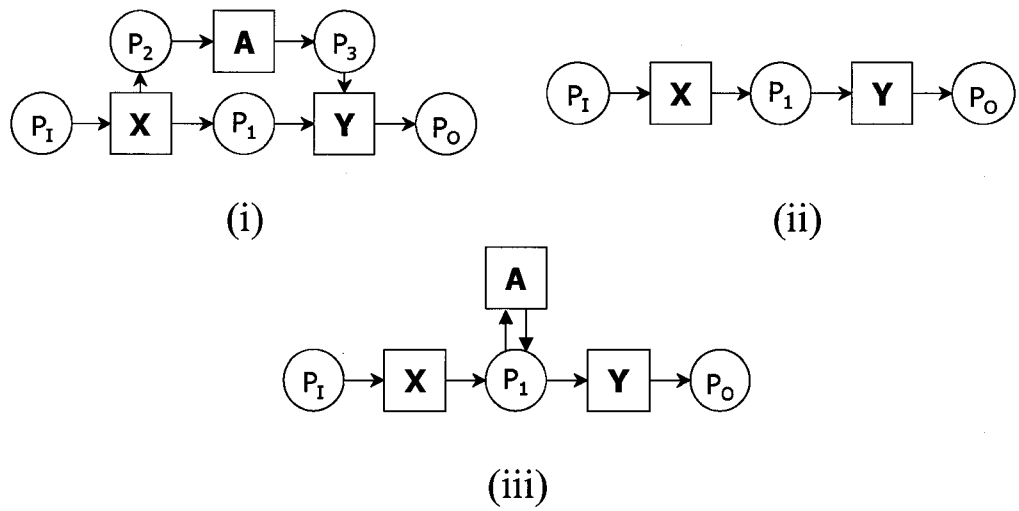

FIG. 4.11 – Détection des Boucles sur une tâches.

La figure 4.11 montre un exemple d'application de cette méthode sur le fichier log W. L'établissement des relations d'ordre entre les tâches on montré l'existence d'une boucle sur la tâche A qui est exécuté 0, 1 ou 2 fois dans les instances XY, XAY, $XAAY$ respectivement. (i) représente le réseau de Pétri construit sans prendre en considération l'existence d'une boucle sur la tâche A. Ce réseau et considéré comme erroné car il ne permet de reproduire que les instances XY, XAY, mais pas $XAAY$. Le réseau de Pétri présenté dans (ii) est construit après la suppression de la tâche A de toutes les instances lors du pré-traitement du fichier log W. Et enfin, le réseaux de Pétri (iii) est obtenu après l'ajout de la boucle sur A au niveau de la place présente entre les transitions X et Y.

4.7 déploiement du modèle de processus

Dans cette section, nous proposons des méthodes d'évolution du workflow initial sur la base du modèle de processus généré. Ces méthodes contribuent à l'évolution du processus métier initial et à sa normalisation. L'objectif étant de placer l'extraction des modèles de processus dans le cycle de développement d'un workflow : conception, exploitation, extraction de modèles de processus, conception...

Nous comparons le modèle de processus extrait avec le modèle initial du processus, et nous en déduisons les différences. Ces différences seront particulièrement utiles pour faire évoluer positivement la comportement du processus métier.

Pour ce faire, nous proposons, par la suite, un ensemble de règles qui permettent de faire évoluer en mieux le comportement du processus métier, en réduisant les erreurs, en supprimant les comportements inutiles et en normalisant en cas d'échec d'une tâche.

Le modèle de processus extrait nous permet de faire évoluer le modèle de processus conçu dans la phase initiale, dans le sens où il permet de détecter des comportements qui peuvent être erronés, ou jugés inutiles, car ils ne coïncident pas avec les observations réelles. Par exemple, à quoi ça sert de garder une tâche qui n'est jamais exécutée. Nous partons du principe que la taille du journal des événements est suffisamment riche et porte sur une période suffisamment importante pour que les comportements soient réellement représentatifs du comportement réel du workflow. C'est à dire un comportment qui soit représentatif des interactions des utilisateurs du workflow.

Les règles d'évolution que nous mettons en avant dépendent du processus métier. En effet, le processus métier exprimé par le workflow doit respecter le contexte métier. Ainsi, nous proposons un ensemble de règles génériques qui permettent de :

- supprimer ou au moins corriger tout comportement erroné,
- suggérer un ensemble de comportements additionnels.

4.7.1 Analyse comparative

L'analyse comparative entre le modèle de workflow initial et le modèle de workflow extrait est nécessaire pour voir ce qui différencie la conception initiale et la réalité des usages. En

effet, l'objectif est de voir comment le modèle initial évolue dans la réalité des applications, et de savoir s'il s'agit de différences rares, exceptionnelles, voire éphémères ou s'il s'agit de différences notables et importantes, qui exige de la part du concepteur une reconsidération de ses choix initiaux de conception, et de reconcevoir le workflow. En effet, tout l'intérêt de l'extraction des modèles de processus est de faire évoluer le workflow initial en fonction d'observations réelles, lorsque les différences entre le workflow initial et le modèle de workflow extrait sont jugées importantes.

Si les différences, issues de l'analyse comparative entre le workflow initial et le modèle de processus extrait, sont non seulement importantes, mais aussi fréquentes, alors cela met en avant une carence certaine dans l'analyse et la prise en compte des réels besoins des utilisateurs dans la conception du workflow. Le rôle du concepteur est d'arriver à un modèle de processus relativement proche du modèle de processus extrait, suite aux usages.

L'analyse comparative peut être vue comme un observatoire qui tire la sonnette d'alarme, chaque fois qu'il y a écart entre entre le modèle de processus initial et le modèle de processus extrait.

Il faut préciser aussi que l'écart entre le modèle de processus initial et le modèle de processus extrait, suites aux observations des usages, peut signifier aussi que les utilisateurs ont mal exploités le workflow initial. Comme résultat, nous recommandons les utilisateurs de mieux respecter le comportement initial. Ce cas de figure part du principe que les écarts ne correspondent pas à une réelle évolution du processus métier. Seul le contexte du workflow permet de discerner entre un écart qui exprime une évolution réelle du processus métier et un écart qui exprime un mauvais usage du workflow par les utilisateurs.

Quelque soit l'interprétation de l'écart entre le modèle de processus métier initial et le modèle de processus métier extrait, notre principal rôle est d'extraire le modèle de processus métier sur la base d'observations sauvegardées dans le journal des événements. Notre rôle aussi, secondaire cette fois-ci, est de produire une méthode qui extrait la différence entre le workflow initial et le modèle de processus issu des observations.

Le figure 4.12 résume bien l'intéraction entre le modèle initial et le modèle extrait.

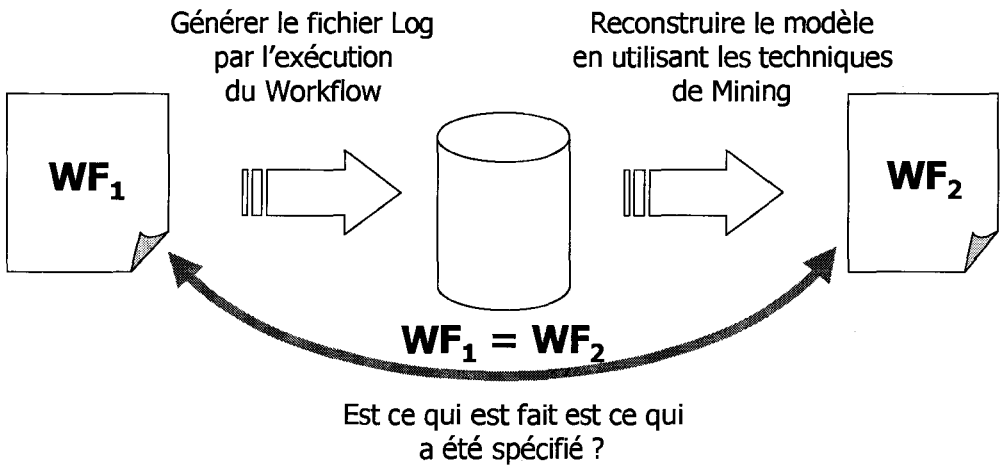

FIG. 4.12 – Analyse comparative.

4.7.2 Procédure de normalisation

4.7.2.1 relations exceptionnelles

Le système workflow exploite les relations exceptionnelles entre ses tâches pour normaliser les instances de processus.

Une instance de processus se trouve à tout moment dans l'un des deux états suivants : normal ou anormal (semi-échec). Ces deux états sont dépendants des états de fin des tâches de l'instance du processus : fin avec succès et fin avec échec. L'état normal et anormal de l'instance de processus dépendent respectivement de l'état de la fin avec succès ou échecs des tâches. En effet, la fin d'une tâche se trouve à tout moment dans la situation d'échec ou de succès.

Si les tâches se terminent avec succès (état fin avec succès), alors l'instance de processus s'exécute normalement, et se trouve dans l'état normal. Si, en revanche, l'une des tâches de l'instance du processus se termine avec échec, alors l'état de l'instance du processus transite à l'état anormal. Par défaut, l'instance de processus se trouve dans un état normal.

L'état anormal de l'instance de processus indique qu'une tâche de l'instance de processus

a échoué. La raison de cet échec peut avoir une multitude raisons qu'elles soient humaines ou logicielles. Quelque soit la raison, notre rôle dans la thèse est d'identifier dynamiquement les tâches qui peuvent échouer, et de proposer une procédure permettant d'y remédier, c'est à dire une procédure qui normalise l'exécution des instances de processus.

Ainsi, lorsqu'une instance de processus, suite à l'échec d'une tâche, se trouve dans un état anormal, alors une procédure est enclenchée pour faire face à la tâche échouée et pour retrouver un état normal de l'instance de processus.

La réalisation de la procédure peut être automatique ou manuelle (e.g. acteurs externes de type administrateur, utilisateur). Le choix du caractère manuel ou automatique dans la réalisation des tâches est à la charge de l'administrateur du workflow et de l'expert du processus métier.

Le début d'une instance de processus se trouve toujours dans un état normal, l'échec d'une tâche la transforme en un état anormal. L'état anormal est traité grâce à la procédure de normalisation, qui la ramène à un état normal avec l'espoir qu'elle se termine alors avec succès. Le but ici, est permettre à l'instance de processus de terminer son exécution, même si certaines tâches ont échouées.

Deux types de relations sont mises en oeuvre dans la procédure de normalisation. Nous insistons sur le caractère exceptionnel de ces relations. En effet, ces relations sont exécutées seulement en cas d'échec d'une tâche, ce qui rend la relation, en théorie, un phénomène exceptionnel. Nous utiliserons par la suite, d'une manière non séparée, les expressions de relation exceptionnelle ou tout simplement de relation.

- les relations exceptionnelles de continuité,
- les relations exceptionnelles de fin.

Les relations exceptionnelles de continuité permettent de définir la continuité d'exécution comme procédure de normalisation. Ainsi, une relation exceptionnelle de continuité de tâche T_1 vers la tâche T_2 signifie que l'échec de T_1 active T_2. Nous distinguons deux types de continuité :

- une continuité vers une tâche précédente, c'est à dire que la tâche T_2 se situe avant T_1 dans la structure du processus.
- une continuité vers une tâche ultérieure, c'est à dire que T_2 se situe après T_1 dans la

structure du processus.

Exemple : Le workflow de traitement d'une commande spécifie, pour la tâche de vérification de stock, les relations exceptionnelles de continuité vont vers la tâche de rejet de la commande. Ainsi, la tâche de rejet de commande sera activée lorsque la vérification du stock échoue.

Les relations exceptionnelles de fin arrêtent des tâches qui s'exécutent en parallèle avec la tâche qui vient de subir l'échec d'exécution. Ainsi, une relation exceptionnelle de suppression de T_1 vers T_2 signifie que l'échec de T_1 déclenche la fin de T_2.

Exemple : Le workflow de traitement de commande spécifie que l'échec de la tâche de vérification de la solvabilité du client, engendre la fin de validation de la commande.

Les relations exceptionnelles doivent respecter des contraintes en fonction des motifs de workflow. En effet, le comportement représenté par les relations exceptionnelles de continuité et de fin est étroitement lié aux motifs de workflow. La procédure de normalisation, présentée ci-dessous, respecte ces contraintes.

Contrainte 1 Une relation exceptionnelle de fin entre deux tâches signifie que les deux tâches sont en parallèles par diffusion ou jointure (split ou join). La relation exceptionnelle de fin ne peut exister entre des tâches séquentielles.

Contrainte 2 Une tâche autonome ne peut pas être reliée à d'autres tâches par des relations de de fin, même si elle est exécutée en concurrence avec d'autres tâches qui peuvent échouer. En d'autres termes, pour les tâches parallèles des motifs split, Il n'y a pas de relations exceptionnelles de fin vers les tâches atomiques ;

Contrainte 3 Un emplacement de normalité ne doit pas se trouver dans une structure parallèle du processus.

Contrainte 4 Toutes les tâches doivent être soient autonomes ou leur être assignée une relation exceptionnelle de continuité ou de fin.

Ces contraintes sont partagées par tous les workflows indépendamment du processus métier traité. Bien entendu, ces contraintes peuvent être enrichies de nouvelles contraintes dépendantes du processus métier.

4.7.2.2 Procédure générale

Le système de workflow décide après l'échec d'une tâche d'une instance de processus, de la suite à donner. Soit une procédure de normalisation est enclenchée. Soit l'exécution de l'instance de processus continue son exécution en ignorant l'échec. Dans le deuxième cas, l'administrateur du workflow estime que l'échec ne remet pas en cause la normalité de l'instance de processus. Dans le premier cas, nous considérons que la tâche est normalisée, c'est à dire qu'elle propose une procédure de normalisation (une relation exceptionnelle lui est assignée).

Nous partons du principe que l'administrateur du workflow identifie des emplacements de normalité de la structure du processus. Ces emplacements représentent des étapes intermédiaires d'exécution des instances du processus métier. Ces emplacements dans la structure du processus seront utilisés par la procédure de normalisation pour traiter les problèmes d'échecs. Un emplacement de normalité correspond à un point de reprise potentiel dans le cas d'un échec. Ces choix dépendent du processus métier définis par les experts du domaine et mises en oeuvre par l'administrateur du processus métier.

Par exemple, l'emplacement de normalité, suite à l'échec de la tâche de vérification de stock dans notre exemple, est situé avant les tâches d'accord exceptionnel et de rejet de la commande. Quant à l'emplacement relatif à la tâche de transmission de la réponse, il s'agit d'une tâche autonome en elle même, elle est rejouable.

Dans la procédure de normalisation, si l'emplacement de normalité ne se situe pas au niveau de la tâche elle-même, alors elle est certainement liée par relations exceptionnelles de continuité ou de fin. Selon la position de l'emplacement, les relations exceptionnelles de continuité concernent des emplacements situés avant ou après la tâche qui subit l'échec. Nous parlons alors de relations exceptionnelles de continuité avant et arrière. Encore une fois, l'objectif est de ramener le workflow à un état de normalité, ce qui peut nécessiter de compenser les tâches déjà réalisées par de nouvelles tâches de compensation qui remplace la tâche échouée. Les nouvelles tâches peuvent ne pas se trouver dans le processus initial, et qui sont exécutées

seulement en cas d'échec d'exécution afin de compenser ses effets.

Par exemple, le workflow de traitement de la commande montre que suite à l'échec de l'a tâche de récupération de la décision, l'instance de processus exécute une nouvelle tâche de compensation des risques qui n'existe pas dans le processus initial, afin de compenser les effets de cet échec et reprendre le processus de décision à partir d'un emplacement cohérent se trouvant après la tâche de vérification de la solvabilité du client.

En outre, un échec d'une tâche peut causer une fin volontaire d'une ou plusieurs tâches actives. Si une telle situation se produit, l'échec d'une tâche induit le fin d'autres tâches. Ce comportement est décrit par la relation exceptionnelle de fin. Le workflow de notre exemple de indique que suite à l'échec de l'activité de vérification de la solvabilité du client, la tâche de validation de la décision est supprimée si elle n'a pas encore fini son exécution, puisque cet échec de vérification de la solvabilité du client cause la fin de l'instance du workflow.

En récapitulant, après l'échec d'une tâche, si l'instance de processus auxquelle appartient la tâche se trouve dans un état anormal, alors plusieurs actions sont possibles pour y remédier :

- L'échec de la tâche n'a pas d'incidence sur l'exécution de l'instance de workflow. Dans ce cas, nous n'avons pas besoin de procedure de normalisation.

- L'échec de la tâche a des incidences sur l'instance de processus. Elle nécessite qu'une procédure de normalisation soit activée. Nous distinguons trois traitements différents :

- La tâche est rejouée automatique jusqu'à son succès. C'est le cas le plus simple, il est mis en avant, si l'échec de la tâche est sans conséquences sur la normalité de l'instance de processus. En d'autres termes, l'effet général de son exécution ou non avec succès n'influence pas l'exécution d'autres tâches, mais interrompt nécessairement l'exécution de l'instance, ce qui nécessite une re-exécution jusqu'au succès.

- La tâche est arrêtée et la relation exceptionnelle de continuité est mise en avant. Le flux de contrôle est transféré à un emplacement de normalité. Dans ce cas, l'échec de la tâche influence d'autres tâches et le transfert du flux de contrôle est nécessaire. Sans ce traitement, l'instance de processus reste interrompue causant l'échec global de l'instance du processus. Par ailleurs, le traitement peut nécessiter l'exécution d'une tâche qui compense les effets de l'échec. Le concepteur peut choisir après l'échec d'une tâche de changer le traitement initial de l'instance en choisissant un chemin alternatif et en continuant l'instance de workflow vers

l'avant. L'instance de processus peut exécuter une autre tâche et termine au mieux l'instance de processus.

4.7.2.3 Règles appliquées par la procédure de normalisation

Règle 1 : Elimination des procédures inutiles

- éliminer les procédures de normalisation, réalisées dans le workflow initial, et qui ne sont pas utilisées dans le modèle de processus extrait. En effet, à quoi ça sert de garder une procédure de nomalisation, dans la mise en oeuvre, est forte coûteuse, si la tâche se termine toujours avec succès.

- éliminer toutes les relations de fin sur des tâches non concurrentes. En effet, les procédures de terminaison forcée sont utilisées sur des tâches concurrentes, dans le cadre des motifs *XOR-split, AND-split*. Si une tâche concurrente de ces motifs se termine, alors la procédure force la terminaison (la fin) des autres tâches concurrentes afin de respecter le motif. En conséquence, si les observations révèlent que les tâches ne sont pas en concurrence dans le cadre de ces motifs, alors il n y a aucun intérêt de garder une telle procédure. En effet, une telle procédure est forte coûteuse à supporter et à gérer.

Reprenons l'exemple du traitement d'une commande, et illustrons la mise en oeuvre de la règle 1, via cet exemple. Supposons, sur la base de l'analyse du journal des événements, que la tâche de vérification de la solvabilité du client (*Tâche_VSC*) se termine toujours avec succès. C'est à dire qu'elle ne s'est jamais terminée avec un échec. Or, dans le workflow initial, il a été spécifié que la vérification de la solvabilité du client peut se terminer avec un échec. Cela constitue donc un écart avec le modèle de processus extrait. En conclusion, la procédure de normalisation qui associe la tâche de vérification de la solvabilité du client (*Tâche_VSC*) et la tâche de validation de la décision (*Tâche_VD*) est supprimée.

Règle 2 : recommandation de procédures de normalisation.

Prenons le problème de façon inverse. Supposons que le concepteur n'a pas prévu des situations d'échec d'une tâche. Le concepteur n'a donc pas prévu une procédure de normalisation à cette tâche. Dans le cas, où les observations mettent en avant des situations d'échecs associées à cette tâche, alors une procédure d'de terminaison forcée est recommandée pour y remédier. Généralement, l'association à une tâche, une procédure de normalisation dépend du contexte

métier. Le contexte métier peut ne pas stipuler une procédure de normalisation, même si, sur la base des observations réelles, la tâche peut aboutir à des échecs. C'est pour cela que nous considérons que la décision d'assigner une procédure de normalisation aux tâches qui peuvent aboutir à des situations d'échecs reste de la responsabilité de l'administrateur du workflow. En d'autres termes, la décision nécessite l'intervention humaine. Pour simplifier, nous mettons l'accent sur l'application de la règle d'association de procédure de normalisation à une tâche, suite à la présence d'échecs, indépendamment du contexte métier. Dans le cadre actuel du travail, nous ne considérons pas le contexte métier dans l'application de ces règles. En effet, la prise en compte du contexte métier peut faire l'objet d'une thèse entière. Cela dépasse largement le cadre de notre travail. En conséquence, définir l'influence ou pas d'une tâche sur les autres tâches dépend du contexte métier, et revient au jugement du concepteur.

La règle 2 est caractérisée par les comportements suivants :

(Règle1) Si la tâche t n'a pas d'influence sur les autres tâches (t est considéré comme autonome), alors suggérer que t soit rejouable.

(Règle2) Si la tâche t a une influence sur les autres tâches, alors suggérer une relation exceptionnelle de continuité vers un emplacement en amont/aval de la tâche courante. Le choix en amont/aval dépend à la fois de la position des emplacements de normalité et le contexte métier du processus. La règle suggère une dépendance alternative en avant/arrière dans le cas de l'existence d'une tâche qui représente un point de normalité en avant pour la tâche échouée. Cette règle respecte la contrainte selon laquelle la tâche représentant ce point cohérent ne soit pas en concurrence avec la tâche échouée.

(Règle3) Suggérer une relation exceptionnelle de fin vers les tâches parallèles. La règle suggère suggèrent des relations exceptionnelles de fin vers d'autres tâches. Ces relations sont nécessaires pour supporter la procédure de normalisation initialement conçue.

La règle déduit le fait qu'on peut proposer une relation de fin à partir de la tâche échouée vers les tâches qui s'exécutent en parallèle avec elle et qui sont non autonomes.

Les règles suggèrent des propositions différentes pour répondre au besoin de procédure de normalisation d'une tâche qui n'était pas prévue dans la phase initiale de conception.

Exemple : Nous nous proposons d'illustrer l'applicabilité de nos règles de suggestions de la procédure de normalisation sur notre exemple de traitement d'une commande.

Supposons que nous ayons observé que la spécification de la commande peut échouer. Ceci constitue une différence entre le modèle extrait et le modèle initialement conçu qui suppose que la spécification de la commande n'échoue jamais. Si nous observons que spécification de la commande influence d'autres tâches, nous pouvons recommander au concepteur, sur la base des règles ci-dessus, la normalisation de la validation de la décision :

- Par l'application de règle2, nous pouvons suggérer la normalisation avec une relation exceptionnelle de continuité vers le point avant la spécification de la commande.

- Par l'application de règle2, nous pouvons suggérer la normalisation avec une relation de continuité vers le points qui se situe juste après le début du processus.

- Par l'application de règle3, nous pouvons suggérer de spécifier des relations de fin entre la validation de la décision d'une part et toutes les autres relations d'autre part.

4.7.3 Synthèse

Dans cette section, nous nous sommes intéressés à l'extraction et à l'évolution du comportement du workflow. En utilisant les résultats observés et en les comparant au schéma de workflow initialement conçu, nous proposons un ensemble de règles pour l'évolution du comportement du workflow. Les écarts entre les deux modèles conçus et observés peuvent exprimer à la fois des erreurs de choix conceptuels du modèle conçu comme des nouveaux besoins d'évolution. Ainsi, Cette étape nous permet de prendre en compte les nouvelles contraintes.

En conclusion, cette section nous a permis de répondre à la problématique : "Comment utiliser les résultats de l'extraction du modèle de processus pour améliorer le traitement des échecs d'exécution et optimiser la fiabilité et le coût des mécanismes de normalisation dans le cadre du processus de réingénierie ?".

Cet objectif a été atteint en proposant d'intégrer la phase d'extraction dans le processus de réingénierie, en comparant le modèle extrait au modèle initialement conçu, en exploitant les écarts entre ces deux modèles dans la re-conception, et en spécifiant un ensemble de règles qui permettent de générer un ensemble de recommandations dans le but de l'évolution et la normalisation du comportement du workflow.

4.8 Conclusion

Dans cette section, nous avons spécifié nos méthode d'analyse statistique des traces d'exécutions pour l'extraction du modèle de processus. Notre analyse statistique nous a permis de construire une table de relations de dépendance entre les tâches. Cette table a été utilisée par la suite comme base pour l'extraction des motifs de workflow formant le modèle de processus à travers un ensemble de règles.

Notre approche commence par la collecte des événements parus lors de l'exécution des instances du processus, et les sauvegarde dans un journal d'événements. Elle construit ensuite, sur la base du calcul statistique, une table spécifiant les dépendances élémentaires entre les tâches. Ces dépendances sont, par la suite, raffinées pour découvrir les motifs de workflow. Ensuite, nous présentons une nouvelle méthode d'extraction de modèle de processus utile à l'identification des motifs fréquents et des erreurs d'anomalies de conception. Finalement, en utilisant ces résultats, nous présentons une méthode d'évolution du comportement initial. L'approche contribue à détecter des erreurs de conception et à faire évoluer la conception initiale sur la base des observations des exécutions des instances du processus. Nous utilisons le terme d'évolution dynamique du système de workflow.

Les points forts de notre approche est la puissance du formalisme étudié, à savoir les réseaux de Petri. Ce formalisme simplifie la tâche de représentation des motifs de tâches. Il est suffisamment formel pour représenter des situations spécifiques, non encore pourvues dans notre approche. Nous pensons en particulier à des boucles indirects.

Par ailleurs, la découverte des motifs de workflow permet de découvrir des comportement plus complexes avec une meilleure spécification des opérateurs de diffusion (*AND-split, XOR-split*) et de jointure (*AND-Join, OR-join*). Elle traite mieux le problème du parallélisme des tâches et des boucles à travers une analyse des tâches dans des intervalles de tailles variables. En effet, la dynamicité de notre algorithme réside dans le fait que la taille de l'intervalle de concurrence n'est pas statique ou fixe, au contraire elle est variable d'une tâche à une autre.

Ce chapitre nous a permis de traiter de le problème de l'extraction de modèles de processus en proposant des méthodes d'analyses statistique des traces d'exécutions. Résumons que quelques mots les spécificités de notre approche :

1. extraction de motifs de comportements à l'aide d'opérateurs de séquence et de concurrence (diffusion et de jointure),
2. enrichissement des comportements de concurrences via les différentes variantes de diffusion et de jointure : AND-split, XOR-Split, AND-Join, OR-Join,
3. extraction de cas particuliers d'itérations entre les tâches comme les boucles,
4. représentation du modèle de workflow par un modèle forme ; les réseaux de Petri. Ce qui nous permet de faire des simulations,
5. spécification des relations exceptionnelles de continuité et de fin entre les tâches nécessaires à la mise en oeuvre de la procédure de normalisation. Nous avons présenté un ensemble de règles qui régissent les contraintes entre les motifs de workflow et d'autre part les relations exceptionnelles.

Nous avons décrit un workflow qui associe à la fois un flot de contrôle classique des instances de processus et une procédure de normalisation. Le flot de contrôle définit l'ordre d'exécution des tâches composantes. La procédure de normalisation définit un flot de contrôle enclenché après des échecs de tâches. Les propriétés des relations exceptionnelles supportent ce flot de contrôle.

Nous avons répondu à un point important de notre travail, à savoir "Comment spécifier et identifier les propriétés conceptuelles du processus métier qui sont corrélées aux problèmes d'évolution et de normalisation ?".

Chapitre 5

Système de workflow

5.1 Introduction

Dans ce chapitre, nous présentons des aspects de réalisation du système de workflow nécessaire à l'expérimentation de notre approche. En premier, dans la première section nous présentons, le cadre général de notre système. Par la suite, dans la deuxième section, nous présentons l'outil de création et de mise à jour du journal événements, via la collecte des traces d'exécutions. Finalement, nous avons implanté, dans la troisième section, nos algorithmes d'extraction de modèles de processus, via le prototype. Afin de simplifier notre présentation, nous avons choisi de ne pas mettre en avant les détails d'implantation en Java.

5.2 Cadre général

Nous proposons d'abord de décrire l'architecture générale du prototype. Ensuite, nous illustrons comment le système de gestion de workflow a été réalisé, pour qu'il puisse générer un modèle de processus et les relations exceptionnelles. Finalement, nous présentons quelques exemples d'implantation.

5.2.1 Architecture générale de l'environnement du prototype

Le cadre général est un système de workflow coopératif qui permet de spécifier, d'exécuter et de contrôler des processus coopératifs. Il fournit un ensemble complet d'outils graphiques intégrés et basés sur JFC/Swing, pour exécuter le processus défini, instancier et commander ce processus, et assurer une interaction entre les utilisateurs et d'autres applications. Le système inclut aussi un navigateur permettant de gérer et de contrôler l'exécution des processus d'une manière interactive. Le système est implémenté en utilisant Java qui fournit un environnement flexible et portable.

L'architecture générale du système est composée comme suit :

La composante de persistance : Les principaux éléments de la composante persistance sont les processus métiers. Un processus représente un projet de workflow. Chaque processus de workflow va contenir essentiellement des informations relatives au nom du processus, à l'utilisateur, à l'état de l'exécution du processus, les tâches, les connecteurs, les données du processus, les acteurs, etc. Afin de séparer les données du projet des éléments qui composent le projet lui même, un bean (beans ou Java bean est un composant logiciel écrit en Java) portable a été développé pour assumer les relations avec les autres entités qui composent la représentation d'un workflow dans le système : tâches, connecteurs, utilisateurs, rôles, propriétés, etc.

L'éditeur graphique : Dans le système, le composant graphique permet au concepteur de représenter et de visualiser le processus de workflow en spécifiant les tâches et les connecteurs. L'utilisateur peut spécifier les attributs d'une tâche (type, description, mode d'exécution, date limite, rôle, etc.). Ce composant permet de créer des projets et de gérer les différentes tâches du projet. Une fois le projet créé et le schéma du workflow défini, les utilisateurs peuvent se connecter afin de définir son état d'avancement. En effet, lorsqu'un utilisateur entre en session, il voit apparaître trois sous fenêtres. La première contient la liste des projets en cours. La deuxième contient la liste des tâches à effectuer. Et la troisième partie contient la liste des tâches en cours d'exécution. L'utilisateur peut alors, en utilisant le menu contextuel, modifier l'état de la tâche ou du projet sélectionné.

Le moteur : Le moteur est constitué d'un composant logiciel java de gestion de sessions, de modes d'exécution et de la liste des tâches. Le composant logiciel de gestion de sessions défi-

nit toutes les actions d'une tâche. Le moteur d'exécution est responsable de gérer les différents modes d'exécution. Il offre aussi une gestion flexible des données et une exécution flexible du workflow. La gestion flexible des données permet aux tâches d'échanger des résultats intermédiaires permettant d'anticiper l'exécution des tâches. Par ailleurs, le moteur introduit des unités de code source qui peuvent être exécutées par le moteur pendant l'exécution. Elles sont associées aux tâches des processus de workflow. Les unités de code source sont écrites en Java. L'utilisateur peut enclencher ces unités de codes n'importe quel instant du cycle de vie d'une tâche.

L'interface : L'interface permet de visualiser l'ensemble des processus métier et des tâches relatives à l'utilisateur en mode session. L'utilisateur peut ensuite, en sélectionnant un processus ou une tâche, accéder à une page contenant les informations relatives. A partir de cette page, l'utilisateur peut modifier certaines caractéristiques comme le rôle, la date de fin, la description et la capacité de la tâche à être anticipée.

Les outils généraux du système : Ils intègrent un grand nombre de procédures de gestion :

- outil de messagerie qui notifie la définition et l'exécution des changements dans un processus de workflow. Chaque interaction d'un utilisateur est notifiée au noyau du système et lance un événement de messagerie.

- Le système impose des dates minimales et limites aux tâches. Ce qui permet de notifier à l'utilisateur qu'une tâche ne se termine pas à sa date attendue.

- L'outil d'alerte qui permet aux utilisateurs de recevoir des notifications en temps réel et échanger différents types de messages.

5.2.2 Fonctionnalités du prototype

Afin de supporter les fonctionnalités de notre prototype, nous avons enrichi le système de workflow pour qu'il puisse prendre en compte l'extraction des modèles de processus et les relations exceptionnelles entre les tâches. Notre approche est d'abord d'intégrer des codes sources au système actuel, et en généralisant leur utilisation pour spécifier les procédures de réactions en face des échecs des tâches. Cette approche est intéressante par sa simplicité. Elle permet la personnalisation du comportement d'une tâche, mais pas l'implémentation du modèle de pro-

cessus. Elle n'est pas généralisable au cadre général d'extraction des modèles de processus. Nous avons adopté une approche complémentaire. Cette approche consiste à mieux rendre indépendant les relations exceptionnelles entre les tâches et le modèle de processus extrait. L'approche complémentaire associe un automate d'états à chaque tâche du processus métier. L'approche détermine l'état suivant d'une étape et est capable d'ajouter de nouveaux états à ceux qui sont définis. Nous présentons, en particulier, les états et les opérations des étapes et nous précisons les éléments de base et les éléments nécessaires pour construire le prototype.

Les états des tâches sont les suivants : initial (Initialement une étape est dans l'état initial), prêt (condition d'activation est valide), run (en cours d'exécution), suspendue (suspendu après une exécution, possibilité de reprise ultérieure), fin (fin avec succès), échec (fin anormale d'une étape ou durée minimale non respecte ou durée maximale expirée), détruit (étape détruite). Plusieurs opérations sont possibles sur les tâches : démarrer (initialiser une étape), activer, suspendre, reprise, terminer, désactiver, détruire. Le comportement d'une tâche est résumé par un tableau de transitions.

En plus des propriétés générales des workflows, le prototype est capable de gérer des propriétés spécifiques pour tout le processus ou pour une tâche particulière. Les propriétés spécifiques peuvent être utilisées pour sauvegarder des données correspondantes au comportement spécifique du processus métier intégré par le prototype.

L'architecture du prototype est composé des modules suivants :

- Le noyau : Ce module permet au moteur d'accéder à tous les éléments du prototype.

- Les tableaux à transitions d'états des tâches : Les types de comportement des tâches sont représentés par des tableaux à transitions d'états.

- Le gestionnaire des changements d'états des connecteurs : Ce module aide le moteur d'exécution à déterminer l'opération à appliquer à une tâche quand ses connecteurs entrants changent d'états.

- Les constantes : Ce module définit les états, les opérations, les types et les comportements des tâches.

Les deux modules précédents de l'architecture s'intéressent au comportement du modèle de processus. Ces modules s'intéressent aux définitions du modèle de processus.

L'interface du prototype est un groupe de méthodes définies dans des classes qui définissent

tous les états et les opérations. L'interface définit aussi les méthodes pour faciliter l'accès aux concepts du moteur, avec la possibilité de supporter des extensions.

Ci-dessous quelques exemples de fonctionnalités du prototype : Rejouer : Cette fonctionnalité permet de gérer la propriété "rejouable" d'une tâche. Cette propriété est définie en spécifiant le nombre maximal et la date maximale et minimale d'activation. Alterner : Cette fonctionnalité permet de gérer la procédure de normalisation d'une tâche. Une tâche normalisable par alternatives possède un ensemble d'emplacement de normalité définissant des chemins d'alternatives.

5.2.3 Synthèse

Dans cette section, nous avons présenté une implémentation possible du comportement du prototype. Concrètement, nous avons présenté comment nous avons adapté un système de workflow existant, via différentes fonctionnalités, pour qu'il puisse supporter différents comportements de notre workflow. Nous avons présenté en particulier les fonctionnalités rejouable et alternative que nous avons définies pour que le système puisse supporter notre approche.

Par ailleurs, l'approche conceptuelle que nous avons choisie pour étendre un système de workflow existant permet d'intégrer n'importe quel fonctionnalité d'extraction de modèles de processus et d'intégration de relations exceptionnelles. Notre implantation a été réalisée dans une logique de supporter des évolutions permanentes du système de workflow hôte.

5.3 Création et mise à jour du journal des événements

Dans cette section, nous présentons l'approche que nous avons adoptée pour récupérer les traces d'exécutions nécessaires à la validation de nos méthodes d'extraction des modèles de processus et de mise en oeuvre de la procédure de normalisation.

Nous avons implanté deux moyens complémentaires : la collecte de traces d'exécutions réelles et la génération de traces d'exécutions simulées. Le premier moyen a été atteint par l'implantation d'un composant logiciel au système de workflow existant pour collecter les traces d'exécutions des instances de processus dans le journal des événements. Le deuxième

moyen, plus simple, utilise un journal des événements créé manuellement.

5.3.1 Journal des événements réels

La composante de persistance du système sauvegarde la trace des tâches des instances de processus métier, ces sauvegardes restent pauvres et manquent cruellement d'informations nécessaires à l'extraction des modèles de processus. Nous avons donc implémenté un composant logiciel d'interface qui met à jour le journal des événements par la sauvegarde de traces enrichies par des informations de type (délai maximum, délai minimum).

Le composant logiciel va générer un journal d'événements qui va contenir des informations se rapportant aux événements qui ont eu lieu lors de l'exécution d'instances de processus. Cet outil permet de collecter tout événement se produisant lors de l'exécution d'une tâche de processus métier.

Parmi les fonctionnalités de l'environnement de notre prototype pour le contrôle des aspects coopératifs se trouve le service de gestion des alertes qui est utilisé par les composants logiciels pour assurer un échange d'informations et d'alertes relatant l'exécution d'une instance de processus. En effet, il s'agit d'un composant de la plate-forme J2EE qui permet de construire des alertes pour la communication des données. La communication entre les utilisateurs et le prototype se fait par des interactions de type création/suppression/mise à jour d'un processus, exécution d'une instance, sa terminaison, etc. Ces moyens de communications sont enregistrés comme des alertes.

Ainsi, les fonctionnalités d'alerte permettent de fournir tous les événements qui viennent de se produire lors de l'exécution des tâches d'une instance de processus. Nous avons, donc, notre composant logiciel de mise à jour du journal des événements qui peut être assimilé à un "tracker" intégré au système de workflow qui collecte toutes les actions réalisées lors de l'exécution d'une instance du processus.

Chaque événement rapporte l'identité de l'instance de processus, l'identité de la tâche exécutée, l'instant de son exécution, etc. Toutes ces informations seront collectées et sauvegardées dans un journal formalisé en XML.

Les fonctionnalités d'alertes reposent sur un mode de communication qui permet d'envoyer

et de recevoir des alertes dans une catégorie particulière, et des brokers de message vont assurer la communication des alertes à chaque utilisateur qui est assigné à cette catégorie. L'alerte possède donc potentiellement plusieurs destinataires. Dans ce cas, l'émetteur de l'alerte ne connaît pas les destinataires. Le composant de mise à jour du journal des événements va se mettre en écoute des alertes échangées lors de l'exécution d'une instance de processus, en s'abonnant à la catégorie concernée. Il va écouter les alertes postées dans cette catégorie et qui se produisent à chaque nouvel événement. Il sauvegarde ces événements dans le journal des événements en format XML. A chaque instanciation d'un processus métier, le composant logiciel produit un journal des événements en XML approprié. Finalement, tous les journaux d'événements associés à un processus métier seront regroupés dans un journal des événements qui regroupe ainsi toutes les traces d'exécutions de ce processus métier.

Après l'ouverture d'une session et l'identification d'un utilisateur, l'éditeur graphique est lancé, c'est là où l'utilisateur peut procéder à l'instanciation du processus. Grâce à l'interface graphique, l'utilisateur arrive à créer de nouveaux processus ou à modifier des processus existants. L'execution d'une instance de processus produit des événements qui sont collectés, produisant les journaux d'événements XML correspondants.

Une fois les processus sont exécutés et leurs traces d'exécutions collectées dans le journal des événements, nous faisons appel à un outil d'accès aux contenus XML. Le but de l'outil est de faciliter la manipulation au sens large de documents XML (lecture d'un document, représentation sous une forme arborescente, manipulation de cet arborescence, exportation vers des nouvelles structures arborescentes, etc.). En particulier, l'outil analyse lexicalement les données XML générées par l'outil de collecte de traces d'exécutions et le traduit au format XML adopté dans notre approche. Il est utilisé comme outil aussi pour parcourir chaque document et extraire des informations sur les activités (tels que leur nombre d'occurrence, la liste des tâches précédentes, suivantes, etc.).

5.3.2 Simulation des traces d'exécutions

Pour tester notre système sur tous ces aspects, les données réelles ne sont suffisantes. en effet, parmi les aspects testés, citons le volume des données et les différents motifs. En effet,

sans les données simulées, il est difficile de progresser dans les meilleures conditions. Obtenir des traces d'exécutions réelles pour des motifs complexes et de grandes tailles s'est avéré une tâche difficile. L'avantage d'exécuter des expériences utilisant la simulation est qu'il est plus facile de commander des variables externes, assurant une meilleure validité interne.

Les modèles sont extérieurement validés en testant et en vérifiant le modèle contre des données résultant de différents exemples de workflow simulés. L'expérimentation est mieux traitée sur les traces d'exécutions simulées, car elle nous permet de mieux couvrir un ensemble très varié de jeux de tests, notamment sur les aspects de motifs variés.

Pour ce faire, nous présentons un outil pour la simulation d'exécution d'instances de processus qui génère des traces d'exécutions. L'idée principale est de créer les traces d'exécutions aléatoires XML en simulant des instances de processus d'un workflow. Nous avons élaboré un outil qui assiste l'utilisateur à générer des traces d'instances d'exécutions en respectant la structure XML.

Par la suite, nous regroupons les traces d'exécutions des différentes instances d'exécution dans un document XML unique qui représente toutes les traces d'exécutions.

5.4 Le prototype

Nous présentons les fonctionnalités de notre prototype qui réalise les méthodes d'extraction de modèles de processus et propose une interface graphique de visualisation. En particulier, nous mettons en oeuvre des outils qui vont exploiter l'ensemble des traces d'exécutions réelles ou simulées pour la validation de notre approche. Pour ce faire, nous avons développé un prototype d'extraction de modèles et de calcul de performance de workflows.

Nous comparons notre prototype avec d'autres outils d'extraction de modèles de processus.

5.4.1 Architecture

Le prototype est un moteur d'analyse de traces d'exécutions d'instances de processus. Le prototype exploite les traces d'exécutions pour extraire le modèle de processus. Ce moteur est spécifié et développé en Java. Il reprend les calculs statistiques et les règles d'extraction

des motifs de processus que nous avons spécifiés dans les chapitres précédents. Notons que le prototype est le fruit d'une collaboration entre le LIFL et l'entreprise SNEDA.

Le prototype est composé en 4 modules :

Transformation des événements : Le module charge les traces d'exécutions, à partir des documents XML, et les transforme dans un format propre au prototype. La source des traces d'exécutions peut être n'importe quel système d'information (système de workflow, de web services, etc.) sujet d'un processus d'extraction et d'analyse de performances. Nous avons construit un composant logiciel qui sauvegarde les données et y optimise l'accès.

Moteur d'analyse : Le module définit le moteur d'analyse des traces d'exécutions. C'est le cœur de notre système. Il extrait les modèles de processus, au travers l'analyse statistique et met en oeuvre la procédure de normalisation sur la base des relations exceptionnelles. Ce module constitue le cœur du prototype.

Extraction des motifs : Le module extrait les motifs de processus en se basant sur les statistiques des corrélations entre les tâches.

Calcul de performances : Le module exploite les relations entre les tâches et les motifs de processus pour mesurer les performances métriques du workflow. Le module mesure la performance des workflows (e.g. durée moyenne d'exécution). Ces métriques mesurent la performance d'un workflow. Elles sont complémentaires au modèle de processus extrait. Ce module ne fait pas partie des objectifs de nos travaux, mais il s'inscrit dans une approche complémentaire.

Pour pouvoir visualiser le modèle de processus extrait, nous avons conçu une interface graphique, qui affiche le résultat sous forme d'un graphe de réseau de Petri. La visualisation est incrémentale, dans le sens où la visualisation se dessine à fur et à mesure que le graphe se concrétise. La visualisation permet à l'utilisateur d'interagir avec le graphe en visionnant une partie et en cachant une autre. Pour ce faire, le prototype offre une interface graphique conviviale contenant des fonctions de visualisation.

La première fonction visualise le journal d'événements sous forme XML. La deuxième, affiche les événements d'une manière simple à partir du journal des événements. La troisième fonction affiche la table de relations déduite du journal des événements. La fonction offre à l'utilisateur la possibilité de modifier/visualiser la taille de la fenêtre de concurrence. Pour

une meilleure visualisation du processus métier, nous avons ajouté la visualisation des valeurs initiales. Et enfin, la quatrième fonction regénère la visualisation d'un graphe. Cette fonction nous permet de visualiser les motifs de workflows extraits. La représentation graphique de ces motifs permet de les isoler et de les étudier d'une manière séparée.

5.4.2 Généralisation

Les outils d'extraction de modèles de processus utilisent généralement différents formats pour collecter ou lire les traces d'exécutions et présentent leurs résultats dans différents modèles de processus. Ceci rend difficile la (re)utilisation de différents outils pour le même ensemble de traces d'exécutions et la comparaison des résultats d'extraction.

Par ailleurs, certains de ces outils mettent en application des concepts qui peuvent être très utiles pour d'autres outils, mais il reste difficile de les combiner et les réutiliser dans des environnements indépendants. En conséquence, les chercheurs travaillant sur des nouvelles techniques d'extraction de modèles de processus se retrouvent à construire une infrastructure d'extraction de modèles de processus et à examiner leurs techniques de façon isolée, se déconnectant des applications réelles.

Pour répondre à ce besoin, nous avons choisi d'implanter notre approche comme un composant logiciel à intégrer dans un progiciel qui fédère un grand nombre d'outils de gestion de processus et contribue à la coopération entre les composants. Ce système offre une certaine flexibilité sur le format d'entrée et de sortie grâce à un ensemble d'outils de conversion. Il est également assez ouvert pour tenir compte d'une réutilisation facile du code pour l'implantation de nouvelles techniques.

Un autre dispositif important dans le progiciel est qu'il permet l'interaction entre un grand nombre de composants logiciels de traitement de traces d'exécutions, ainsi que le support, via un module supplémentaire, d'un format générique XML du journal des événements basé sur une comparaison complète des besoins d'entrée de divers outils d'extraction existants et les traces d'exécutions typiquement produites par les workflows.

Pour l'intégration de notre composant logiciel, nous avons pu profiter de plusieurs composants, essentiellement, le chargement, la lecture, la conversion et le pré-traitement des journaux

d'événements sont pris en charge par des composants génériques.

Nous avons commencé un travail d'implantation pour améliorer l'aspect visuel de notre composant logiciel pour une meilleure interaction du prototype avec l'utilisateur. L'interaction est importante dans un processus de re-conception des processus qui est souvent un interactif où l'interprétation humaine est une connaissance additionnelle importante pour les prises de décision.

La réalisation du composant logiciel nous a permis de mener une étude comparative avec les autres méthodes d'extraction de processus. Cette étude comparative a été la base du tableau comparatif que nous avons présenté dans le chapitre précédent. Notons que notre comparaison était d'ordre qualitatif, ne traitant que des aspects concernant la nature des motifs extraits et des procédures de normalisation mises en oeuvre.

5.5 Expérimentation

5.5.1 Description des données

Pour l'application des tests au données réelles, une recherche de traces et de fichiers logs provenant de processus métiers déjà déployés était nécessaire. La société *Sneda* a accepté de fournir un échantillon de données. Il s'agit de fichiers et de tables représentant l'historique des actions et des tâches effectuées par des collaborateurs dans le progiciel de gestion de cette entreprise. Ces tâches sont de différentes natures vu les services qu'elles couvrent : Service comptabilité, Service juridique, Service construction. Pour les processus métiers, il s'agit souvent de processus de comptabilité, de validation, de numérisation, de tri ou de contrôle.

Code	Description	Type	format
TYPE		Caractère	X(40)
LIBELLE	Description	Caractère	X(40)
NTACHE	Numéro de tache	Entier	»»»»9
LTACHE	Libellé	Caractère	X(40)
ACTEUR	Tache soumise pour (acteur prévu)	Caractère	X(12)
CODREF1	Type de pièce associée à la tache	Caractère	X(10)
REPONSE	Validée, Refusée, Transmise ou Retournée	Caractère	X(2)
ORDONNATEUR	Tache soumise par...	Caractère	X(12)
DATECREATION	Soumise le...	Date	99/99/9999
HEURECREATIO	Soumise à...	Entier	ZZZZ9
DELAI	En jours	Entier	Z9
ACTEUR2	Transmise à...	Caractère	X(12)
DATEREPONSE	Réponse le...	Date	99/99/9999
HEUREREPONSE	Réponse à...	Entier	»»9
NTACHEPREC	Numéro de la tache précédente	Entier	»»»»9
SUPERVISEUR	Réponse par <Acteur réel>	Caractère	X(12)
CMODELE	Code du modèle de flux	Entier	»9
NETAPE	Numéro d'étape du modèle pour cette tache	Entier	»9

Partant de l'idée que les sources de données sont variées, un format XML est recommandé pour les fichiers logs, son schéma est décrit en détails dans [vdAvDH+03].

```
   <!ELEMENT WorkFlow_log (source?, process+)>
<!ELEMENT source EMPTY>
    <!ATTLIST source
        program (staffware|inconcert|pnet|IBM_MQ|other) #REQUIRED>
    <!ELEMENT process (case*)>
   <!ATTLIST process
        id ID #REQUIRED
```

```
    description CDATA 00none00>
<!ELEMENT case (log_line*)>
<!ATTLIST case
    id ID #REQUIRED
    description CDATA 00none00>
<!ELEMENT log_line (task_name, task_instance?, event?, date?, time?)>
<!ELEMENT task_name (#PCDATA)>
<!ELEMENT task_instance (#PCDATA)>
<!ELEMENT event EMPTY>
<!ATTLIST event
    kind (normal|schedule|start|withdraw|suspend|resume|abort|complete)
    #REQUIRED>
<!ELEMENT date (#PCDATA)>
<!ELEMENT time (#PCDATA)>
```

Le système de contrôle de processus est semi-automatique. Il tire partie de l'administrateur comme source d'information sémantique.

5.5.2 Architecture de l'application

Les fichiers journaux sont récupérées à partir des traces sauvegardées par les systèmes transactionnels tels que les *Progiciels de Gestion Intégrés* (PGI), de *Gestion de la Relation Client* (GRC), de *Business to Business* (B2B), de *Gestion de la Chaîne Logistique* (GCL), sans oublier les *Systèmes de Gestion de Workflows.*

5.5.3 Le prototype

Le prototype a été présenté en démonstration dans [ID06]

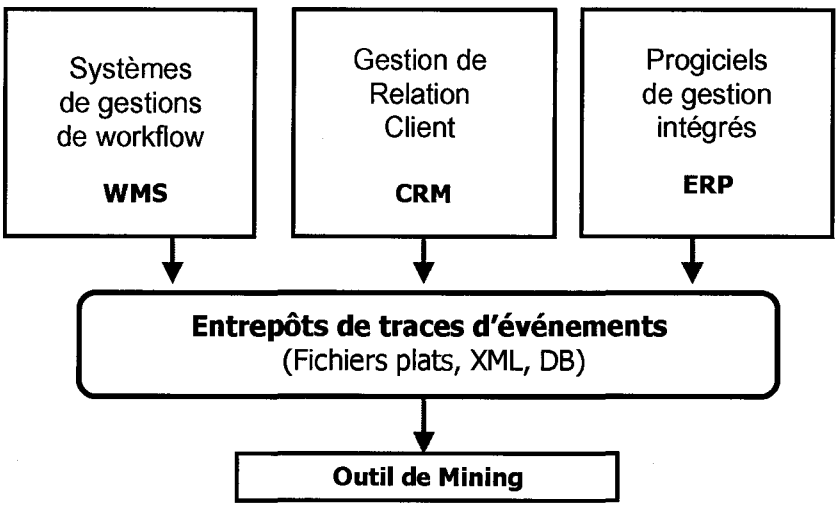

FIG. 5.1 – Architecture de l'application.

5.5.4 Tests de performance

Les tests ont été effectués sur plusieurs modèles de processus. Un jeu de test a été créé en variant les valeurs des paramètres suivants :

- N_T : Nombre de tâches dans le processus.
- N_I : Nombre d'instances d'exécution du processus.
- N_E : Nombre d'événements enregistrés dans le fichier Log.

Le nombre de tâches N_T varie entre 10 et 100.

Le tableau suivant montre les différents temps d'exécution de l'algorithme sur un ensemble de fichiers Logs en fonction du nombre de tâches dans le modèle de Workflow. On distingue deux étapes principales dans l'exécution : le chargement des donnés depuis le fichier log ($T_{Chargement}$) et la procédure d'extraction du modèle de processus ($T_{Traitement}$).

La figure 5.3 montre la croissance du temps d'exécution en fonction du nombre de tâches N_T. Cette croissance est aussi liée au nombre d'instances et d'événements comme le montre la figure 5.4.

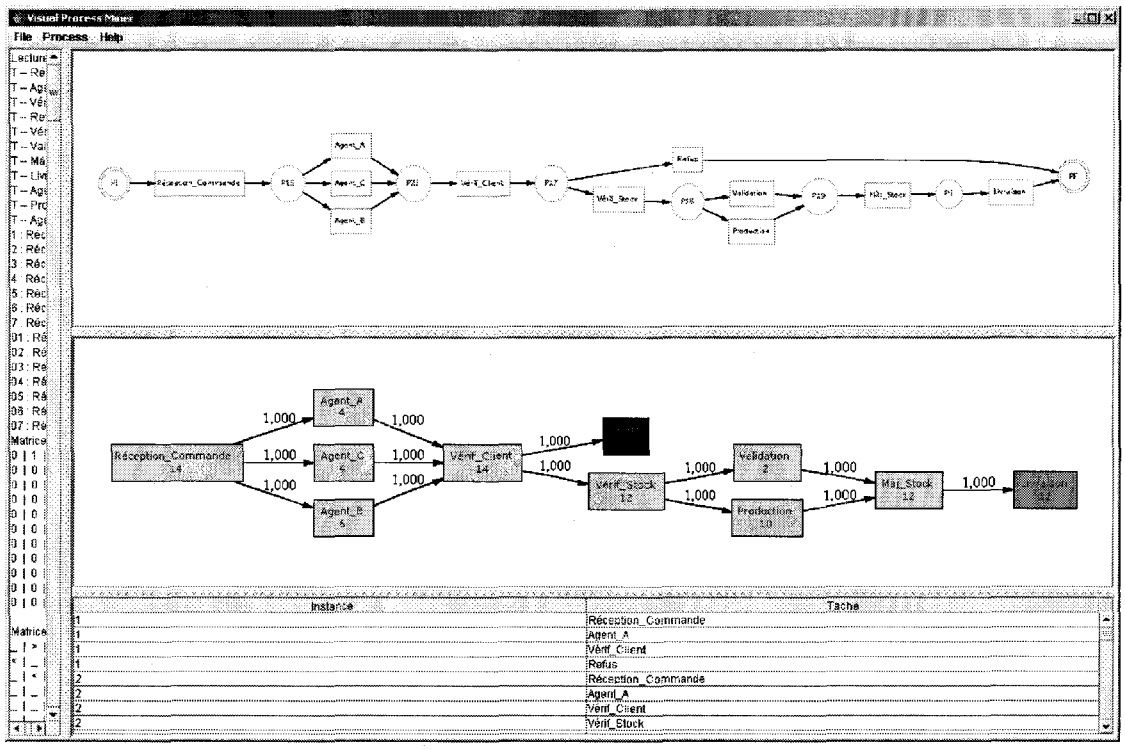

FIG. 5.2 – Outil d'extraction et de visualisation de Processus à partir de fichiers Logs.

En pratique, le nombre de tâches dans les modèles de processus métiers ne dépasse pas la centaine. Cependant le nombre d'instances N_I et le nombre de d'événements dans le fichiers logs N_E sont beaucoup plus grands. Le nombre d'événements N_E s'avère être le facteur discriminant pour le temps d'exécution de l'algorithme.

5.6 Conclusion

Nous avons présenté le prototype de notre méthode et des détails d'implantation nécessaire à l'étude de faisabilité de notre approche et à sa validation. Nous avons proposé une extension d'un système de gestion de workflow classique pour l'intégration de la méthode. Nous avons présenté un outil de collecte de traces d'exécutions simulées ou réelles. Et, nous avons im-

N_T	$T_{Chargement}$(s)	$T_{Traitement}$(s)	T_{Total} (s)
10	58,219	1,625	59,844
20	61,843	5,231	67,074
40	73,406	9,444	82,850
60	100,502	11,731	112,233
80	120,845	22,892	143,737
100	150,178	48,610	198,788

TAB. 5.1 – Différents temps d'exécution en fonction du nombre de tâches.

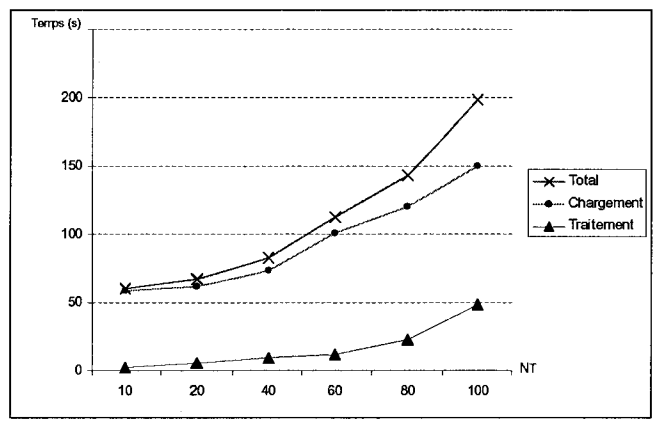

FIG. 5.3 – Graphe du Temps d'exécution en fonction de N_T.

planté notre méthode d'extraction de modèles de processus. A ce stade, nous pouvons formuler deux limites à notre approche : La première limite concerne la relation d'équivalence entre le journal des événements et le modèle de workflow extrait dans le cas d'un facteur d'échelle important. En effet, comment prouver que le modèle de processus généré est équivalent au journal des événements. La question prend tout son sens lorsque le journal des événements est volumineux, voire complexe. La deuxième limite concerne la nécessite de mieux intégrer les différentes règles d'évolution que nous avons proposées dans le chapitre précédent. Ainsi nous visons à l'implantation d'un composant logiciel dont le rôle sera d'exploiter le modèle extrait pour proposer des suggestions d'évolution aux concepteurs d'une manière interactive. Cepen-

N_T	$T_{Chargement}$(s)	$T_{Traitement}$(s)	T_{Total} (s)
10	58,219	1,625	59,844
20	61,843	5,231	67,074
40	73,406	9,444	82,850
60	100,502	11,731	112,233
80	120,845	22,892	143,737
100	150,178	48,610	198,788

TAB. 5.2 – Différents temps d'exécution en fonction du nombre d'événements.

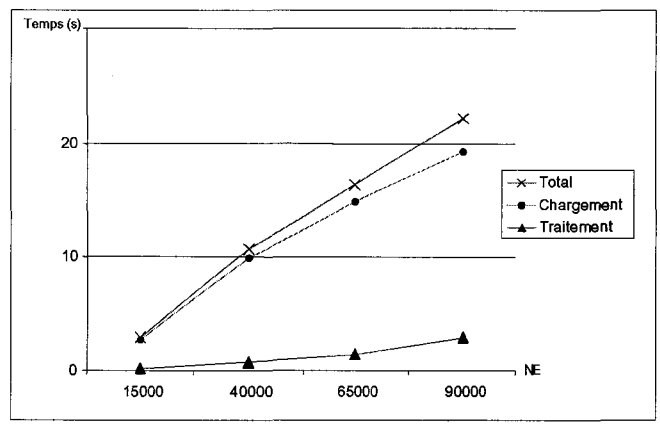

FIG. 5.4 – Graphe du Temps d'exécution en fonction de N_E.

dant le défis majeur auquel nous faisons face afin de répondre à ce besoin est l'acquisition de cas d'utilisation réels contenant des anomalies conceptuelles causantes des échecs de tâches. En effet, l'exploitation de traces d'exécution simulées pour la validation de notre méthode est une étape de validation. Une étape supplémentaire de validation sur des données réelles est nécessaire pour compléter la validation de l'approche.

Chapitre 6

Conclusion

6.1 Contribution

Les processus métiers sont actuellement créés et gérés par les systèmes de gestion de workflow. Ces derniers, quoique populaires sur le marché, souffrent de limites importantes dans l'évolution et la normalisation, suite à leur mise en exploitation. La conception reste un processus statique, c'est à dire un processus qui ne tient pas compte des usages réels, et donc des véritables besoins des utilisateurs.

Afin de répondre au besoin d'évolution du modèle de processus initial, suite aux observations, nous avons spécifié une nouvelle étape d'évolution et de correction du comportement du modèle initial par l'analyse du journal des événements.

Notre approche commence par l'extraction du modèle de processus, en composant des motifs. Le modèle extrait est ensuite augmenté par des procédures de normalisation du workflow initialement conçu. La comparaison entre le modèle de processus initial et celui qui est extrait relève des écarts entre la conception initiale et la réalité des usages. Cet écart peut signifier un besoin d'évolution du processus métier initial. Il peut signifier aussi que les utilisateurs ne respectent pas le processus métier initialement conçu. Quelque soit la raison, sur la base de cet écart, notre approche propose un ensemble de suggestions pour faire évoluer le schéma de workflow initial afin mieux répondre à la réalité des usages. Bien entendu, de telles suggestions

doivent être validées par l'administrateur. Ainsi, les évolutions induites par ces suggestions ne sont pas faites de façon automatique mais plutôt interactive avec l'administrateur du workflow.

Notre approche contribue aux méthodes de conception de processus métier en se basant sur des faits réels et non sur des informations subjectives qui ne coincident pas toujours avec la réalité. Notre approche simplifie le diagnostic du processus, en proposant des évolutions du comportement des processus métiers, en meilleure adéquation avec la réalité des usages.

Par ce travail, nous contribuons à l'évolution et à la normalisation des processus métier, à travers :

- l'enrichissement du journal des événements par des durées minimales et maximales. Notre philosophie est d'analyser le journal des événements avec le moins d'informations possibles des événements du journal (identité de l'instance, instant de fin, durée minimale, durée maximale).

- l'enrichissement de la description du modèle de workflow par des motifs de processus (diffusion (AND/XOR-Split) et jointure (AND/OR-Join)) et par des relations exceptionnelles (de continuité et de fin) entre les tâches,

- la prise en compte des procédures de normalisation en cas d'échec de tâches, afin de normaliser au maximum les instances de processus. Nous ajoutons ainsi au niveau de représentation des modèles de processus, un niveau de représentation des relations exceptionnelles. La prise en compte à la fois de l'extraction du modèle de processus, enrichi des motifs de processus, et de la procédure de normalisation est un point fort additionnel de notre travail de thèse.

Pour terminer, nous soulignons les deux méthodes complémentaires d'acquisition de traces d'exécutions nécessaires à la validation de nos méthodes d'extraction des modèles de processus et de la procédure de normalisation. La première méthode nous a menée à implanter un outil de collecte de traces d'exécutions réelles au système de gestion de workflow existant. En effet, lors de l'exécution du workflow, le système collecte l'ensemble des événements enregistrés traduisant l'exécution réelle du workflow. Ces évènements sont sauvegardés selon le format XML. La deuxième méthode est proposée pour satisfaire la complétude du panel d'exemples de workflow nécessaires aux tests de validation de nos approches. Cette méthode s'inscrit dans une approche de simulation de traces d'exécutions qui nous permet de mieux tenir compte de

la complexité des modèles de processus.

Nous avons présenté l'implantation nécessaire à la validation de nos méthodes d'extraction de motifs de processus. Elle consiste au développement du prototype qui se base sur des journaux d'événements en XML pour l'analyse des traces d'exécutions et l'extraction de modèles de processus. Ce prototype propose aussi un composant d'analyse de performances.

6.2 Perspectives

Nous regroupons les perspectives en deux grands groupes : Le premier groupe de perspectives concerne des extensions de l'approche. Le deuxième groupe se focalise sur de nouveaux motifs de comportements.

6.2.1 Extensions

Une première perspective concerne la prise en compte du contexte dans l'extraction des modèles de processus. Plusieurs questions sont ouvertes : comment définir le contexte ? comment le prendre en compte dans la représentation du journal des événements ? comment l'utiliser dans l'extraction des modèles de processus, etc. L'idée du contexte est de dire qu'une tâche donnée s'exécute efficacement ou non selon telle ou telle information sur le contexte de la tâche (par exemple le service qui le traite, la période, etc.). Ou encore, l'échec d'une tâche est corrélée à un événement contextuel (e.g. période de vacances, situation géographique d'un service).

Une deuxième perspective concerne la méthode elle même. Si notre approche s'appuie sur une analyse statistique (table des relations) entre les tâches pour extraire les motifs de processus, il serait intéressant d'étudier d'autres méthodes de fouille de données, notamment des méthodes d'extraction des motifs.

Une troisième perspective concerne l'outil formel de simulation : Une autre manière de collecter les traces serait d'utiliser un réseau de Petri. En effet, si le réseau de Petri a été généré pour représenter le modèle de processus, il pourrait être utilisé pour représenter et simuler les scénarios de processus, et engendrer automatiquement des traces d'exécutions pour chaque

instance exécutée. Cette perspective implique des adaptations appropriées au niveau de la modélisation des déclarations des workflows en réseaux de Petri. Les déclarations sont modifiées pour importer des fonctions pour collecter les traces d'exécutions de chaque transition. Ces fonctions indiquent en particulier l'endroit, le préfixe et l'extension des documents XML que le réseau de Petri crée pour chaque instance qu'il exécute. Une fois que les déclarations du workflow en réseau de Petri mises à jour, des déclarations relatives aux transitions sont modifiées pour appeler les fonctions de collecte des traces d'exécution. Le workflow en réseau de Petri composé crée le journal des événements qui trace les instances de processus et les représente en XML.

Une quatrième perspective concerne les métriques. L'idée est de combiner les performances issues de métriques et les modèles de processus extraits. Les métriques exploitent au mieux les relations entre les tâches et les motifs de processus pour mesurer les performances métriques du workflow. Ces métriques (e.g. durée moyenne d'exécution) mesurent la performance d'un workflow. Elles sont complémentaires au modèle de processus extrait.

Une cinquième perspective concerne l'étude de notre approche sur des comportements réels de processus. L'idée adjacente à cette perspective est de corréler des motifs de comportement aux applications visées. Cela induit, la mise en oeuvre de procédures qui tracent les instances de processus d'un système d'information. Il est question aussi de qualifier le modèle de processus extrait. La quantité des événements qui respectent un motif ou le degré d'écart entre un modèle de processus extrait et le processus initial peut induire sur la qualité du modèle extrait. Par ailleurs, la combinaison du modèle extrait et d'autres sources (e.g. des réponses à des questionnaires, peuvent contribuer à une meilleure qualification du modèle de processus métier.

Enfin, la dernière perspective est d'étudier notre approche dans le contexte applicatif de la fouille des usages des données multimédia. Un thème de recherche actif de notre équipe du LIFL. Le but est de construire des modèles de processus d'accès aux contenus multimédia pour améliorer les accès aux contenus. Un des défis majeurs est la collecte des traces d'exécutions des services multimédia. Pour ce faire, nous avons mené un travail que nous venons de co-publier. Dans ce travail, nous proposons une première solution de collecte des fixations des regards. Ces traces d'exécution sont par la suite exploitées pour la compréhension

des comportements oculaires des utilisateurs. L'objectif ultime est d'automatiser les processus d'extraction des modèles d'accès aux contenus multimédia.

6.2.2 Motifs de comportements

Plusieurs motifs, relativement complexes, méritent d'être explorés. Les motifs présentés ci-dessous représente un échantillon que nous estimons important dans la richesse des modèles de processus.

Motif de Discrimination :

Ce motif modélise des structures composées d'un point dans le processus de workflow qui attend la complétude d'une branche avant d'activer une tâche. Exemples : Pour améliorer le temps de réponse d'une requête, celle-ci est envoyée à deux différentes bases de données sur le web. La première réponse obtenue de l'une des deux bases de données est prise en compte, la deuxième est ignorée. La plupart des workflow n'implantent pas une telle fonctionnalité.

Les workflows comme Staffware et i-Flow n'acceptent pas qu'une seconde instance de la tâche soit active. Elle ne peut pas être utilisée pour implanter le motif de discrimination. Il y a cependant des tentatives plus ou moins complexes de mise en oeuvre de ce motif dans les systèmes comme Verve, SAP R/3, etc. Par exemple dans le workflow SAP R/3, il est possible de spécifier le nombre de branches qui doivent être complètes dans le motif de diffusion (AND-Split et AND-Join). Sur la base de ce nombre, nous pouvons réaliser un motif de discrimination. Il suffit de mettre la valeur à 1, ainsi la branche non complète reçoit l'état de "fin", et le motif restreint le parallélisme.

En revanche, réaliser le motif dans une boucle est très complexe. Ce qui expliquerait peut être que la majorité des workflow ne l'ont pas implanté.

Fin explicite :

Les tâches du processus sont terminées si elles n'ont rien d'autres à faire. En d'autres termes, à la fin de l'instance de processus, aucune tâche ne doit être active, pour autant l'instance de processus n'est pas dans un état bloqué. Dans la plus part des moteurs de workflow, le système termine toutes les tâches lorsque un point de synchronisation final est atteint. Toutes les tâches actives à cet instant sont détruites.

Certains moteurs de workflow (e.g. Staffware, MQSeries Workflow) supportent ce motif. Et pour les moteurs de workflow qui ne supportent pas directement ce motif, l'approche adoptée est souvent de transformer la représentation du modèle de processus en une autre représentation cible qui se termine par un nœud final. La complexité d'une telle tâche dépend beaucoup de la représentation courante. Certains moments, elle est simple, en utilisant une combinaison de motif de jointure. Cependant, il existe d'autres cas où c'est extrêmement compliqué, voire impossible à réaliser. Une représentation qui comprend des instances multiples et une fin implicite est typiquement une situation qui est très difficile pour la conversion.

Bibliographie

[AGL98] R. Agrawal, D. Gunopulos, and F. Leymann. Mining process models from workflow logs. *In Hans-Jörg Schek, Fèlix Saltor, Isidro Ramos, and Gustavo Alonso, editors, EDBT,*, volume 1377 of Lecture Notes in Computer Science :pages 469–483, 1998.

[BEA04] BEA. Bea systems - bea weblogic integration. online at http ://www.beasys.com/content/products/weblogic/integrate/ (October 2007), 2004.

[CW98] J.E. Cook and A.L. Wolf. Discovering models of software processes from event-based data. *ACM Transactions on Software Engineering and Methodology*, 7(3) :pages 215–249, 1998.

[CW99] J. E. Cook and A. L. Wolf. Software process validation : quantitatively measuring the correspondence of a process to a model. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 8(2) :pages 147–176, 1999.

[Dav01] B. David. Ihm pour les collecticiels. *Réseaux et Systèmes Réparties (RSR-CP), Hermes Science*, vol. 13 :pages 169–206, 2001.

[dMWvdA05] A. K. A. de Medeiros, A. J. M. M. Weijters, and W. M. P. van der Aalst. Genetic process mining : A basic approach and its challenges. *In Business Process Management Workshops*, pages pages 203–215, 2005.

[DNR90] M. Downson, B. Nejmeh, and W. Riddle. Concepts for process definition support. in proceedings of the sixth international software process workshop, hakodate, japan. *IEEE Computer Society Press*, pages pages 87–90, 1990.

[ECO01] ECOO. Projet ecoo : environnements et coopération. Rapport d'activité, INRIA, 2001.

[EGR91] C. A. Ellis, S.J. Gibbs, and G.L. Rein. Groupware : Some issues and experiences. *Communications of the ACM, ACM Press*, vol. 34 :pages 38–58, 1991.

[FH92] P.H. Feiler and W.S. Humphrey. Software process development and enactment : Concepts and definitions. *Technical Report SEI-92-TR-004, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA*, 1992.

[GGMS03] G. Greco, A. Guzzo, G. Manco, and D. Sacca. Mining frequent instances on workflows. *Proceedings of the 7th Pacic-Asia Conference "Advances in Knowledge Discovery and Data Mining", PAKDD 2003, Seoul, Korea.*, LNCS 2637 :pages 209–221, 2003.

[GGP05] G. Greco, A. Guzzo, and L. Pontieri. Mining hierarchies of models : From abstract views to concrete specifications. *In Business Process Management*, pages pages 32–47, 2005.

[GHS95] D. Georgakopoulos, M. Hornick, and A. Sheth. An overview of workflow management : From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases*, 3(2) April :pages 119–153, 1995.

[GP03] M. Golani and S. S. Pinter. Generating a process model from a process audit log. *In Business Process Management*, pages pages 136–151, 2003.

[GT98] D. Georgakopoulos and A. Tsalgatidou. Technology and tools for comprehensive business process lifecycle management. in dogac, a., kalinichenko, l., ozsu, t., and sheth, a. *Workflow Management Systems and Interoperability*, NATO SI Series F. Springer-Verlag, 1998.

[Ham90] M. Hammer. Reengineering work : Don't automate, obliterate. *Harvard Business Review*, pages pages 70–91, 1990.

[HK04] J. Herbst and D. Karagiannis. Workflow mining with inwolve. *Computers in Industry*, 53(3) :pages 245–264, 2004.

[IBM04] IBM. "ibm software - websphere mq workflow". Online at : http ://www-306.ibm.com/software/integration/wmqwf/ (october 2007), 2004.

[ID06] N. Ihaddadene and C. Djeraba. Extraction de modèle de processus à partir de journaux d'événements. *22èmes Journées Bases de Données Avancées (BDA'2006)*, 2006.

[JVvdAR06] M. H. Jansen-Vullers, W. M. P. van der Aalst, and M. Rosemann. Mining configurable enterprise information systems. *Data & Knowledge Engineering*, 56(3) :pages 195–244, 2006.

[KPLB01] M. Kirsch-Pinheiro, J.V. Lima, and M.R.S. Borges. Awareness em sistemas de groupware. *Proceedings of IDEAS'01, Centre de Información Tecnológica (CIT), San Diego, Costa Rica*, pages pages : 323–335, 2001.

[Lau02] Y. Laurillau. *Conception et réalisation logicielles pour les collecticiels centrées sur l'activité de groupe : le modèle et la plateforme Clover.* PhD thesis, Thèse de Doctorat, Institut National Polytechnique de Grenoble, Grenoble, France, septembre 2002.

[LR99] F. Leymann and D. Roller. Production workflow, concepts and techniques. *Prentice-Hall PTR*, ISBN 0-130-21753-0 :479 pp, 1999.

[LS97] K. Lei and M. Singh. A comparison of workflow metamodels. *In Proceedings of the Workshop on Behavioral Modeling and Design Transformations : Issues and Opportunities in Conceptual Modeling at the 16th International Conference on Conceptual Modeling / the Entity Relationship Approach (ER'97), Los Angeles, CA*, 1997.

[LS03] P.G. Lopez and A.F.G. Skarmeta. Ants framework for cooperative work environments. *Computer, IEEE Computer Society*, Vol. 36, N° 3 :pages 56–62, 2003.

[Man98] M.L. Manheim. Beyond groupware and workflow : The theory of cognitive informatics an dits implications for a people-based enterprise information architecture. 1998.

[McC92] S. McCready. There is more than one kind of workflow software. *Computerworld*, November 2 :pages 86–90, 1992.

[RD98] M. Reichert and P. Dadam. Adeptflex - supporting dynamic changes of workflows without losing control. *Journal of Intelligent Information Systems, Special Issue on Workflow Management*, vol. 10 :pages 93–129, 1998.

[RRD04] S. Rinderle, M. Reichert, and P. Dadam. Correctness criteria for dynamic changes in workflow systems - a survey. *In Data & Knowledge Engineering (Special issue on advances in business process management)*, 50(1) :pages 9–34, 2004.

[SABS02] H. Schuldt, G. Alonso, C. Beeri, and H. Schek. Atomicity and isolation for transactional processes. *ACM Trans. Database Syst.*, vol. 27, no. 1 :pages 63–116, 2002.

[Sal98] M.R. Salcedo. *Alliance sur l'Internet : support pour l'édition coopérative des documents structurés sur un réseau à grande distance.* PhD thesis, Thèse de Doctorat, Institut National Polytechnique de Grenoble, Grenoble, France, 1998.

[Sch02] G. Schimm. Process miner - a tool for mining process schemes from event-based data. *In European Conference on Logics in AI*, pages pages 525–528, 2002.

[Sch04] G. Schimm. Mining exact models of concurrent workflows. *Computers in Industry*, 53(3) :pages 265–281, 2004.

[SO99] W. Sadiq and M. Orlowska. On capturing process requirements of workflow based business information system. *In proceedings of 3rd International Conference on Business Information Systems (BIS '99)*, 1999.

[TB97] F. Tarpin-Bernard. *Travail coopératif synchrone assisté par ordinateur : approche AMF-C.* PhD thesis, Thèse de Doctorat, Ecole Centrale de Lyon, Lyon, France, 1997.

[VBvdA01] H. M. W. Verbeek, T. Basten, and W. M. P. van der Aalst. Diagnosing workflow processes using woflan. *The Computer Journal*, 44(4) :pages 246–279, 2001.

[vdA98a] W. M. P. van der Aalst. Three good reasons for using a petri-net-based workflow management system. *The Kluwer International Series in Engineering and Computer Science : Information and Process Integration in Enterprises : Rethinking Documents*, Chapter 10, 428 :pages 161–182, 1998.

[vdA98b] W.M.P. van der Aalst. The application of petri nets to workflow management. *Journal of circuits, Systems, and Computers*, vol. 8, no. 1 :pages 21–66, 1998.

[vdARS05] W. M. P. van der Aalst, H. A. Reijers, and M. Song. Discovering social networks from event logs. *Computer Supported Cooperative Work*, 14(6) :pages 549–593, 2005.

[vdAtH05] W. M. P. van der Aalst and A.H.M. ter Hofstede. Yawl - yet another workflow language. *Information Systems,*, vol. 30(4) :pages 245–275, 2005.

[vdAvD02] W.M.P. van der Aalst and B.F. van Dongen. Discovering workflow performance models from timed logs. *In 1st International Conference on Engineering and Deployment of Cooperative Information Systems*, pages pages 45–63, 2002.

[vdAvDH+03] W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow mining : A survey of issues and approaches. *Data and Knowledge Engineering*, vol. 47, no. 3 :pages 237–267, 2003.

[vdAvH02] W.M.P. van der Aalst and K. van Hee. Workflow management : Models,methods, and systems. *The MIT Press*, ISBN 0-262-01189-1 :368 pp, 2002.

[WfM99] WfMC. Terminology and glossary. technical report wfms-tc-1011. Technical report, WorkFlow Management Coalition, 1999.

[WfM02] WfMC. Workflow management coalition, workflow process definition interface, xml process definition language. version 1.0. WFMC-TC-1025, October 2002.