République Algérienne Démocratique et Populaire

Université Mentouri - Constantine

Faculté des Sciences et de l'ingénieur

THÈSE

Pour l'obtention du

Numéro d'ordre:
Numéro de Serie:

Doctorat en Sciences

Spécialité: Informatique

Présentée par

Mounira ZERARI

Une approche pour l'amélioration de la flexibilité des processus métier basée sur les techniques du process mining

---

Directeur de thèse : **Pr.Mahmoud Boufaida**

Soutenue publiquement à Constantine le 12 Mars 2012

Devant le jury composé de:

JURY

Pr.Zaidi Sahnoun	Université Mentouri, Constantine	Président
Pr.Mahmoud Boufaida	Université Mentouri, Constantine	Rapporteur
Dr.Nadia Zeghib	Université Mentouri, Constantine	Examinateur
Dr.Hassina Seridi	Université Baji Mokhtar, Annaba	Examinateur
Dr.Abdelmajid Zidani	Université de Batna	Examinateur

A mes Parents qui, sans eux rien n'aurait été possible.
A mes frères : Karim, Yasser et Kamel eddine.
A l'homme qui partage ma vie : Redha.

Remerciements

Je tiens à remercier, tout d'abord, mon directeur de thèse

Monsieur **Mahmoud Boufaida**, Professeur à l'Université Mentouri de Constantine, qui a dirigé cette thèse, m'a conseillée tout au long de son élaboration dans un cadre de travail très favorable. Je le remercie également d'avoir consacré beaucoup de son temps pour les nombreuses relectures de mon document et les articles publiés. Ses remarques m'ont permis de faire progresser ce travail.

Je tiens également à remercier les membres du jury qui m'ont fait l'honneur de bien vouloir évaluer mon travail, et plus précisément :

Monsieur **Zaidi Sahnoun**, Professeur à l'Université Mentouri de Constantine, pour l'honneur qu'il m'a fait, en acceptant la présidence de ce jury.

Madame **Nadia Zeghib**, Maitre de conférence A à l'Université Mentouri de Constantine, pour avoir accepté d'évaluer le présent document.

Madame **Hassina Seridi**, Maitre de conférence A à l'Université Baji Mokhtar d'Annaba, pour avoir accordé de son temps précieux pour l'analyse de mon humble travail et les suggestions constructives pour l'amélioration de ce document.

Monsieur **Abdelmagid Zidani** , Maitre de Conférence A à l'Université de Batna pour avoir bien voulu se pencher sur ce travail et faire partie de ce jury. Je le remercie également pour le temps qu'il m'a accordé et pour ses commentaires constructifs qui ont permis d'améliorer la qualité de mon document.

Table des matières

Table des figures **vii**

Liste des tableaux **ix**

Introduction générale **1**

1 Contexte de la thèse 1
2 Problématique 4
3 Objectifs et contributions de la thèse 5
4 Organisation de la thèse 8

I État de l'art **9**

1 Introduction 9
2 Notion de processus métier 9
2.1 Concept de processus métier 10
2.2 Processus métier VS Workflow 11
2.3 Gestion des processus métier 12
2.4 Cycle de vie d'un processus métier 16
3 Process-Aware Information Systems 19
3.1 Définition 19
3.2 Apport à la gestion des processus métier 19
3.2.1 Les systèmes Personne à Personne 20
3.2.2 Les systèmes Application à Application 20
3.2.3 Les systèmes Personne à Application 21

3.2.4 Prévisibilité des processus métier 21
3.3 Discussion 23
4 Spécificité et apport des processus métier flexibles 23
4.1 Définition de la flexibilité 24
4.2 Gestion des processus métier flexibles 25
4.3 Discussion 29
5 Analyse, diagnostic et découverte des processus métier 31
5.1 Re-ingénierie des processus métier 32
5.2 Découverte des processus métier 34
5.3 Fonctionnalités du "Process mining" 36
5.4 Techniques et formalismes du process mining 38
5.5 Apport du process mining sur la flexibilité des processus métier 40
6 Applications sensibles au contexte 42
6.1 Définition du contexte 42
6.2 Présentation de la notion du "Context-aware computing" 43
6.3 Caractéristiques des applications sensibles aux contextes 44
6.3.1 Acquisition du contexte 44
6.3.2 Modélisation des informations de contexte 45
6.4 Discussion 47
7 Conclusion 48

II Une solution sensible au contexte pour la flexibilité des processus métier 49

1 Introduction 49
2 Travaux antérieurs 50
2.1 Taxonomie de la flexibilité des processus métier 50
2.2 Classification des méthodes et solutions pour la flexibilité 51
2.3 Synthèse 53
3 Présentation globale de l'approche proposée 54
3.1 Vue locale du système 55
3.2 Vue globale du système 55
3.3 Relation entre vue locale et vue globale 56
4 Représentation du processus métier 56
4.1 Vers une décomposition du processus métier 57
4.2 Processus métier : Regroupement d'entités 59
5 Récupération du contexte d'exécution 60

5.1 Acquisition du contexte . . . 61
5.2 Interprétation et analyse du contexte . . . 61
5.3 Adaptation aux changements de contexte . . . 61
6 Spécification des règles métier . . . 62
7 Conclusion . . . 64

III Une architecture basée artifact : exploitation du contexte d'exécution 67
1 Introduction . . . 67
2 Les artifacts d'exécution . . . 68
2.1 Besoin d'une nouvelle entité . . . 68
2.2 Le concept d'artifact . . . 69
2.3 Utilisations possibles . . . 70
2.4 Instructions opératoires . . . 72
2.5 Discussion . . . 73
3 Description architecturale . . . 74
3.1 Aperçu de l'architecture proposée . . . 74
3.2 Structure de l'artifact . . . 77
3.3 Instructions opératoires de l'artifact métier . . . 77
4 Roles et comportements des différents composants . . . 79
4.1 Composant process mining . . . 79
4.2 Composant sensible au contexte d'exécution . . . 82
5 Conclusion . . . 84

IV Etude de cas et implémentation 87
1 Introduction . . . 87
2 Énoncé de l'étude de cas . . . 87
2.1 Caractéristiques des systèmes «Health-care» . . . 88
2.2 Présentation de l'étude de cas : Les systèmes "Health-care" . . . 89
2.3 Le langage YAWL : Yet Another Workflow Language . . . 91
2.4 La modélisation du processus métier avec YAWL . . . 91
3 Implémentation de l'architecture proposée . . . 92
3.1 Collecte de traces : Génération des fichiers log . . . 94
3.2 Extraction du contexte d'exécution . . . 95
3.3 Le langage Jason et le framework CArtAgO . . . 96

3.4 Utilisation de CArtAgO et Jason pour implémenter les artifacts et le raisonneur 100
3.5 Invocation des artifacts lors de la prise de décision 101
4 Discussion 102
5 Conclusion 104

Conclusion générale **105**

Références bibliographiques **109**

Table des figures

1 Séparation de la logique du processus du code dans un système de gestion de processus [Rinderle 2004]. . . . 3
2 Vue globale de l'approche proposée. . . . 6

I.1 Différents aspects d'un processus métier. . . . 12
I.2 Caractéristiques du BPM. . . . 13
I.3 Système d'information pour les entreprises. . . . 14
I.4 Cycle de vie de workflow et cycle de vie BPM. . . . 16
I.5 Catégorisation des systèmes PAISs avec les outils développés associés[Russell 2007]. 22
I.6 Les trois dimensions d'évaluation de l'entreprise d'après l'espace de processus. 24
I.7 Position de FLOWer pour la prise en charge des processus structurés 28
I.8 Représentation des solutions existantes selon les critères fonctionnels. . . . 30
I.9 Le niveau de support du cycle de vie BPM.[Gaaloul 2006] 31
I.10 La re-ingénierie des processus métier 33
I.11 La découverte de workflow 35
I.12 Les trois types du process mining : (1) Découverte (2) Conformité (3) Amélioration [van der Aalst 2009a] 37
I.13 Les différentes étapes dans l'analyse de processus. . . . 38
I.14 Description graphique du schéma MXML. [de Medeiros. 2006a] 40
I.15 Niveaux d'abstraction du changement. . . . 41

II.1 Différentes vues de l'approche proposée 55
II.2 Exemple d'un composant métier 57
II.3 Représentation d'un ensemble de processus en diagramme de caractéristiques . 58

Table des figures

II.4 Représentation d'un processus métier de manière classique 58
II.5 Représentation du regroupement d'entités pas l'Administrateur 59
II.6 Représentation des fonctionnalités d'un système sensible au contexte. 60
II.7 Exemple de variable de contexte d'une actitvité selon[Crerie 2009]. 64

III.1 Représentation métaphorique du concept d'artifact 69
III.2 Représentation abstraite d'un artifact 71
III.3 Architecture générale du système proposé 75
III.4 Diagramme d'état transition représentant les différents états d'une instance. . 78
III.5 Un framework sensible au contexte pour la flexibilité des processus métier. . . 80
III.6 Les phases de pré-analyse et d'analyse 81

IV.1 Architecture d'un système "health-care". 89
IV.2 Structure de l'étude de cas "Admission d'un patient" 90
IV.3 Modélisation par YAWL du processus métier "Admission d'un patient" 92
IV.4 Scénario d'exécution du système 93
IV.5 L'ensemble des tables de la base de données 94
IV.6 Fichier MXML obtenu après conversion 95
IV.7 Diagramme de classe d'implémentation du plug-in Business rules extraction. . 96
IV.8 Codification des différentes classes du plug-in Business rules extraction. 97
IV.9 Le schéma XML de la partie règles ECAPE extension de ECA. 98
IV.10 La classe artifact 100
IV.11 La définition du MAS avec le module raisonneur 100
IV.12 La définition de l'artifact "test_blood" 101
IV.13 La définition de l'agent "raisonneur" 102

A.1 Diagramme d'état transition représentant les différents états d'une instance. . 122

Liste des tableaux

I.1 Synthèse sur les approches orientées modèle 46

II.1 Evaluation des produits considérés flexibles 53

III.1 Catégories de variables de contexte 83

IV.1 Projection de notre solution sur les critères de solutions flexibles 103

Introduction générale

1 Contexte de la thèse

De nos jours, pour la majorité des entreprises et organisations, l'automatisation des processus métier est devenue de plus en plus importante pour l'atteinte de la performance imposée par la concurrence dans le marché et par l'évolution des moyens matériels notamment informatiques.

Cette automatisation concerne autant les systèmes d'applications traditionnelles (ex : Système de planification de ressources) que les systèmes d'applications e-business évolutives (ex : e-procurement, supply chain management). Un support compréhensif et clair des processus est alors requis par les utilisateurs. Les systèmes d'information sensibles au processus [1] [Rinderle 2004] devront permettre la définition explicite de la logique des processus, coordonner leur exécution, leur supervision ainsi que l'intégration des composants distribués de manière robuste et sécurisée.

Dans un tel contexte, l'utilisation croissante des systèmes de gestion des processus métier dans les entreprises exprime alors l'importance indéniable de cette automatisation des processus. Ces systèmes de gestion sont classés parmi les systèmes les plus élaborés pour définir et exécuter des processus. Ils permettent, en particulier, de décrire explicitement les méthodes de travail réalisant un processus, de les expérimenter, de mesurer leur qualité et de les optimiser afin de pouvoir assurer l'amélioration et la réutilisation des processus. Durant cette dernière décennie, les systèmes de gestion de processus métier se sont répandus, offrant un fort potentiel de modélisation et d'automatisation.

[1] Connu sous le nom de Process-Aware Information Systems

Gestion des processus métier

Il n'est plus besoin de justifier que, la transition des systèmes d'information centrés-données vers des systèmes d'information dirigés vers les processus, prenne une place prépondérante dans les domaines de recherche actuels. La tendance actuelle est celle des architectures orientées services. En effet, cette nouvelle tendance augmente les besoins des entreprises en termes de flexibilité et d'adaptabilité. Ce besoin est ressenti autant au niveau conceptuel qu'au niveau opérationnel. An niveau conceptuel, le besoin de flexibilité est reflété par le développement de langages pour la composition de Web services tels que : BPEL4WS[2] et BPEL [3]. Au niveau opérationnel, de nouveaux middleware, tels que : BizTalk, IBM Websphere Choreographer, servent de plate-formes pour l'exécution des services (flux)[Rinderle 2004].

L'orchestration et la chorégraphie des flux de services a revitalisé l'idée de la gestion des processus métier et de workflow. La principale caractéristique des systèmes de gestion des processus métier [4] est la séparation de la logique des processus de leurs codes. En d'autres termes, la logique métier derrière le processus (données de flux et de controle entre tâches) est explicitement modélisée est non dissimulée dans le code (voir figure 1). Pour cela, des langages graphiques tels que : Réseaux de Petri, Diagramme d'activité UML et les réseaux d'activités sont utilisés.

Un système de gestion de processus métier consiste à gérer ces processus métier : (1) sur un plan global en cherchant à prendre en compte les processus de bout en bout, depuis la chaîne d'approvisionnement jusqu'aux activités internes et externes d'une entreprise, (2) sur le plan du cycle de gestion en s'intéressant aux différentes étapes du cycle de vie des processus depuis la modélisation jusqu'à l'exécution et le diagnostic.

Systèmes de gestion de processus flexibles

En dépit de ce potentiel établi, les systèmes de gestion de processus métier montrent certaines limites. Les entreprises expriment un véritable besoin de systèmes de gestion de processus évolués qui s'adaptent à la capitalisation de leurs systèmes d'information, à l'optimisation de leurs méthodes de production et à l'évolution de leurs besoins ainsi qu'à leurs méthodes de travail. Elles ressentent également un besoin de mécanismes de contrôle et d'outils de modélisation pour leurs systèmes de gestion de processus afin de concevoir des modèles de processus

[2]Business Process Execution Language For Web Services
[3]Business Process Execution Language
[4]Nous évitons pour le moment d'utiliser le terme "workflow" à cause de ses différentes interprétations. Le terme gestion des processus métier est plus général.

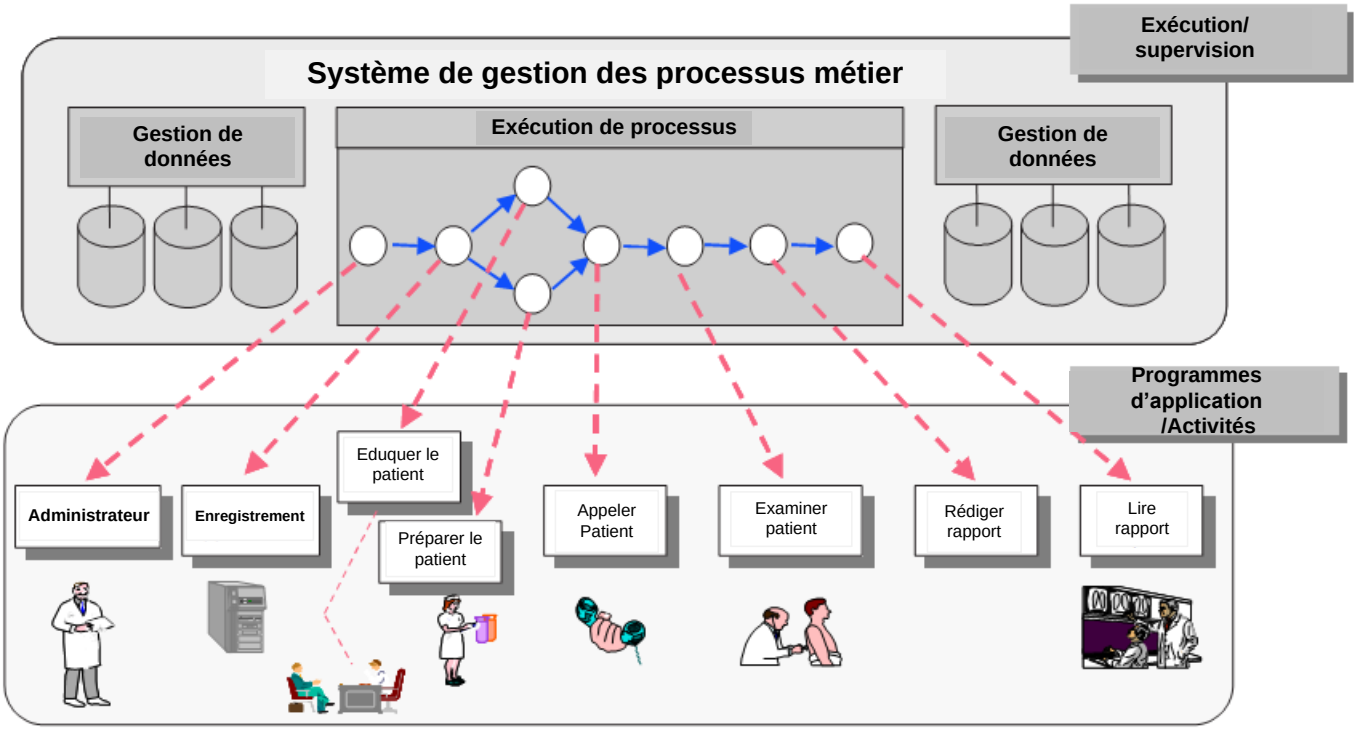


Figure 1: Séparation de la logique du processus du code dans un système de gestion de processus [Rinderle 2004].

robustes assurant un traitement fiable et flexible des erreurs et des exceptions d'exécution.

Généralement, les changements dans les systèmes de gestion de processus métier concernent, entres autres, le niveau d'abstraction. Il s'agit du niveau de l'application du changement dans un processus métier. Le changement peut concerner soit la spécification (Type) ou l'instance du processus (Instance). Les changements concernant la spécification sont nécessaires pour les processus à exécution longue (Ex : Changement des clauses d'un contrat). Par opposition, les changements concernant l'instance de processus peuvent concerner une instance en particulier (Ex : insertion, suppression,...). Ils surviennent de manière ad-hoc pour supporter le traitement des exceptions lors de l'exécution.

Par ailleurs, une évolution continue des processus est le témoin, de nos jours, d'une très grande diversification des services et des produits des entreprises. De nouveaux besoins émergent et les processus existants changent ("la seule constante est le changement"). Par conséquent, l'alignement des processus sur les évolutions observées exige une attention et une action continues.

Pour "maintenir cet alignement" il est important de détecter les changements dans le temps,

c.à.d, les déviations du comportement décrit ou prescrit. Des techniques de "découverte de processus" utilisant des traces d'exécutions pour découvrir le processus "réel" sont des outils appropriés pour détecter ces déviations et les utiliser pour supporter l'évolution du processus.

2 Problématique

Bien que les systèmes de gestion de processus métier actuels soient établis par des modèles de processus explicites, une des difficultés est née du fait que les processus ne sont pas tous explicitement définis, du fait également de la diversité des tâches et des intervenants (concepteurs, utilisateurs, gestionnaires, etc.) dans une application de conception/développement complexe ainsi que du caractère initialement imprévisible d'autres paramètres qui se manifestent après l'établissement du processus (besoins des utilisateurs, échec ou exception d'exécution inattendues, etc.). En effet, dans le cadre d'une application complexe, où il faut parfois coordonner des fragments de processus issus de modèles de processus différents, d'environnements hétérogènes, et des intervenants issus d'organisations différentes, il est impossible de prévoir et de rendre compte initialement et facilement de tous les paramètres nécessaires à une conception "parfaite".

Par ailleurs, quand un processus métier est représenté par un modèle, il n'est que le médiateur d'une réalité métier. En effet, ce modèle représente au mieux la réalité à un instant donné du monde réel du processus. Par conséquent, une flexibilité permettant l'évolution naturelle du métier du processus est manquante dans ces systèmes. Dans ce cas les interventions manuelles (humaines) deviennent plus fréquentes pour manipuler les entrées et sorties des workflow pour les adapter aux changements survenus dans l'environnement de travail. Ces interventions humaines ont un impact sur la productivité et le temps d'exécution des processus. Les processus non structurés sont les plus concernés par ce type de problème (Système de santé et de banque) car dus à l'incapacité de représenter leur réalité et à prévoir toutes leurs exécutions.

Dans ce travail nous avons identifié des problématiques clés. Les frontières entres ces problématiques ne sont pas distinctes. Elles sont relatives à la rigidité des systèmes de gestion de processus métier et leurs conséquences sur la difficulté d'assurer le dynamisme et l'évolution.

Les problématiques auxquelles nous cherchons à trouver des solutions sont :

Comment représenter les processus métier pour supporter les processus ad-hoc et les changements inattendus ?

En principe un système de gestion de processus métier doit utiliser une représentation qui reflète le processus métier. Les tâches humaines sont complexes et gouvernées par des règles plus qu'elles ne sont automatisables. Pour cela, les techniques appliquées pour modéliser le métier dans des processus doivent éviter les suppositions sur les régularités dans les procedures de travaux [Rinderle 2004]. Par conséquent une nouvelle approche inspirée de la théorie humaine est requise. Pour cela, il est nécessaire d'avoir une vision différente du processus métier et ses activités.

Comment exploiter l'historique et les traces d'exécution des processus métier pour améliorer la flexibilité ?

Les approches classiques de conception de processus prennent généralement comme point de départ les perceptions et suppositions des concepteurs et souffrent d'une implication déficiente des utilisateurs tant dans la conception que dans la vérification. En plus, elles manquent de mécanismes de correction permettant d'optimiser la structure du workflow initialement conçu du fait qu'elle analyse un modèle figé ne traitant pas le besoin d'évolution pendant la phase d'exécution.

Ainsi le défi de relever le besoin d'évolution et d'adaptation des processus métier suite à des nouvelles contraintes d'exécution perçues dans le contexte d'exécution, doit prendre comme point de départ les connaissances acquises suite à la phase d'exécution(Contexte d'exécution).

Une solution possible pour remédier à ces problèmes consiste à utiliser conjointement des techniques de "découverte de processus[5]" pour le contexte d'exécution des processus métier et le concept d'artifact pour la représentation de processus. Dans la section suivante, nous illustrons ce couplage par une présentation générale de notre approche.

3 Objectifs et contributions de la thèse

Un premier objectif de recherche de cette thèse est de fournir une représentation du processus métier **Adaptable**, **réutilisable** est **compréhensible** pour améliorer la flexibilité. Ce processus doit être **réactif** intégrant les nouvelles contraintes d'exécution et ses besoins d'évolution. Ceci en étant sensible à son environnement d'exécution.

[5]Nous utiliserons dans la suite de ce manuscrit le terme anglophone Process mining

Pour répondre au premier objectif notre point de départ était de constater qu'un processus métier est un ensemble d'activités ayant chacune un objectif à son niveau et que la coordination de l'ensemble faisait l'objectif global. Par conséquent, nous considérons un processus métier comme un ensemble de fragments supervisés pour atteindre un objectif.

Pour répondre au deuxième objectif de mécanismes d'exécution **réactifs**, nous avons d'abord relevé que les approches existantes de prise en charge du contexte prennent généralement comme point de départ les perceptions et suppositions des concepteurs et souffrent d'une implication déficiente des utilisateurs pour considérer le contexte à la phase de modélisation.

Concrètement, nous proposons l'utilisation des techniques de Process mining en spécifiant les méthodes qui permettent de découvrir le contexte en termes de règles métier à partir des traces d'exécutions en appliquant des techniques de fouille de données.

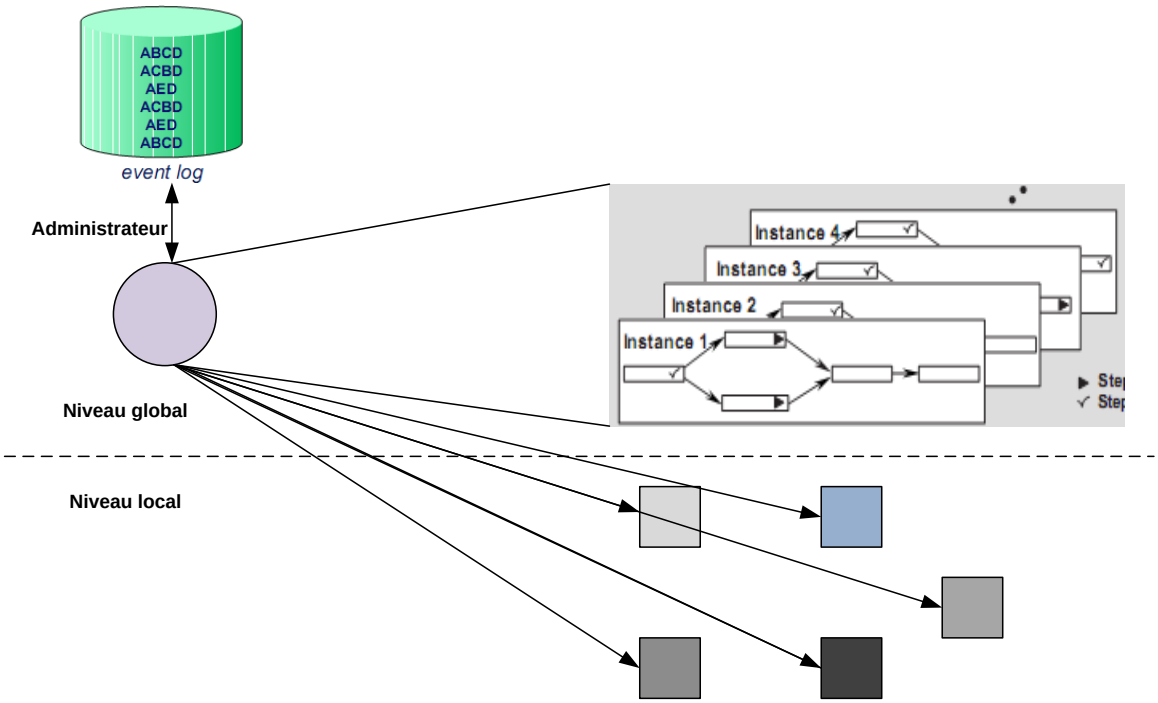


Figure 2: Vue globale de l'approche proposée.

Le but de notre travail se situe donc à deux niveaux (voir figure 2) :

- Niveau local : La representation de notre processus selon un nouveau concept (Niveau local de la figure 2). Il s'agit de concevoir un processus métier en termes de fragment représentant chacun une activité ayant une vue locale sur l'objectif à atteindre.
- Niveau global : Les décisions sont prises à ce niveau en se référant au contexte d'exécution utilisant les techniques du process mining (Niveau global de la figure 2). Ces décisions

sont prises par un administrateur.

Nous nous intéressons plus particulièrement dans cette thèse aux aspects suivants :

Support de la flexibilité

Il existe plusieurs types de flexibilité dans les processus métier. Nous nous intéressons dans nos travaux à la flexibilité par specification partielle. Elle concerne la possibilité d'exécuter un processus partiellement spécifié c-à-d. la possibilité de définir certaines structures de processus dans la phase d'exécution en permettant la sélection tardive d'un fragment de processus parmi plusieurs alternatives ou la modélisation tardive pour spécifier des fragments à la volée durant l'exécution du processus. La sélection se fait dans notre approche selon le contexte d'exécution. Nous considérons le processus tel un ensemble d'activités chacune encapsulée par une entité inspirée du domaine de la coordination dans les systèmes SMA appelés artifact.

Compréhensibilité

Le facteur majeur limitant les solutions workflow est la complexité des représentations des processus métier. Plus un modèle est complexe plus il est difficile de le modifier et de le comprendre. Le fait de diviser le processus métier en fragments indépendants aide à la compréhensibilité et diminue la complexité.

Localisation du changement

La localisation du changement est liée très étroitement à la flexibilité. Cela permet de limiter les changements à un nombre réduit de composants constituant le processus métier pour qu'un workflow soit adaptable. Par ailleurs, le changement survenu à un composant doit avoir le moins d'impact possible sur les autres processus.

La définition du processus métier en un ensemble de sous-processus et composants est une solution possible. Le changement d'un fragment sans changer la totalité du processus et sans avoir d'impacts sur les autres processus sont des avantages de la représentation des processus métier en entités. Le fait d'avoir une approche modulaire permet, également, que chaque portion peut être vérifiée indépendamment du reste du processus.

Réutilisabilité

La réutilisabilité des portions (Fragments) composant un processus métier est l'un des avantages majeurs. En effet, la réutilisation impose que la composition se fasse (en grande partie) en terme de parties existantes (exécuter des processus en réutilisant) et inversement que les parties existantes soient suffisamment flexibles pour répondre aux besoins d'un grand nombre de décompositions. La modularité permet de construire de nouveaux processus sur la base de ceux existants selon les conditions d'exécution (contexte d'exécution).

4 Organisation de la thèse

Le manuscrit est structuré en quatre chapitres et une conclusion générale.

Le chapitre I est consacré à la présentation du contexte de notre travail : l'ingénierie des processus métiers. Nous montrons en particulier le cycle de vie des processus et les avancées technologiques existantes pour relever les défis de cohérence et d'évolution de modèles. Nous présentons ensuite les caractéristiques des processus métier flexibles et leurs systèmes de gestion. Nous décrivons dans la section qui suit la nécessité de fermer la boucle du cycle de vie des processus métier par les techniques d'analyse, de diagnostic et de découverte. Nous concluons ce chapitre par une présentation de l'informatique sensible au contexte afin d'en extraire les étapes qui nous inspirent dans nos travaux.

Notre contribution démarre au chapitre II. Nous y proposons une « **solution sensible au contexte pour la flexibilité des processus métier**». Cette approche permet de pallier les limites d'intégration du contexte dans les systèmes de gestion de processus métier. Nous présentons les concepts de base et les vues générales de notre système.

Dans le chapitre III nous détaillons respectivement les motivations de l'utilisation du concept artifact, la structure de l'artifact ainsi que son utilisation pour l'exécution du processus métier. Nous décrivons également la spécification de l'architecture en termes de structures de modules, de comportements de ces modules, ainsi que la communication entre ces modules.

Le chapitre IV illustre l'application de notre approche à une étude de cas. Nous décrivons tout d'abord les différentes étapes et les concepts utilisés dans notre travail ainsi que l'architecture proposée. Nous précisons ensuite la plate-forme d'implémentation et comment les concepts de base proposés par notre approche (spécification d'artifacts, de l'administrateur et leurs interactions, etc.) peuvent être réalisés via cette plate-forme.

Enfin, le manuscrit se termine par une conclusion générale qui récapitule les travaux réalisés et propose quelques visions pour les travaux futurs.

Chapitre I

État de l'art

1 Introduction

Ce chapitre est consacré à un état de l'art sur les problèmes que pose la gestion des processus métier en général et ceux que pose la gestion des processus métier flexibles en particulier. Nous les présentons en deux parties. Dans une première partie nous présentons les systèmes de gestion des processus métier et les processus métier flexibles. En effet, nous entamons cette partie par donner la définition de la notion de processus métier pour ensuite aborder la sensibilité des systèmes d'information à cette notion de processus métier connue sous le nom de "process-aware information systems".

Dans la seconde partie, nous présentons un panorama de concepts utilisés dans notre solution du problème et nous discutons leurs apports sur la flexibilité. En effet, dans ce travail nous exploitons les techniques d'analyse de découverte et de diagnostic dont nous exposons le principe en premier lieu, pour ensuite présenter le deuxième concept utilisé à savoir les systèmes sensibles au contexte (context-aware systems).

Nous concluons ce chapitre par les limites que les travaux existants présentent pour l'amélioration de la flexibilité.

2 Notion de processus métier

Comme déjà constaté, l'utilisation croissante des systèmes de gestion de processus métier dans les entreprises exprime leur importance indéniable comme outil d'automatisation de leurs processus. Les systèmes de gestion de processus métier sont parmi les systèmes les plus élaborés pour définir et exécuter des processus. Ils permettent, en particulier, de décrire explicitement les méthodes de travail réalisants un processus(BPR[1]), de les expérimenter, de mesurer leurs

[1]En anglais BPR pour Business Process Re-Engineering

qualités et de les optimiser afin de pouvoir assurer l'amélioration et la réutilisation des processus(BPA[2]).

2.1 Concept de processus métier

Le terme de « processus métier » est souvent utilisé pour désigner des notions différentes : processus abstrait, processus exécutable ou processus collaboratif [Giaccari 2002].

Un processus métier est une chorégraphie d'activités incluant une interaction entre participants sous la forme d'échange d'informations [Tanguy 2003]. Les participants peuvent être :

- Des applications / services du système d'information,
- Des acteurs humains,
- D'autres processus métier.

Un processus métier peut être interne à une entreprise ou mettre en jeu des entreprises partenaires. On parle alors de processus collaboratif.

Processus collaboratif

Un processus collaboratif est un processus métier mettant en jeu des entreprises partenaires. Un processus collaboratif incluant n partenaires est composé de deux parties : une interface et n implémentations. L'interface définit la partie visible du processus, c'est-à-dire le contrat entre les partenaires : définition des documents métiers échangés, du séquencement des activités, des rôles et responsabilités de chaque partenaire. L'exécution spécifique de chaque partenaire est abstraite grâce à cette interface. Les implémentations (une pour chaque partenaire) définissent le comportement interne de chaque partenaire pour réaliser le processus et respecter les contraintes définies dans l'interface publique.

Processus exécutable

Un processus peut être rendu exécutable de trois manières [Tanguy 2003] :

- Il peut orchestrer les applications / services du SI ainsi que des actions utilisateurs pour rendre une tâche automatisée. Par exemple, un processus de gestion de bons de commande va recevoir les bons de commande via des messages XML, les transmettre aux personnes adéquates, se renseigner sur la disponibilité des éléments commandés dans les bases de données de l'entreprise, etc.

[2]Business Process Analysis

– Rendre un processus exécutable ne signifie pas nécessairement l'automatiser. Par exemple un processus peut uniquement être l'automatisation de la transmission d'informations entre acteurs (approche Workflow), les actions étant effectuées manuellement par les utilisateurs.
– Rendre un processus exécutable peut aussi signifier introduire des points de contrôle pour permettre le contrôle de son déroulement. On parle alors de processus de BAM[3].

2.2 Processus métier VS Workflow

Plusieurs définitions ont été apportées dans la littérature au concept de Workflow.

Définitions

Un Workflow (terme anglais) est un flux d'informations au sein d'une organisation[Leymann 2000]. Le terme officiellement recommandé par la commission générale de terminologie et de néologie est « flux des travaux », ou « automatisation des processus ». Nous optons pour la définition suivante : Le Workflow peut être défini comme « l'automatisation de processus métier par échange de documents, informations et tâches entre acteurs pour action ». Le workflow a pour objectif la coordination automatisée de tâches réalisées par des intervenants humains. Le moteur de workflow transfère des documents entre les participants d'un processus en leur assignant des tâches (valider le document, effectuer une modification, etc.).

Différence entre workflow et processus métier

D'après toutes les définitions qui existent, nous pouvons retenir que le workflow est l'automatisation des documents et des tâches. Ceci dit, les documents et les tâches ne sont pas suffisants pour l'automatisation des processus métier. Il est nécessaire d'avoir un niveau d'abstraction supplémentaire. Il est également important de prendre en considération tous les acteurs participants à un processus, un humain ou une application.

Différents aspects de processus métier

Dans une entreprise un processus métier regroupe plusieurs aspects de base qui peuvent être modélisés et traités indépendamment(Figure I.1)[Russell 2007]. Ces aspects ont inspiré la taxonomie de la flexibilité des processus métier que nous décrirons dans les sections suivantes :

[3]Terme anglais pour Business Activity Monitoring

- Aspect Organisationnel : Il décrit la structure organisationnelle de l'entreprise (départements, services, etc.) qui est invoquée pour la réalisation du processus métier à travers les concepts de rôle, de groupe de rôles, d'acteur (ou d'agent), d'équipe, de compétences de chaque membre de l'équipe, de position dans l'équipe ou de responsabilité dans la structure.
- Aspect Informationnel : Il décrit les informations (données, documents, etc.) qui sont manipulées par l'utilisateur (ou acteur) du processus métier pour exécuter l'activité ou le processus.
- Aspect Comportemental : Il correspond à la modélisation de la dynamique du processus métier, c'est à dire, la manière dont les activités sont chronologiquement exécutées et les conditions les déclenchant.
- Aspect Opérationnel : Il décrit l'ensemble des activités participant à un processus métier.
- Aspect Fonctionnel : Il décrit l'objectif à atteindre par le processus métier.

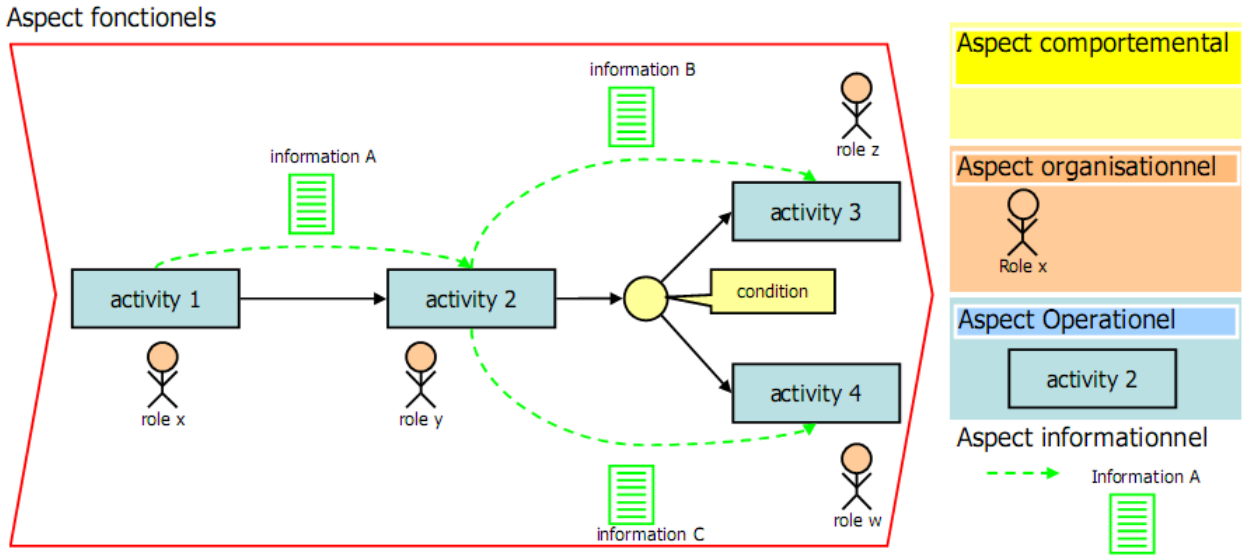


Figure I.1: Différents aspects d'un processus métier.

2.3 Gestion des processus métier

Le BPM [4] peut être défini comme « l'ingénierie des processus métier des organisations à l'aide des technologies de l'information » [Gaaloul 2006]. Le terme «la gestion des processus métier (BPM)» est un terme populaire aussi bien dans le domaine métier que scientifique qui

[4]Business Process Management

soulève des questions concernant, notamment la conception, l'analyse, la modélisation, l'exécution et le contrôle des processus métier [Gaaloul 2006]. Nous n'essayons pas d'étendre ou de raffiner cette définition d'autant que d'autres définitions ont été proposées, mais nous intéresser aux ingrédients remarquables de la gestion du processus métier relatifs, en particulier, aux méthodes, techniques et outils supportant la conception, l'exécution et l'analyse des processus métier. Nous adoptons la même approche processus métier que Leymann et autres [Leymann 2000] qui distinguent deux aspects fondamentaux, à savoir l'aspect temps de construction et l'aspect exécution (voir figure I.2.

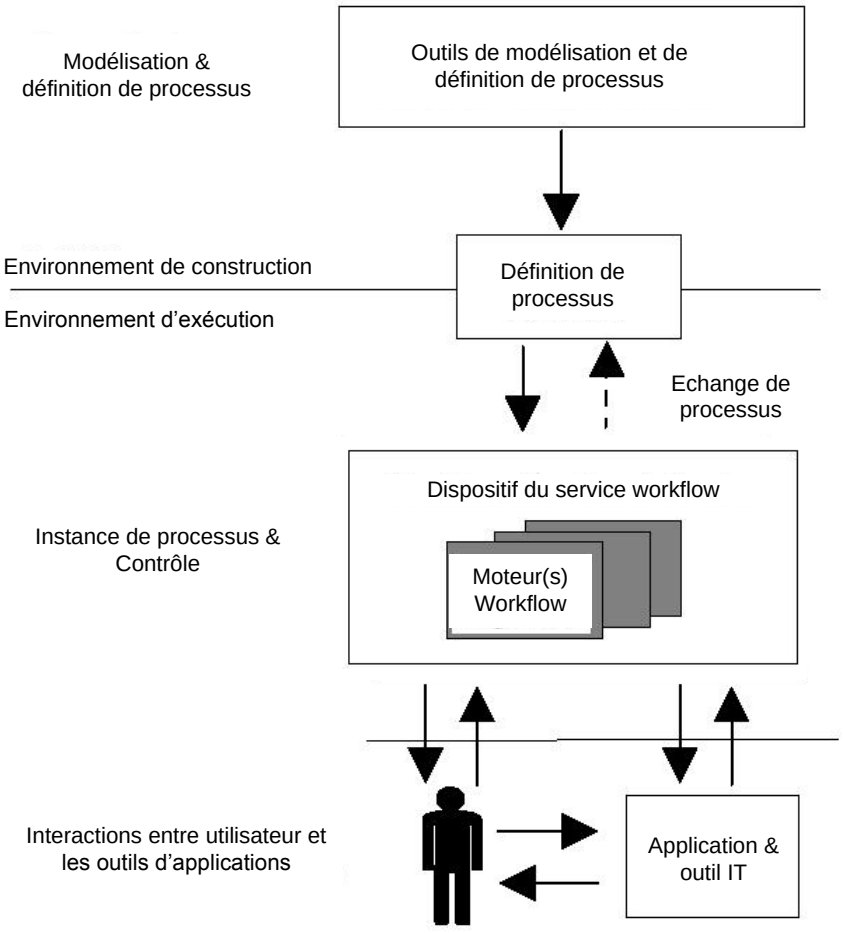


Figure I.2: Caractéristiques du BPM.

L'aspect temps de construction est centré sur la conception du processus métier ; l'aspect exécution est centré sur le contrôle de l'exécution et le traitement des échecs d'exécution.

Ils existent également des approches qui soulignent l'aspect centré client d'un processus

métier qui est «une collection d'activités qui prend un ou plusieurs types d'entrées et fournit un résultat qui est de valeur pour l'utilisateur» .

L'aspect centré client d'un processus se concentre sur le rôle central des utilisateurs dans la validation de l'efficacité du processus. En utilisant cette distinction nous considérons BPM comme le domaine de conception et d'exécution des processus métier en vue de satisfaire les besoins métiers de l'utilisateur. La distinction de ces trois concepts est également devenue commune au domaine des systèmes de gestion de workfows pour discuter leurs fonctionnalités principales.

Évolution des systèmes d'information d'entreprises

Pour montrer l'importance des systèmes de gestion de processus métier, il est intéressant de les mettre dans une perspective historique. La figure I.3 montre que les systèmes d'information se composent d'un certain nombre de couches [Gaaloul 2006]. Le centre est constitué par le système d'exploitation. La deuxième couche se compose des applications génériques utilisées par un grand nombre d'entreprises. D'ailleurs, ces applications sont typiquement utilisées dans des départements multiples au sein de la même entreprise (par exemple, système de gestion de base de données, éditeur de texte, tableurs, etc.). La troisième couche se compose des applications spécifiques au domaine. Ces applications sont seulement utilisées dans des types spécifiques d'entreprises et de départements. La quatrième couche se compose des applications sur mesure. Ces applications sont développées pour des organisations spécifiques. Les tendances représentées

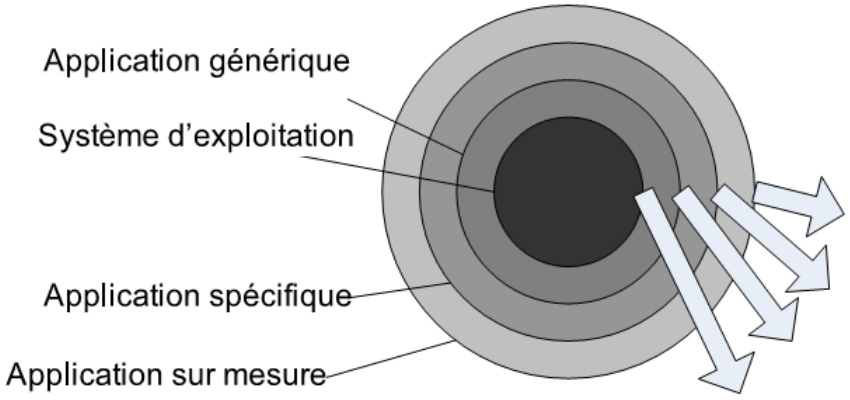


Figure I.3: Système d'information pour les entreprises.

dans la figure I.3 fournissent un contexte historique pour les systèmes de gestion de processus métier. En effet, les systèmes de gestion de processus métier sont des applications séparées résidants dans la deuxième couche. Les exemples les plus remarquables des systèmes de gestion

de processus métier résidants dans la deuxième couche sont les systèmes de gestion de workflows comme Staffware, MQSeries, COSA et FLOWer [der Aalst 2005] [Gaaloul 2006].

Notons que les principaux systèmes de planification de ressources d'entreprises peuplant la troisième couche offrent également un module de gestion de workflows. Les moteurs de workflows de SAP, Baan, PeopleSoft, Oracle, et JD Edwards peuvent être considérés comme des systèmes intégrés de gestion de processus métier.

Dans les années 60, la deuxième et la troisième couche se sont développées et la tendance courante est que les quatre cercles augmentent de taille, c.à.d, elles se déplacent vers l'extérieur en absorbant de nouvelles fonctionnalités métiers.

Dans les années 90, un regain d'intérêt énorme pour ces systèmes s'est manifesté.

D'un point de vue technique, l'arrivée des BPMS marque une nouvelle évolution dans la structuration des applications du système d'information de l'entreprise. Les architectures deux tiers (client-serveur), trois tiers puis aujourd'hui n-tiers de type web services ont permis de séparer de plus en plus clairement la couche présentation (client web léger), de la couche application (règles et traitements) et de la couche données (SGBD, ERP, lecacy applications, EAI ...). Pour autant sans BPMS les processus, et ce qu'on appelle la logique métier (business logic), sont encore codés dans la logique applicative. Il en est d'ailleurs de même des règles de gestion (business rules). Au coeur des applications critiques d'entreprise, les BPMS constituent une nouvelle couche du système d'information et visent à en extraire les processus (et même les décisions ou règles métier) selon les mêmes raisons qui ont motivé dans les années 90 les informaticiens à extraire des applications les données dans les SGBD. L'extraction des processus métier des applications du système d'information, notamment des ERP, constitue un niveau d'abstraction inédit qui doit permettre, selon les défenseurs du BPM, d'orchestrer les services d'un processus interne propre à une entreprise et de chorégraphier l'enchainement des opérations à réaliser dans une collaboration inter-entreprises selon les évolutions du métier et du marché, que ces services se présentent sous la forme de web services ou non.

D'un point de vue stratégique, la mise en évidence d'une nouvelle couche de services de gestion des processus métier au coeur du système d'information de l'entreprise vise à résoudre le problème de la rigidité endémique des systèmes d'information informatisés [Tanguy 2003].

Le nombre de systèmes de gestion de workflows développés dans la dernière décennie et les nombreux articles sur la technologie de workflows illustrent la renaissance des systèmes d'information dédiés aux processus métier. Cependant, il y avait également des problèmes plus fondamentaux comme l'absence d'une méthode unifiée de modélisation de processus. La plupart des problèmes techniques ont été résolus à ce jour. Cependant, des problèmes plus conceptuels demeurent non résolus.

Un des problèmes est que ces systèmes exigent une conception de workflow, c.à.d, qu'un concepteur doit construire un modèle détaillé décrivant exactement le déroulement du travail. La conception des workflows est loin d'être insignifiante : elle exige une connaissance profonde du processus métier réel et des paramètres d'évolution au cours du temps. Les besoins d'évolution et d'adaptation aux nouveaux contextes peuvent se manifester particulièrement pendant l'exécution et influencer l'efficacité et la fiabilité du processus métier.

2.4 Cycle de vie d'un processus métier

Comme nous l'avons indiqué dans la section précédente, la gestion des processus métier inclut des méthodes, des techniques, et des outils pour supporter la conception, l'exécution, la gestion et l'analyse des processus métiers opérationnels. Elle peut être considérée comme une extension des systèmes et des approches classiques de gestion de workflows (WfM).

Le schéma de la figure I.4 montre la relation entre WFM et BPM en utilisant le cycle de vie de BPM [Gaaloul 2006]. Le cycle de vie de BPM décrit les différentes phases de support des processus métier. Dans la phase de conception, les processus sont (re)conçus. Dans la phase de

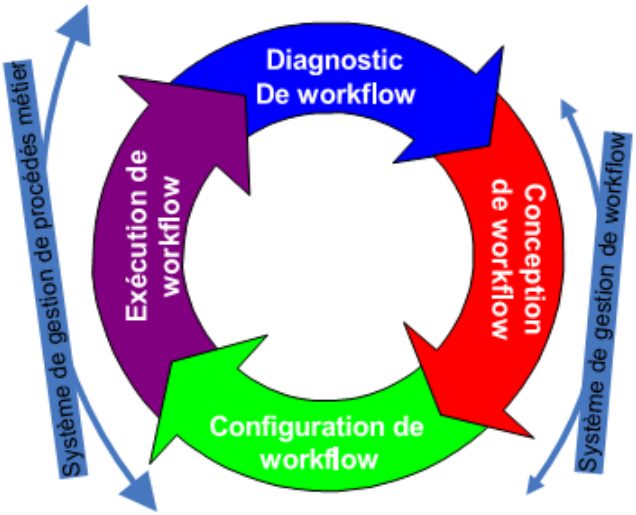


Figure I.4: Cycle de vie de workflow et cycle de vie BPM.

configuration, les conceptions sont mises en application en configurant le système d'information (par exemple, un système de gestion de workflows) pour implanter le processus métier. Après la configuration, la phase d'exécution commence là où les processus métier sont exécutés en utilisant le système configuré.

Lien modélisation / exécution

On considère que la finalité du BPM consiste à fournir à la direction d'une entreprise la maîtrise toujours plus grande dans l'exercice de son métier. Pour ce faire, les processus métier doivent être identifiés et conçus. La conception des processus consiste à formaliser leur description. La sortie de la conception de processus se matérialise par une définition textuelle de processus, renseignée de manière exhaustive, et prête à être déployée pour son exécution. Notation (BPMN 1.0), proposé par l'organisation BPMI.org. (http ://www.bpmi.org : BPMN 1.0 working draft).

Mais le véritable défi se situe dans la capacité des processus dont la description a été formalisée par les utilisateurs métier de déboucher directement sur la génération de langages exécutables par des programmes informatiques et des web services. Le langage BPEL4WS (Business Process Execution Language for Web Services, qui remplace BPML Business Process Modeling Langage devenu ensuite BPEL Business Process Execution Langage) est une spécification de standards pour l'orchestration des web services. Il supplante les initiatives plus généralistes (telle que BPML de BPMI) qui n'ont pas abouti car trop éloignées d'un modèle d'exécution. La plate-forme d'exécution d'un BPMS propose généralement un moteur d'exécution (runtime) qui exécute du code généré où, plus proprement, interprète un modèle exécutable. Le lien modélisation/exécution est ce qui fait l'attrait des BPMS actuels. Le principal intérêt est de pouvoir, en théorie du moins, orchestrer les services applicatifs selon le contexte d'utilisation et les évolutions du métier. Ceux-ci doivent pouvoir être rapidement et facilement modifiés, si possible par les utilisateurs métier, sans avoir à repasser par la cascade de modifications généralement imposées par le cycle de développement du logiciel des informaticiens. Pour autant, ce lien repose sur des partis pris très forts. C'est notamment à ce niveau que se situe le débat sur les web services et leur lien avec le BPM. Selon de nombreux analystes, la gestion de processus ne pourra véritablement être déployée sans exploiter les avantages offerts par la technologie des web services et plus généralement les architectures orientées services (Services Oriented Architectures ou SOA). Les web services s'imposent graduellement aujourd'hui comme les connecteurs standards applicatifs, fournis de plus en plus par les éditeurs de progiciels eux-mêmes, en remplacement des outils d'EAI permettant d'accéder aux applications existantes (legacy applications) et aux progiciels du SI.

Lien exécution/supervision

C'est le passage de témoin entre l'exécution des processus et l'intégration du système d'information, qui seul autorise un lien exécution/supervision.

Les logiciels de workflow, comme les outils d'EAI ou les systèmes B2B, n'ont pas réussi à couvrir la gestion des processus car ils ne sont jamais parvenus à intégrer le système d'information au niveau fonctionnel et surtout dans son ensemble, se cantonnant le plus souvent à intégrer les logiciels du système d'information au niveau technique et sur un périmètre très parcellaire. Toute évolution du métier de même que toute modification mineure du système d'information nécessitait de reprendre le cycle complet de développement. Outre l'impact technique négatif de la non intégration de la gestion des processus au système d'information, celle-ci affecte également les capacités de supervision (monitoring) des processus. L'exécution contrôlée des processus par le moteur de processus génère une source considérable de mesures concernant les délais (durées), les charges (personnes) ou les coûts, offrant ainsi aux gens du métier la possibilité de corriger en ligne les dysfonctionnements ou encore d'effectuer en quasi-temps réel les adaptations nécessaires, selon les objectifs stratégiques de l'entreprise. La mise en relation entre la stratégie de l'entreprise et son exécution opérationnelle conduit à une nouvelle discipline au croisement du BPM et de la business intelligence, que certains analystes appellent la Business Activity Monitoring (BAM). Cette nouvelle technique modifie considérablement les approches actuelles d'aide à la décision en ce qu'elle travaille directement sur des données réelles du système d'information

Dans la phase de diagnostic, les processus sont analysés pour calculer les différentes mesures de performance évaluant le processus exécuté. Comme le schéma de la figure I.4 le montre, l'intérêt traditionnel d'un système de gestion de workflow est centré sur la moitié inférieure (c.à.d les phases de conception, de configuration et d'exécution) du cycle de vie de BPM [van der Aalst 2003a]. Par conséquent, il y a peu de support pour la phase de diagnostic. En plus, le support de la phase de conception se limite à fournir un éditeur, alors qu'une analyse et une vraie aide à la conception sont absentes. Il est remarquable que peu de systèmes de gestion de workflows soutiennent le support, la simulation, la vérification et la validation de la phase de conception de processus. Il est également remarquable que peu de systèmes soutiennent l'interprétation des données d'exécution en temps réel. Notons que la plupart de systèmes de gestion de workflows collectent les traces d'exécutions des instances exécutées de processus. Cependant, aucun outil pour supporter une conception se basant sur cette source importante d'information n'est offert par les systèmes de gestion de workflows actuels.

3 Process-Aware Information Systems

Au cours de ces deux dernières décennies, nous avons assisté à un changement des systèmes d'information orientés données «data-aware» vers des systèmes d'information orientés processus «process-aware» connus sous le nom de Process-Aware Information System (PAIS)[Rinderle 2004]. Dans le but de supporter les processus métier, un système d'information d'une entreprise doit être centré sur ses processus et leurs contextes organisationnels. L'un des ancêtres des PAIS sont les systèmes de gestion workflow [5]que nous avons présenté dans la section précédente.

La différence principale entre un WfMS et BPM est qu'un WfMS ne prend en charge qu'une partie du cycle de vie d'un processus métier. Tandis qu'un BPM prend en charge la phase d'analyse et de diagnostic. Les BPMs ont une portée plus large que celle des WfMSs. Ils ne se concentrent pas seulement sur l'automatisation comme le font les systèmes WfMSs. Ceci dit, les deux systèmes ont pour objectif l'exécution des processus métier opérationnels appelés «processus workflow » ou simplement « Workflow ». Le terme PAIS est générique et désigne les systèmes qui gèrent et exécutent de tels workflow.

3.1 Définition

Un PAIS peut être défini comme «un système logiciel qui gère et exécute des processus opérationnels impliquant des personnes, des applications et / ou des sources d'information sur la base des modèles de processus»[Russell 2007].

Notons que les modèles de processus mentionnés dans cette définition sont généralement exprimés en langage graphique tels que les réseaux de Petri. Les modèles sont généralement instanciés plusieurs fois (par exemple, pour chaque commande de client) et chaque instance est traitée d'une manière prédéfinie (éventuellement avec des variations).

D'après la définition ci-dessus, un éditeur de texte n'est pas sensible à ces processus «process aware» dans la mesure où il est utilisé pour faciliter l'exécution des tâches spécifiques sans aucune connaissance du processus métier auquel ces tâches appartiennent. Les éditeurs de texte sont des applications qui supportent les tâches et non la gestion des processus. Plusieurs applications sont, dans ce contexte, des systèmes d'information [Russell 2007].

3.2 Apport à la gestion des processus métier

Les exemples classiques des systèmes PAISs sont les systèmes BPMs et WfMSs. Ces systèmes exécutent des processus opérationnels guidés par des représentations explicites de ces mêmes

[5]WfMS : Workflow Management System

processus. Le fait d'avoir une connaissance globale de ces processus métier est une propriété importante pour les SIs. Le passage du principe des SIs dirigés vers les tâches aux processus présente un certain nombre d'avantages :

- L'utilisation de modèles pour représenter les processus offre un moyen de communication plus facile entre experts du métier.
- Les systèmes dirigés par les modèles s'adaptent mieux aux éventuels changements que les systèmes dirigés par le code.
- La représentation des processus pris en charge par une organisation permet leur exécution automatique. Cela peut conduire à une meilleure performance.
- La modélisation permet la (re)conception des processus comme résultat des opérations de contrôle et d'analyse.

Il existe, dans la littérature [Russell 2007] une classification des PAISs que nous résumons dans ce qui suit. Cette classification démontre également l'évolution des WfMSs et les BPMs. Les PAISs peuvent être classés en fonction des participants au processus métier. Ils peuvent être orientés personne ou orientés système (application).

3.2.1 Les systèmes Personne à Personne

Les systèmes PAIS Personne à Personne (P2P)[6] gèrent les processus où les acteurs principaux sont des êtres humains, c'est à dire les processus dont les tâches nécessitent une intervention humaine. Le suivi d'emploi, la gestion de projets ou encore les outils groupware sont des exemples de tels processus. Les processus de type personne à personne relient les employés d'une entreprise dans un but collaboratif. Généralement, ce genre de processus n'implique pas la participation d'application. De plus, les applications qui participent à ces processus (par exemple les serveurs de suivi de projet, les clients e-mail, les outils de vidéo-conférence, etc) sont essentiellement utilisées pour executer des tâches assistées pas ordinateurs.

3.2.2 Les systèmes Application à Application

Par opposition aux systèmes P2P, existent les systèmes application à application (A2A). Les processus métier gérés par ces systèmes ne font intervenir comme acteurs que des logiciels (applications). De tels processus sont utilisés dans le domaine de l'informatique distribuée et de l'integration des applications. Les systèmes de traitement des transactions, les plates-formes EAI et les serveurs d'intégration basés sur le Web sont des exemples de tels processus.

[6] Terme anglais pour Person To Person

3.2.3 Les systèmes Personne à Application

Les systèmes Personne à Application (P2A) [7] impliquent à la fois des tâches concernant des interactions entre personnes et les tâches qui concernent des applications qui agissent sans intervention humaine. Les systèmes workflow sont un excellent exemple de ce genre de système car leur objectif principal et l'automatisation et la gestion du flux de travail entre humains.

3.2.4 Prévisibilité des processus métier

Le degré de structure du processus à automatiser (qui est fortement lié à sa prévisibilité) est également utilisé comme une dimension (critère) de classement des PAISs. Les processus métier structurés sont plus faciles à gérer que les processus métier non structurés. De plus, il est également évident que les plus petits processus sont plus faciles à gérer que les plus grands. Cependant, nous aimerions détailler cet aspect de prévisibilité.

La figure I.5 montre la distinction entre processus non structurés[8], processus structurés de manière ad hoc[9], faiblement structurés[10] et fortement structurés[11].

Un processus est dit non structuré s'il n'y a pas de modèle explicite qui lui est associé. C'est le cas des processus métier collaboratifs supportés par les systèmes groupware qui n'offrent pas la possibilité de définir des modèles de processus.

Un processus est dit structuré de manière ad hoc si un modèle de processus est à priori défini mais exécuté une fois ou un nombre limité de fois avant d'être rejeté ou modifié. L'exemple d'un tel processus se trouve dans les environnements de gestion de projet où un modèle de processus (par exemple un graphique de projet) est souvent exécuté seulement une fois. C'est également le cas dans les environnements de grille de calculs où un scientifique peut définir un modèle de processus correspondant à un calcul, impliquant un certain nombre d'ensembles de données et des ressources informatiques et ensuite exécuter ce processus une seule fois.

Un processus faiblement structuré est un processus pour lequel il est un modèle défini a-priori accompagné d'un ensemble de contraintes. Ce modèle décrit l'exécution classique (voie normale) du processus tout en permettant de s'écarter de ce modèle en respectant certaines contraintes.

Enfin, un processus bien structuré est un processus qui suit toujours un modèle défini a-priori. C'est le cas des systèmes de gestion workflow.

[7] Person To Application
[8] unframed
[9] ad hoc framed
[10] loosely framed
[11] tightly framed

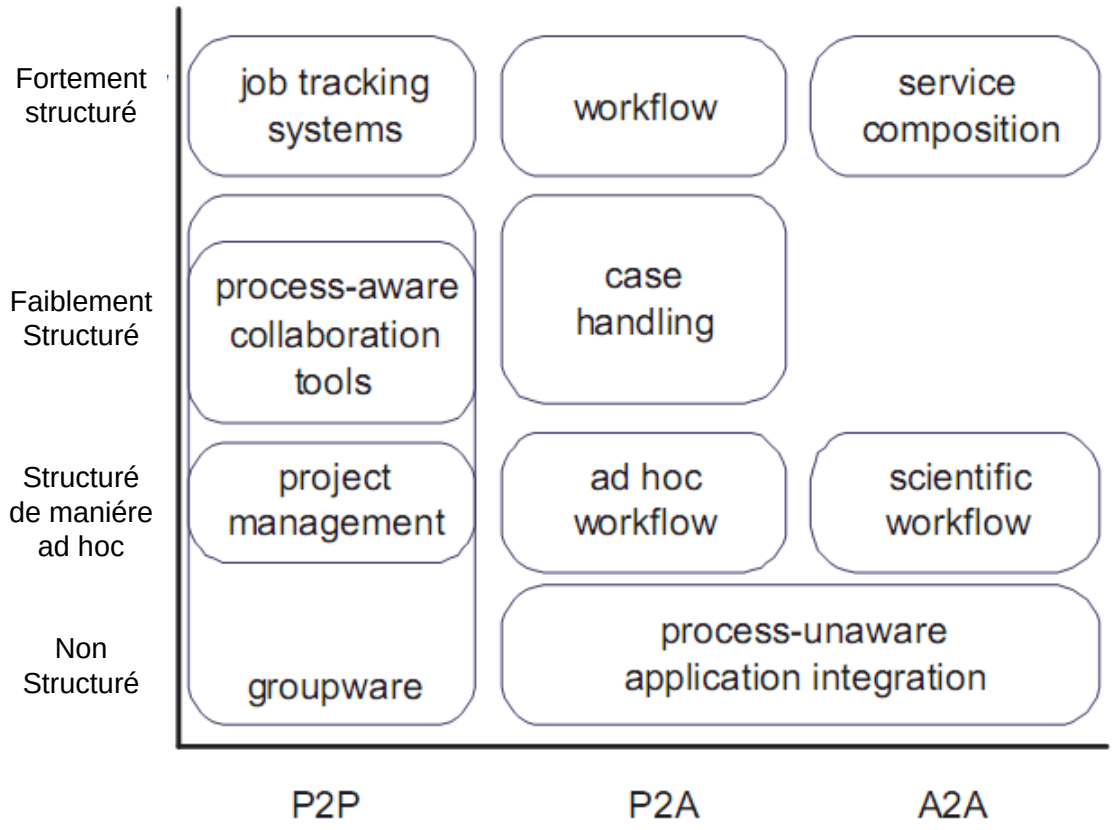


Figure I.5: Catégorisation des systèmes PAISs avec les outils développés associés[Russell 2007].

La figure I.5 illustre également les différents outils existants relatifs aux différents types de PAISs en fonction du degré de la structure des processus métiers et de la nature des acteurs du processus (Personne ou Application).

Comme avec les P2P, P2A, et A2A, les frontières entre processus non structurés, processus structurés de manière ad hoc, mal structurés, et bien structurés ne sont pas du tout bien claires. En particulier, entre les processus faiblement et fortement(bien) structurés où existe une continuité évidente. En effet, au cours de sa vie opérationnelle (exécution), un processus considéré comme un processus fortement structuré peut dévier du chemin défini par le modèle initial de manière imprévisible et devient faiblement structuré. Inversement, un processus faiblement structuré qui s'est exécuté un nombre important de fois peut mener à l'apparition d'une structure commune qui peut le rendre un processus bien structuré.

3.3 Discussion

Il ressort de ce qui précède que les systèmes Workflow, case handling et les systèmes d'information sont tout simplement des exemples de système PAISs. Au début, les systèmes PAISs ont été développés à des fins inter-entreprises. Cependant, ces dernières années il y a eu une avancée vers les processus qui franchissent les barrières organisationnelles (entreprises). Ces processus peuvent être :

- Un à un : c'est une relation bilatérale
- Un à plusieurs : Une relation qui relie une entreprise avec plusieurs d'autres.
- Plusieurs à plusieurs : Il s'agit de l'interaction de plusieurs partenaires.

Il ressort également l'importance de la modélisation dans de tels systèmes, d'une part et d'autre part, nous remarquons l'inadéquation entre les modèles de processus définissant leurs comportements et leurs structures et la réalité dans l'exécution. Dans cette optique le domaine du process mining [Günther 2007] apporte de nouvelles possibilités de solution que nous décrirons dans une section ultérieure. Notons également que l'actuelle génération des systèmes PAISs doit être flexible et doit s'adapter aux changements de l'environnement des entreprises surtout si celles ci sont ouvertes sur les autres (Plusieurs à plusieurs). Dans la section suivante nous étudierons les processus métier flexibles.

4 Spécificité et apport des processus métier flexibles

Les organisations sont constamment à la recherche d'améliorations de l'efficacité de leurs processus métier en termes de temps et de coût. En effet, elles doivent survivre dans un environnement dynamique en assurant trois dimensions (Voir Figure I.6)[Adams 2007a] :

- Coût d'exécution par processus (Axe Y).
- Coût de configuration par processus (Axe Z).
- Délai d'exécution de processus (Axe X).

Ainsi, ces entreprises doivent réduire au minimum ces trois dimensions. Elles doivent être efficaces en réduisant le coût opérationnel des processus. Elles doivent être flexibles (réduire la différence entre anciens processus et les processus modifiés). Elles doivent également être capables de faire évoluer leurs processus en un nombre réduit d'itérations avec le moins de paramètres possibles.

Dans ce contexte, les systèmes workflow actuels assurent la gestion des activités « standard » et répétitives qui ne changent pas entre deux instances de processus à l'exécution.

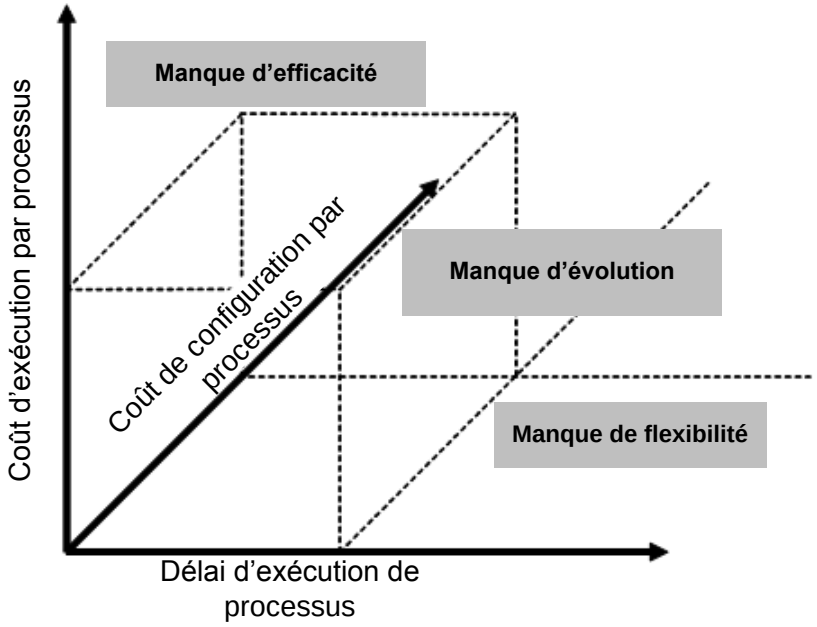


Figure I.6: Les trois dimensions d'évaluation de l'entreprise d'après l'espace de processus.

Dans certains environnements de travail et pour certains types d'activités, des représentations formelles du travail jouent un rôle fondamental en assurant l'ordonnancement des tâches et contribuer à l'assignation des rôles correctement (les systèmes médicaux et bancaires).

Cependant, même dans des environnements très structurés, il est difficile (voire impossible) de capturer correctement toutes les activités du travail dans un modèle, du moins sans produire des modèles très complexes. La notion de flexibilité émerge comme un domaine de recherche pivot dans le domaine des BPMs. La nécessité pour les systèmes workflow d'être flexibles est vite apparue évidente aux chercheurs. Le prochain point présente la définition de cette notion de flexibilité.

4.1 Définition de la flexibilité

Selon la définition du dictionnaire, est flexible ce qui est [Regev 2005] :

1. Capable d'être fléchi : Pliant.
2. Capable de céder à l'influence : Tractable.
3. Susceptible de s'adapter aux circonstances.

La flexibilité est une qualité essentielle reflétant l'efficacité des BPMs afin de s'adapter aux perpétuels changements que subit l'entreprise dans son environnement.

En effet, la flexibilité est souvent définie en terme de capacité de processus métier et des technologies qui les soutiennent en vue de s'adapter à ces changements.

[Regev 2006] définie la flexibilité comme la capacité de céder à un changement sans disparaître.

D'autres considèrent la flexibilité du point de vue opposé, c'est à dire, ils se concentrent sur les parties du processus qui restent inchangées, plutôt que de se concentrer sur celles qui doivent changer.

Processus métier flexibles

Un processus métier est considéré flexible s'il est possible de le modifier sans le remplacer dans sa totalité [Schonenberg 2008a].

La flexibilité d'un processus reflète son habilité à couvrir les changements en variant ou en adaptant les parties du processus métier qui sont affectées tout en retenant le format essentiel des parties non affectées par le changement [Schonenberg 2008b].

En conclusion, il est possible de définir la flexibilité par la capacité à supporter et à prendre en charge les changements métier. Un processus métier est flexible s'il est capable d'atteindre ses objectifs en dépit des variations et des stimulis. Lorsqu'une variation apparaît entre la définition et le déroulement actuel du processus, des moyens sont requis pour assurer que le processus évolue en accord avec ses attentes.

4.2 Gestion des processus métier flexibles

La nécessité pour les systèmes workflow d'être flexibles est vite apparue évidente aux chercheurs. Seules les approches permettant de l'obtenir ont varié [Kobayashi 2005] [Bose 2009][Weber 2009] [Reichert 2009][Weber 2008]. Nous analyserons dans une section ultérieure ces différents travaux. Dans cette section, nous allons exposer les critères fonctionnels que les BPMS devraient satisfaire afin d'atteindre la flexibilité. Chacun de ces critères est considéré comme un axe de recherche à part entière dans le domaine de la flexibilité des processus métier.

Flexibilité et réutilisation

Les systèmes workflow devraient supporter l'utilisation d'actions contextuelles. Ainsi, la réalisation d'une activité dans un éventuel environnement est aidé considérablement par l'utilisation d'un répertoire d'actions et d'opérations offrant un choix sur l'action à exécuter. On désigne ainsi un facteur crucial pour la représentation des workflows flexibles. À tout moment, il

peut y avoir plusieurs séquences possibles qui peuvent être suivies en utilisant un sous-ensemble d'actions disponibles pour atteindre l'objectif de l'activité[Adams 2007b].

Ainsi, l'idéal serait qu'un WfMS flexible gére un catalogue d'actions. En effet, lors de l'exécution une action pourrait être choisie automatiquement ou avec une intervention humaine minimale, selon l'information contextuelle qui se présente. La disponibilité d'un catalogue d'actions pourrait également favoriser la réutilisation. Les actions peuvent être mises à disposition pour une utilisation dans plusieurs activités distinctes et/ou utilisées comme modèles(template) pour la définition de nouvelles actions.

Les produits commerciaux étudiés fournissent des frameworks de modélisation qui sont essentiellement monolithiques, mais avec différents niveaux de soutien à la déconstruction de tâches. SAP R / 3 permet de définir des «blocs» qui peuvent être insérés dans d'autres «blocs», offrant ainsi un soutien pour l'encapsulation et la réutilisation. CSR utilise la notion de processus père où les données peuvent être passées à partir d'un processus à un sous-processus. Le système ADOME [Chiu 2000] fournit des templates qui peuvent être utilisés pour construire un modèle de workflow et fournit un certain appui aux changements dynamiques (manuellement). Un catalogue de «squelettes» de modèles qui peuvent être instanciés ou spécialisés lors de la conception est pris en charge par le système WERDE [Casati 2000].

Aucun produit ou prototype n'offre un répertoire extensible des processus à partir duquel une selection dynamique peut être effectuée à l'exécution.

Adaptation par la réflexion

Les systèmes de gestion des processus métier devraient pouvoir soutenir l'adaptation évolutive des processus en se basant sur l'expérience acquise au cours des exécutions passées. L'anticipation de l'exécution d'une activité serait la conséquence d'une réflexion et un apprentissage sur l'expérience passée stockée en mémoire.

Ainsi, un plan de travail est le fruit d'une réflexion sur des expériences antérieures et une prévision d'un objectif particulier.

Par conséquent, les plans ne doivent pas être rigides et doivent accroître la description de la trace d'exécution des processus. Cependant, ils seront toujours incomplets. L'exécution des processus peut en effet être riche en enseignements qu'il suffit d'analyser pour ensuite y appliquer des règles préconçues pour induire des évolutions possibles.

Tous les produits commerciaux étudiés exigent un modèle totalement défini avant d'être instancié. Les changements ne peuvent être intégrés que par une modification statique du modèle. Ils fournissent un degré assez faible d'apprentissage. En effet, le système Milano considère

que les modèles de processus sont des "ressources pour les actions" plutôt qu'une description rigide des pratiques métiers (plans). Le système ADEPT a été proposé pour supporter les changements dynamiques à l'environnement d'exécution. Ce système est orienté processus et non fonctionnel [Dadam 2009]. Dans cette optique il existe également des travaux qui extraient les modèles de processus métier à partir des traces d'exécutions (event log). Ces techniques sont connues sous le nom de process mining que nous détaillerons ultérieurement.

Evolution dynamique des tâches

Les BPMSs devraient supporter l'évolution individuelle des tâches sans la modification de l'objectif global du processus. Dans la réalité les workflow sont un ensemble d'activités collectives qui engendre une coopération et une dépendance mutuelles entre participants. Les workflow fortement structurés (Figure I.5) sont une illustration de telles situations où les participants deviennent de simples maillons d'une chaîne de production sans aucune considération contextuelle.

Le système FLOWer, qui est basé sur le case handling paradigme, [van der Aalst 2005c] propose une solution à la dependance entre activités en présentant aux participants des données pertinentes à toute l'instance du processus à travers des formulaires. En effet, les autres produits commerciaux ne présentent que les données locales à chaque activité sans avoir une vue globale du processus. Ainsi cette vue globale, grace aux données sur toute l'instance permet une évolution au niveau de l'activité sans modification de l'objectif global. La figure I.7 illustre la position du système FLOWer basé sur le paradigme Case-handling par rapport à la structuration des processus métier [van Dongen].

Localisation du changement

Les modifications qu'un processus métier subit ne doivent avoir d'impact que sur un minimum de composants. Par conséquent, la notion de flexibilité est relative à la localisation du changement survenu. En terme de modèle workflow il s'agit d'atteindre une adaptabilité parfaite des modèles en assurant la limitation de la propagation des changements au minimum de composants possibles.

Il serait intéressant d'envisager la possibilité de voir un processus métier comme un ensemble de sous processus encapsulés par des entités interagissant ensemble.

Tous les produits commerciaux étudiés nécessitent l'application des modifications survenues à l'ensemble du modèle. D'après [Adams 2007b] aucun prototype de recherche n'exploite l'idée de la modélisation par sous processus distincts.

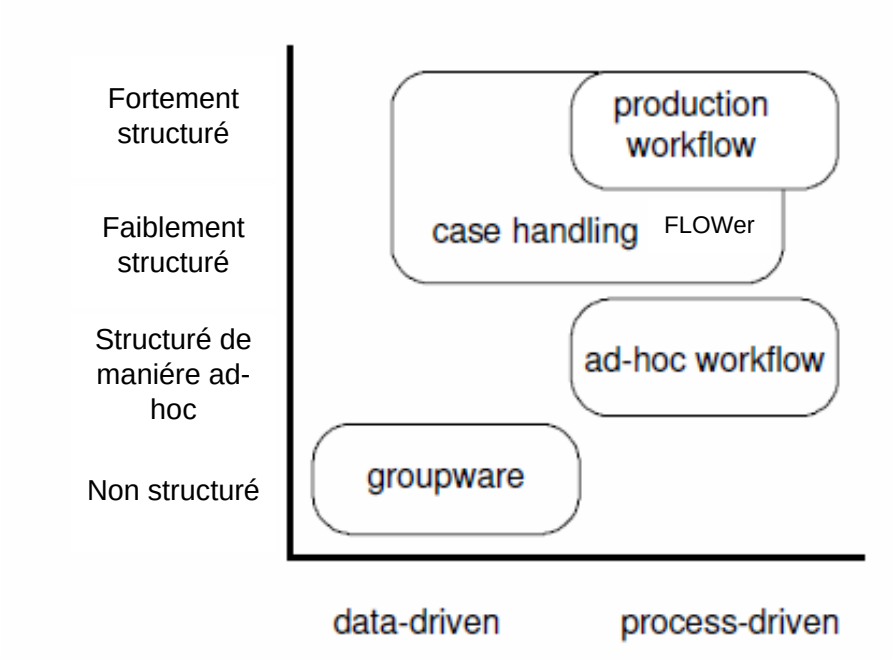


Figure I.7: Position de FLOWer pour la prise en charge des processus structurés .

Compréhensibilité

Le principal facteur limitant l'adoption des solutions workflow par les utilisateurs est la complexité des modèles développés et ceci pour les processus les plus élémentaires. Les systèmes de gestion workflow qui prennent en charge ou nécessitent l'élaboration de modèles complexes ne tiennent pas compte des faits suivants :

- Plus le modèle est complexe plus il est difficile de le modifier, l'améliorer ou le faire évoluer.
- La compréhensibilité pâtit lorsque un modèle contient de nombreux chemins possibles représentés sur un seul plan et
- Il peut arriver que des intervenants dans un processus veulent représenter une activité particulière selon différentes perspectives et à des différents niveaux de compréhensibilité.

La question relative à la participation des divers intervenants dans le développement du processus est importante. Bien que le modélisateur ait une connaissance détaillée des composants et de la structure du modèle, un gestionnaire a généralement une vision plus globale mais n'aura pas l'expertise nécessaire pour le lire. Toutefois, l'acceptation de la solution de workflow peut dépendre de la capacité à fournir au gestionnaire une représentation modélisée d'un processus métier qui soit compréhensible [Carlsen 1997].

Donc, une solution au problème devrait fournir une méthodologie de modélisation qui peut être facilement comprise par tous les intervenants. Par conséquent, la complexité à la phase de conception devrait être minimisée.

Pour faciliter la compréhensibilité, les exceptions devraient être en mesure d'être représentées graphiquement, par opposition à la représentation textuelle, qui est généralement la seule option disponible dans les WfMSs commerciaux. À ce titre, les solutions devraient soutenir l'idée de la modélisation graphique. Le concepteur ne doit pas être forcé d'écrire du code ou de scripts pour atteindre un résultat souhaité. En outre, les processus de gestion des exceptions devraient être formulés et stockés à l'extérieur du modèle de processus qui les concerne [Russell 2007].

L'exploitation des exceptions

Les exceptions workflow ne devraient pas être considérées comme des erreurs, mais comme les événements qui donnent l'occasion d'apprentissage et sont donc un élément important de toute exécution du processus métier. En règle générale, les WfMSs commerciaux considèrent les exceptions comme des erreurs d'exécution.

Bien qu'il y ait eu beaucoup de recherches dans le domaine des exceptions workflow, peu de résultats ont été intégrés par les produits commerciaux. Les travaux de recherches d'Adams [Adams 2005] [Adams 2007a] proposent de s'inspirer de la théorie de l'activité humaine pour utiliser les pattern workflow nommé worklet pour supporter la flexibilité et les exceptions à travers un langage nommé YAWL. Nous y viendrons en détails quand nous présenterons la taxonomie de la flexibilité.

4.3 Discussion

Dans cette section, nous avons défini le terme «flexible» ainsi que la flexibilité dans le contexte des processus métier. Nous avons également décrit les critères fonctionnels que devraient, dans la théorie, satisfaire la gestion des processus métier flexibles par les BPMs. Chacun des produits académiques ou commerciaux tentent de satisfaire l'un ou plusieurs de ces critères en répondant à des axes de recherche différents.

La figure I.8 résume la relation entre les critères de solution abordés dans cette section et les quatre principaux domaines liés aux problèmes relatifs à la flexibilité des BPMSs. Il est évident que plusieurs des critères couvrent plus d'un domaine de recherche, ce qui prouve que la ligne et la distinction entre ces quatre domaines problématiques sont très fines, tandis que d'autres

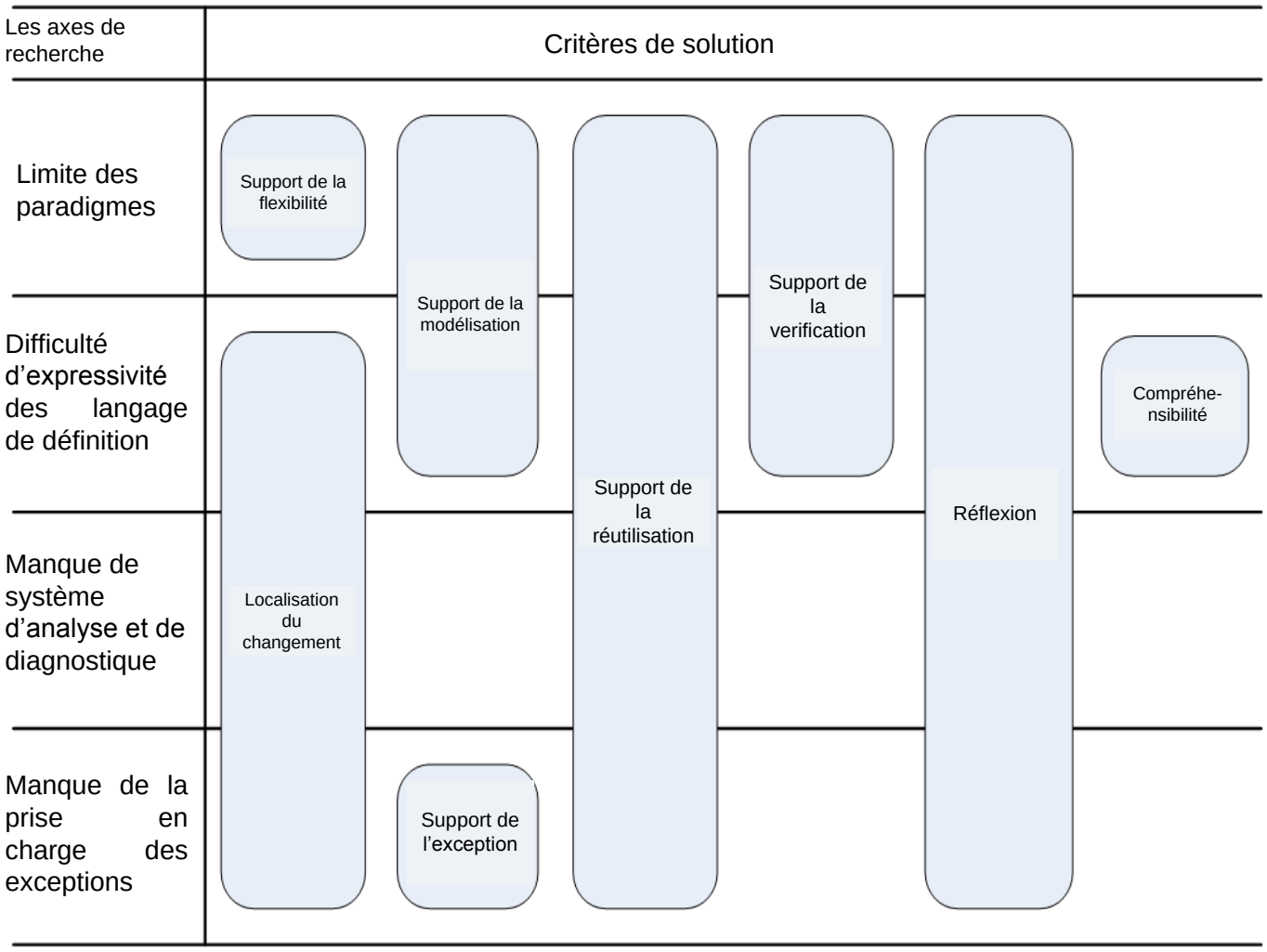


Figure I.8: Représentation des solutions existantes selon les critères fonctionnels.

critères sont directement liés à un domaine en particulier. Cependant, chaque critère doit être considéré comme une solution efficace et globale.

Par exemple, une solution qui a fourni un support à la flexibilité (tels que la possibilité d'ajouter / re-déplacer / modifier les tâches et les séquences) aurait réussi à répondre aux problèmes de «limites des paradigmes». En outre, une solution qui a été construite d'après les framework actuels serait dans les quatre domaines, car il serait nécessaire de se libérer de la rigidité des paradigmes sans grande expressivité des langages de modélisation des processus métier et permettre la déviation par la réflexion et l'apprentissage. L'apprentissage se fait d'après les traces d'exécution et le contexte d'exécution. Dans la section suivante nous allons exposer la phase d'apprentissage et d'analyse pour effectuer un diagnostic à des fins d'optimisation et

de (re)modélisation entre autres.

5 Analyse, diagnostic et découverte des processus métier

Comme le schéma de la figure I.4 de la section 2.4 le montre, il est incontestable que la conception des processus métier est le secteur du BPM qui a suscité une très grande attention pendant les deux dernières décennies. La manière dont les processus métier sont structurés a un grand impact sur le coût et la qualité de leurs produits. Les termes tels que BAM [12], BOM [13], BPI [14] illustrent l'intérêt pour la fermeture de la boucle du cycle de vie de workflow. Ceci est illustré par la figure I.9 qui montre le niveau de support sur quatre années différentes [van der Aalst 2003a]. Les systèmes de gestion de workflows ont commencé par assurer des

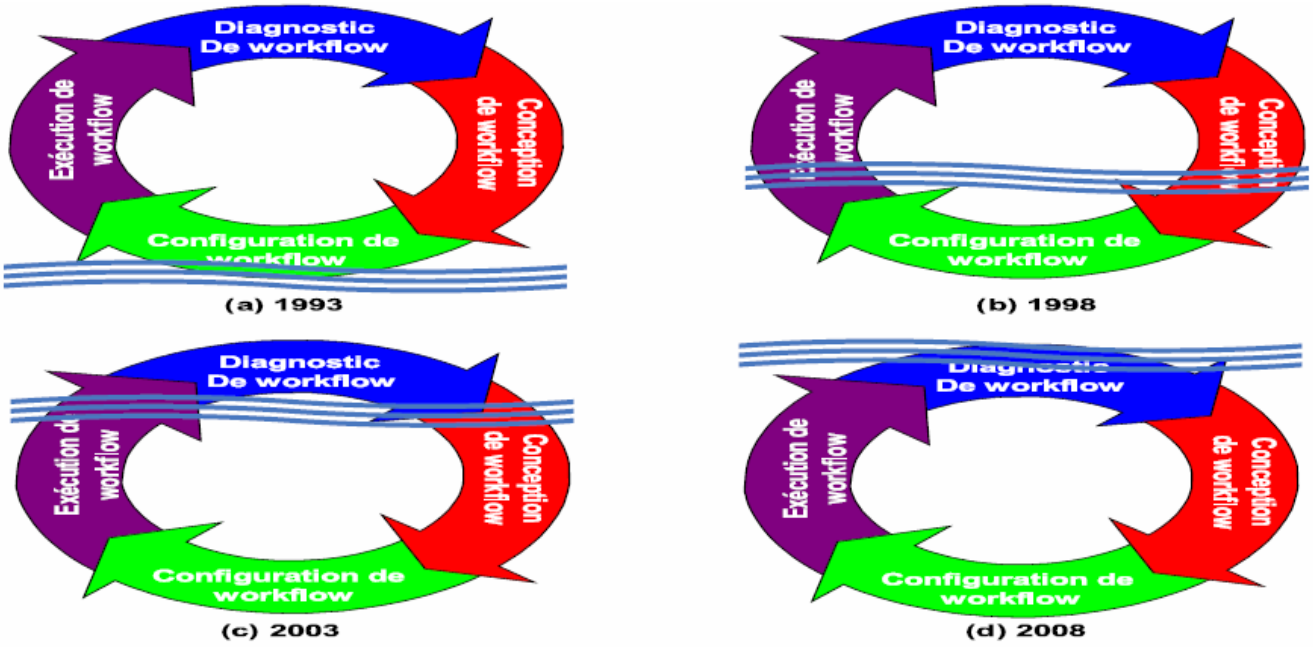


Figure I.9: Le niveau de support du cycle de vie BPM.[Gaaloul 2006]

outils de support et d'automatisation des processus métier (c.à.d, la phase de configuration du système). Depuis les années 90, plus d'emphases étaient mises pour supporter et améliorer la phase de conception. Aujourd'hui, de plus en plus de fournisseurs essayent de clôturer le cycle

[12]Business Activity Monitoring
[13]Business Operations Management
[14]Business Process Intelligence

de vie de BPM en ajoutant la fonctionnalité de diagnostic et de re-ingénierie. Les termes à la mode tels que BAM, BOM, BPI, etc. illustrent ces tentatives [Gaaloul 2006].

Actuellement, beaucoup de fournisseurs de systèmes de gestion de processus sont en train de repositionner leur produit. L'analyse des systèmes de gestion de processus (BPA)[15] devient un aspect important dans le processus de développement et de croissance des systèmes de gestion de processus. Notons que BPA couvre des aspects souvent ignorés par les systèmes de gestion de workflows classiques (par exemple, diagnostic, simulation, etc.).

Par ailleurs, ils existent plusieurs mécanismes de contrôle actifs de processus, également appelés mécanismes de surveillance (monitoring) de workflows, qui analysent des instances de workflows au cours d'exécution.

La surveillance active de l'état actuel des instances de workflows peut répondre à plusieurs objectifs, tels que la génération de rapports d'exceptions, de retard de traitement ou de rapports de détection précoces pour les activités potentiellement en retard de traitement. Elle peut fournir des informations sur l'état des instances courantes de workflows, par exemple pour répondre à une enquête d'un utilisateur au sujet de l'état d'une instance. Cependant elle ne fournit pas d'outils d'amélioration ou de re-ingénierie du modèle de workflow.

5.1 Re-ingénierie des processus métier

Hammer [Hammer 1990]et Davenport et Short [Davenport 1990] étaient les premiers à décrire des approches plus ou moins systématiques pour une amélioration et une refondation entière et radicale de l'ingénierie des processus métier par reconstruction radicale de la conception. Cette approche est désignée avec les termes de "re-ingénierie de processus métiers" par Hammer [Hammer 1990] ou de "re-conception de processus métier" par Davenport et Short [Davenport 1990]. Ces idées regroupées sous l'acronyme BPR (Reconfiguration de processus)[16] ont été largement adoptées par l'industrie.

BPR suit un cycle de vie (voir la figure I.10) qui commence par une initiative, venant la plupart du temps de l'équipe de direction et comporte un certain nombre de phases :

1. 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 détecte là où les méthodes de travail existantes ne produisent pas le résultat désiré.
2. Une fois que le diagnostic a été fait, la phase de (re)conception suit. Les méthodes de travail existantes ne seront pas utilisées comme base de travail : une nouvelle description

[15] Business Process Analysis
[16] le terme anglophone est Business Process Re-Engineerings

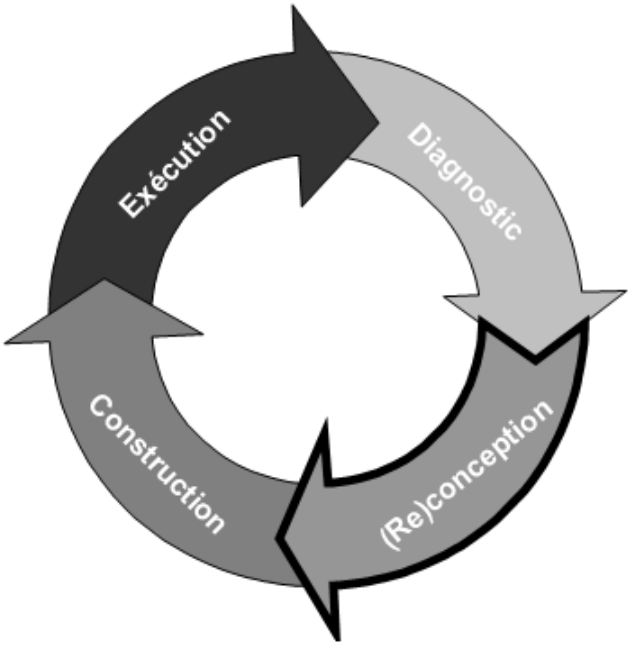


Figure I.10: La re-ingénierie des processus métier

du processus est produite.

3. Dans la phase de (re)construction, le nouveau processus est implanté dans un système de définition de processus (un système de gestion de workflows par exemple).

4. Pendant la phase opérationnelle, la performance des processus est mesurée et évaluée en utilisant des critères prédéfinis. Ceux-ci peuvent bien justifier le lancement d'un nouveau cycle de re-ingénierie.

Après plusieurs décennies d'automatisation, beaucoup d'organismes sont arrivés à la conclusion qu'il en faut beaucoup plus pour réaliser des améliorations réelles. Une approche radicale qui exploite le savoir acquis des expériences passées est donc exigée pour obtenir un plus grand rendement. Ainsi, une méthodologie de conception qui prend l'expérience acquise dans la phase d'exécution du processus existant comme point de départ d'une re-conception, semble la réponse naturelle pour pallier à ces inconvénients et coller au mieux aux besoins. En utilisant cette connaissance on peut définir des objectifs par lesquels le succès des améliorations peut être mesuré. Cette approche nous permet de vérifier la correction des conceptions d'une façon dynamique et interactive en se basant à la fois sur des exécutions multiples et une interactivité accrue avec les utilisateurs, au contraire de la méthodologie de conception classique qui impose un diagnostic statique de la conception [Gaaloul 2006].

5.2 Découverte des processus métier

L'information rassemblée au temps d'exécution peut être utilisée pour dériver un modèle expliquant les événements enregistrés. Un tel modèle peut être utilisé dans la phase de re-conception.

Dans ce cadre, la découverte de processus [17] propose des techniques et des algorithmes inspirés du domaine de fouille de données pour extraire le modèle du processus à partir de ses traces d'exécutions. Le terme de la découverte de processus se rapporte à des méthodes pour extraire une description structurée du processus à partir de ses traces d'exécutions réelles.

Pour comparer la découverte de workflow et l'approche traditionnelle, considérons le cycle de vie de workflow de la figure I.11. Le cycle de vie de workflow se compose de quatre phases : (a) la conception de workflow, (b) la configuration de workflow, (c) l'exécution de workflow, et (d) le diagnostic de workflow. Dans l'approche traditionnelle la phase de conception est employée pour construire un modèle de workflow. Ceci est typiquement fait par un consultant. Le workflow, à la fin de la conception, est configuré comme indiqué dans la phase de conception.

Dans les phases de configuration, on doit traiter des limitations et des particularités du système de gestion de workflow utilisé. Dans la phase d'exécution, des cas (c.à.d, instances de workflow) sont exécutés par le système de workflow comme il a été indiqué dans la phase de conception et réalisé dans la phase de configuration. Dans cette phase, il est possible de rassembler l'information nécessaire au diagnostic qui sera analysée dans la phase de diagnostic.

Au contraire de l'approche traditionnelle, la découverte de workflow permet de comprendre ce qui s'est passé réellement. Son but est d'inverser le processus de conception et de rassembler des données à l'exécution pour soutenir la conception et l'analyse de workflow (c.f. figure I.11) [van der Aalst 2003b][Celino 2007].

Elle commence par la collecte des informations sur les processus de workflows pendant leurs exécutions. Ces traces d'exécutions sont utilisées pour construire des spécifications de processus qui modélisent le comportement enregistré.

Le défi de la découverte de processus est de dériver les "bons modèles de workflows" avec le minimum d'informations possibles. Il existe également la Delta analyse qui détecte l'écart entre le modèle obtenu et ce qui se passe réellement lors de l'exécution.

L'idée d'utiliser les traces d'exécutions pour la découverte de processus a été introduite dans plusieurs travaux [Herbst 2000][Marustera 2001][Schimm 2002][Weijters 2002][van der Aalst 2010c].

Dans leurs premiers travaux [Cook 1998], Cook et autres ont étendu et développé des algo-

[17] Le terme anglophone est "process mining" que nous utiliserons confondemment avec la découverte de processus

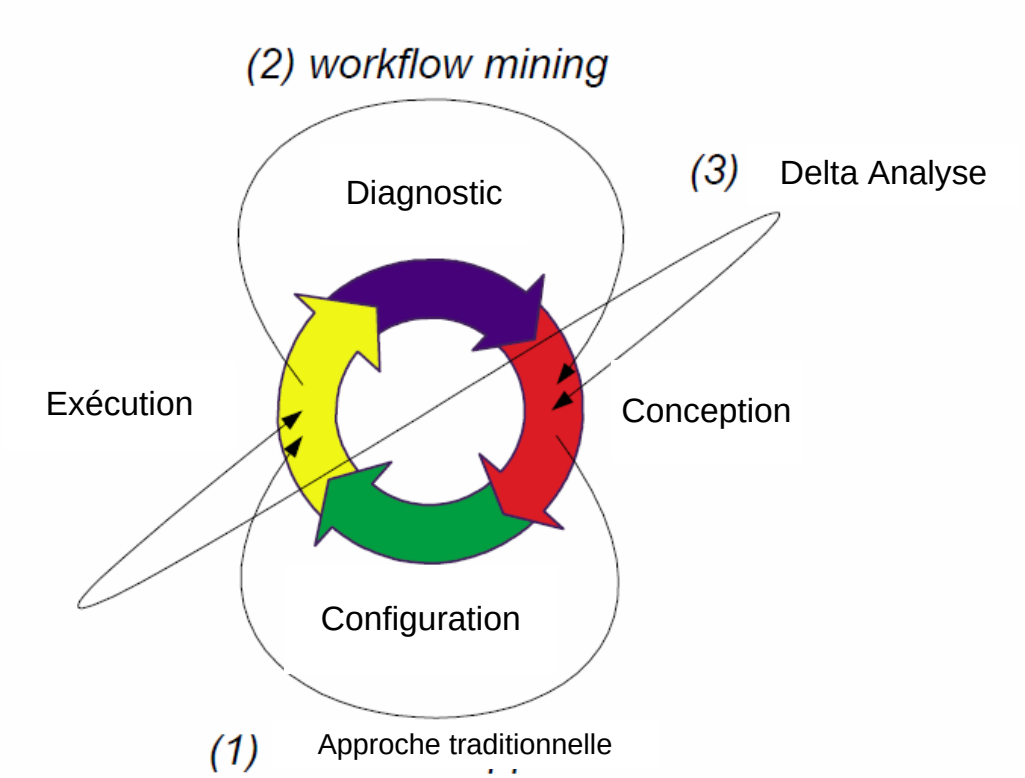


Figure I.11: La découverte de workflow

rithmes pour découvrir des patrons séquentiels dans le domaine de développement du logiciel. Ces modèles ont été exprimés en tant que machines à états finis.

L'aspect remarquable de l'approche de Herbst et autres [Herbst 2000] se trouve dans sa capacité d'aborder les tâches dupliquées. Ils présentent un composant d'apprentissage inductif utilisé pour pouvoir supporter l'acquisition et l'adaptation des modèles de processus séquentiels, généralisant les traces d'exécutions de différentes instances de workflows à un seul workflow couvrant toutes les traces.

Schimm [Schimm 2002] vise à rechercher un modèle complet et minimal. Schimm découvre un modèle du processus par la définition d'un ensemble d'axiomes pour appliquer des règles de réécriture au-dessus d'une algèbre de workflow.

Les algorithmes d'apprentissage ont été utilisés pour déduire des modèles workflow à partir des traces d'exécutions [Herbst 2000].

Aaslt et autres [van der Aalst 2003b] [Günther 2006a] [Günther 2007] [van Dongen] [van der Aalst 201 [van der Aalst 2004] ont développé un algorithme de découverte de processus nommé $\alpha algorithme$ basé sur les réseaux de Pétri.

Les travaux décrits par Aaslt et autres traitent la plupart des constructions des réseaux de Pétri et les classifient en démontrant pour quelles classes de modèles leur approche est sûre de fonctionner.

Bien que la plupart des techniques de découverte se focalisent sur la perspective de la découverte du flux de contrôle, l'exploitation des traces d'exécutions à posteriori sont le coeur de plusieurs autres propositions de recherche dans le domaine des systèmes de gestion de processus. Par exemple, l'analyse du réseau social et ses implications sur la découverte des réseaux sociaux [van der Aalst 2005b].

Bien que le but du process mining soit de faire une analyse à posteriori du processus afin de découvrir le modèle de processus à partir des informations cumulées lors de l'exécution, il possède d'autres fonctionnalités que nous allons présenter dans ce qui suit.

5.3 Fonctionnalités du "Process mining"

Selon [van der Aalst 2008b] l'idée du process mining est de découvrir, de contrôler et d'améliorer des processus par extraction de connaissance à partir de journaux d'évènements (exécution des activités, acteurs, date d'exécutions...etc.). Pour cela trois types de fonctionnalités des process mining sont pris en considération (1) La découverte du modèle original du processus. (2) Le contrôle de la conformité qui compare le modèle à priori du processus avec le comportement observé dans le log afin de vérifier si l'exécution du processus est conforme au besoin (modèle à priori). (3) L'amélioration du modèle du processus en se basant sur les détails extraits des journaux d'événements (voir figureI.12).

Découverte de processus

La découverte de processus, tel que nous l'avons mentionné auparavant, se base sur le principe de la "page blanche" c.à.d que le processus est conçu à partir de zéro. Un modèle conçu "à-priori" n'existe pas. Il est construit à partir des journaux d'évènements[18] (traces d'exécution). Ce modèle sera construit à partir d'algorithmes appliqués à ces traces. Plusieurs de ces algorithmes ont été proposés pour la découverte de processus [van der Aalst 2003b][Marustera 2001][Rozinat 2006] [van Dongen 2009] [Marustera 2001] [Bratosin 2010] [de Medeiros 2007a]. Les techniques de découverte peuvent être utilisées pour découvrir des parties invisibles du processus métier lors de l'exécution [van der Aalst 2010a].

[18]Terme anglosaxon : event logs

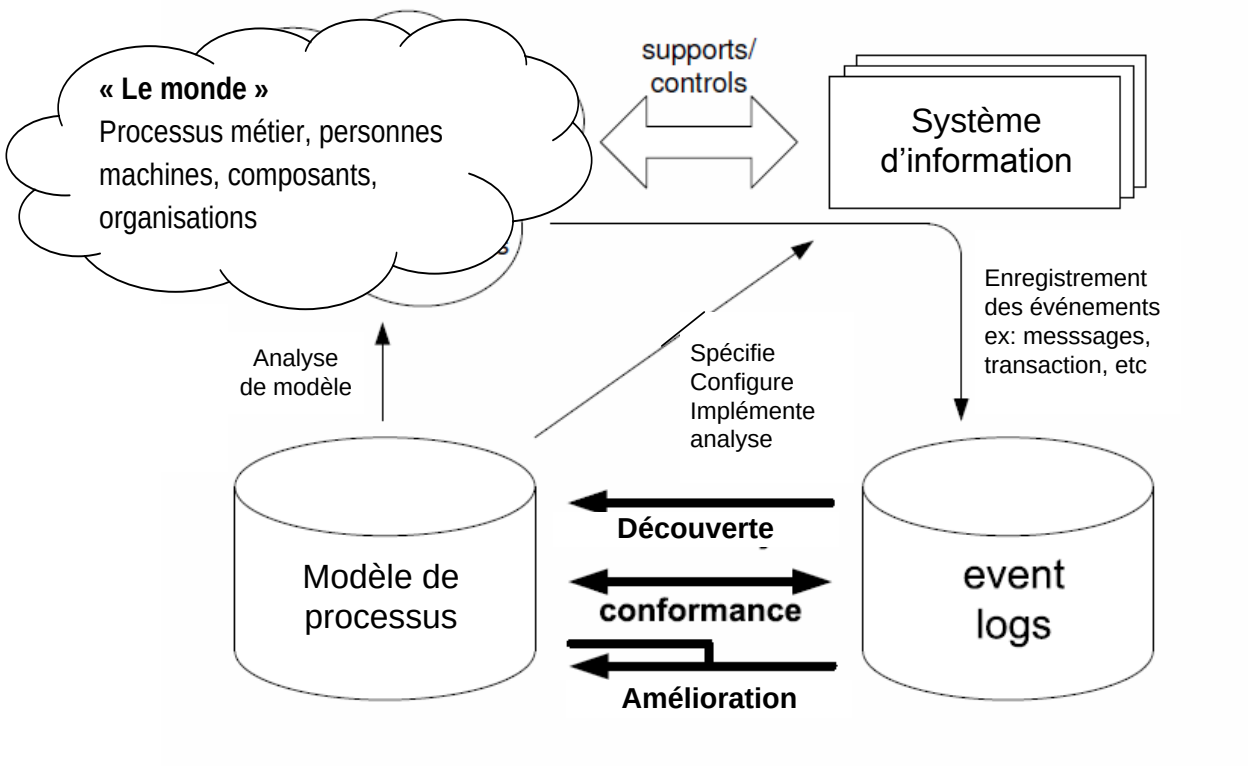


Figure I.12: Les trois types du process mining : (1) Découverte (2) Conformité (3) Amélioration [van der Aalst 2009a]

Contrôle de la conformité

Il s'agit d'une comparaison entre le comportement réel du processus et le modèle existant. Dans ce cas, un modèle "à-priori" existe. L'adéquation entre le modèle du processus et le processus dans la réalité est souvent inexistante. Par conséquent, il est important d'être en mesure de comparer les traces laissées par l'exécution réelles des processus et les modèles de ces mêmes processus métier. Plusieurs travaux et méthodes ont été développés pour vérifier la conformité de la réalité au modèle défini [van der Aalst 2006b] [van der Aalst 2005a] [van der Aalst 2008a] [Rozinat 2005] [Montali 2010]. Ce dernier utilise une technique (LTL Checker) pour la vérification de la chorégraphie de services.

Amélioration du modèle du processus

Dans cette perspective, un schéma de processus préalable existe "à-priori schema" . Ce schéma est étendu avec un nouvel aspect ou une nouvelle perspective. Le but n'est pas de vérifier la conformité, mais d'enrichir le schéma. Il s'agit, à titre d'exemple, de faire une extension d'un

schéma de processus avec des données de performance, c'est à dire, qu'un schéma de processus de départ est utilisé pour refléter les goulots d'étranglement. L'extension peut servir, également, à la détection des dépendances de données qui affectent l'acheminement d'une instance et l'ajout de ces informations pour le modèle sous la forme de règles de décision [Rozinat 2006].

Toutes les techniques et méthodes qui assurent les différentes fonctionnalités du process mining (la découverte, le contrôle de la conformité et l'amélioration du modèle du processus) sont exploitables dans un framework mature appelé ProM [Rozinat 2006],[van Dongen 2005] sur lequel nous viendrons dans ce qui suit.

5.4 Techniques et formalismes du process mining

Le process mining se base sur une multitude d'algorithmes et d'approches qui doivent être implémentés dans un framework sous forme de Plug-in. Ce framework accepte comme entrée les traces dans une certaine forme suivant certaines règles que la figure I.13 illustre.

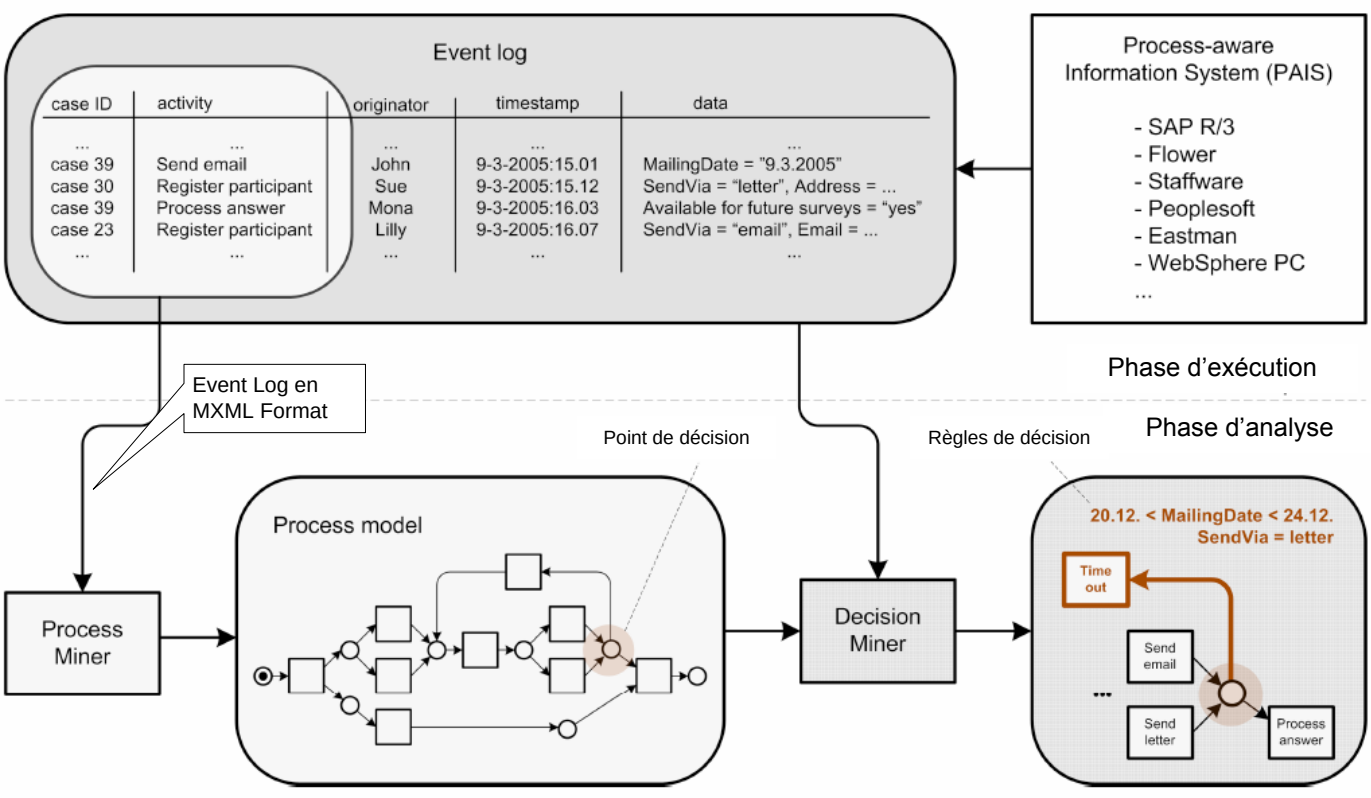


Figure I.13: Les différentes étapes dans l'analyse de processus.

Cette figure (Figure I.13) résume les différents aspects techniques intervenants à savoir :

Le framework ProM

Le framework ProM (**Pro**cess **M**ining) est le seul outil open source [19] supportant le développement d'algorithmes pour le process mining [de Medeiros 2006b]. Il est utilisé par plusieurs groupes de recherche travaillant dans le domaine de la découverte des processus métier. Tous les algorithmes sont développés sous formes de plug-in. Dans l'exemple de la figure I.13, l'algorithme implémenté par [Rozinat 2006] est le "decision miner" sous forme de plug-in. Le résultat de l'application de cet algorithme est un modèle de processus sous forme de réseau de Petri comme sortie du plug-in (voir Figure I.13). Ce modèle sera analysé à des fins de vérification ou d'extraction de règles de décision par un l'algorithme "Decision miner".

Le format MXML

Chaque plug-in (algorithme implémenté) doit avoir une entrée et une sortie. L'entrée représente les traces d'exécution "event logs" laissées par divers systèmes workflow dans la phase d'exécution. Elle doit être dans un format uniforme pour assurer l'interopérabilité entre plusieurs outils. Ce format est le MXML (Mining Extensible Markup Language). MXML [de Medeiros 2007b] considère qu'un event log appelé "L'élément *Workflow Log*" se compose de l'exécution du processus (l'élément *process*) et des informations optionnelles comme des informations sur le programme source générant cet événement (élément *source*) et des données (élément *data*). Chaque processus (l'élément *process*) est composé de zero ou plusieurs instances de processus (l'élément *ProcessInstance*). De même, une instance de processus est composée de zero ou plusieurs activités (l'élément *AuditTrailEntry*). Chaque activité doit avoir au moins un nom (l'élément *WorkflowModelElement*) et un type d'événement (l'élément *Event Type*). Le type d'événement détermine l'état de l'activité. Il existe 13 types d'événements : schedule, assign, reassign, start, resume, suspend, , autoskip, manualskip, withdraw, complete, ate abort, pi abort and unknown. tous les composants du schéma MXML sont représentés dans la figure I.14.

Présentation de $ProM_{import}$

$ProM_{import}$ [Günther 2006a] est aussi un framework open source qui sert à implémenter les plug-in pour la conversion des journaux d'exécution (log) collectés de différents systèmes workflow (tels : Staffware, Flower, Eastman etc) vers le format MXML que le framework ProM accepte comme entrée.

[19] ProM est disponible gratuitement sur le site www.processmining.org

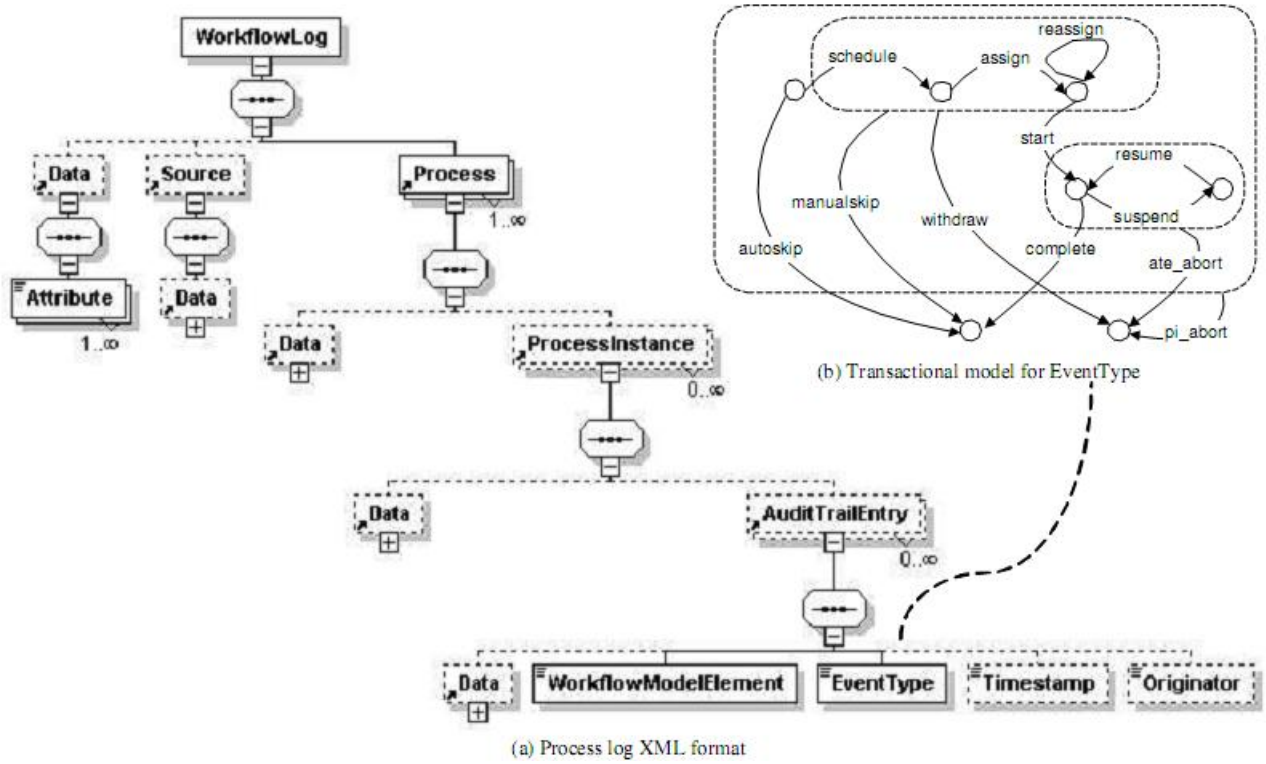


Figure I.14: Description graphique du schéma MXML. [de Medeiros. 2006a]

5.5 Apport du process mining sur la flexibilité des processus métier

Récemment, de nombreux efforts ont été entrepris pour rendre les systèmes de gestion des processus métier plus flexibles. Plusieurs approches dites "adaptatives" pour la gestion des processus métier ont émergé. L'idée de base derrière ces approches est de permettre des changements dynamiques des différents aspects du processus métier, à savoir, perspectives de contrôle de flux, organisationnels, fonctionnels, et d'informations (voir figure I.1) et à des niveaux différents (le niveau d'instance et le niveau type) (voir figure I.15) [Regev 2006]. En effet, des changements ad-hoc survenus au niveau d'instance (ajout ou suppression d'étapes) permettent une adaptation flexible individuelle des instances vers des situations d'exception ou de changement. Cela a comme effet positif l'obtention davantage de traces significatives (fichiers log) par rapport au BPMSs traditionnels.

Toutefois, et jusqu'en 2006 les PMSs adaptatifs n'ont pas abordé des questions fondamentales telles que :

– Que pouvons nous apprendre de ces informations complémentaires tracées lors de l'exécution ?

– Comment peut-on extraire un modèle de processus optimal de ces changements survenus ?

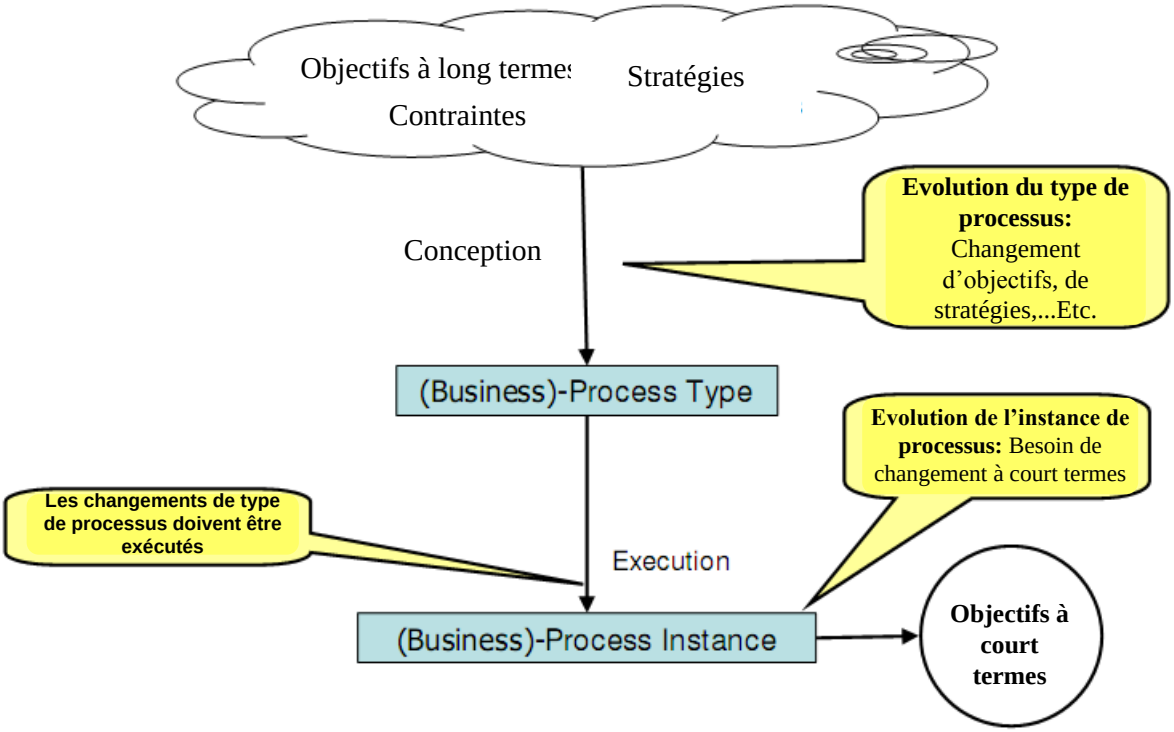


Figure I.15: Niveaux d'abstraction du changement.

Les techniques du process mining ont offert des perspectives prometteuses dans ce contexte [van der Aalst 2010b]. Cependant, elles ne se sont concentrées au début qu'aux perspectives opérationnelles et comportementales [Van der Aalst 2006a].

De toute évidence, l'avantage pratique du process mining dépend du contenu et de la qualité des données des fichiers logs disponibles. Donc, si nous appliquons les techniques actuelles sur les fichiers log des systèmes adaptatifs et flexibles nous obtenons des résultats plus significatifs.

Cependant, l'application des ces techniques n'avaient permis que la capture de la partie réactive et non du stimuli du changement (ce qui provoque le changement [Kumar 2006]).

Dans ce contexte il y a eu des travaux où le process mining a amélioré la flexibilité des processus métier. Nous citons les travaux pour le système ADEPT [Günther 2007]. Il y a eu également la technique du workflow mining incremental qui consiste en la dérivation automatique d'un modèle à partir de son exécution. De cette manière, le modèle devient de plus en plus précis et s'adapte automatiquement aux changements survenus [Kindler 2006].

[Schonenberg 2008c] propose un service de recommandation qui, lorsqu'il est utilisé en combinaison avec des PAISs flexibles, peut appuyer les utilisateurs finaux au cours de l'exécution

des processus en donnant des recommandations sur les prochaines étapes possibles.

Dans [Günther 2006b], on s'interesse à la perspective informationnelle prise en charge par les systèmes flexibles "Case Handling". En effet, les informations sont exploitées afin de parvenir à mieux comprendre le comportement des processus métier dans des environnements flexibles.

Afin d'atteindre une flexibilité des processus métier nous ne pouvons pas passer outre la prise en considération du contexte d'exécution de ces mêmes processus. Nous allons, dans la section suivante, aborder l'informatique sensible au contexte.

6 Applications sensibles au contexte

Dans le cadre de cette thèse, nous nous intéressons aux applications sensibles au contexte[20] pour améliorer la flexibilité des processus métier. Ces processus métier sont caractérisés par le fait qu'ils s'exécutent dans un environnement dynamique. Cet environnement est soumis aux caractéristiques variables des ressources des dispositifs utilisés dans une BPM.

Ceci nous a menés à constater la nécessité de la sensibilité au contexte pour détecter les variations de l'environnement et adapter leurs comportements en conséquence.

Dans cette section, nous commençons par la définition du contexte. Ensuite, nous mettons en évidence l'importance de son utilisation pour augmenter les possibilités des applications, tout en passant en revue différents domaines où le contexte est utilisé . Puis, nous décrivons les caractéristiques des applications sensibles aux contextes.

6.1 Définition du contexte

Il existe dans la littérature plusieurs définitions du mot contexte suivant les domaines de l'ingénierie. Nous commençons par donner une définition générale du terme selon le dictionnaire [national de la recherche scientifique.], puis nous présentons plusieurs autres définitions utilisées dans le domaine de l'informatique.

La définition littérale du mot contexte considère le contexte comme ≪ l'ensemble des unités d'un niveau d'analyse déterminé (phénomène, unité lexicale, phrase...) constituant l'entourage temporel (parole) ou spatiale (écriture) d'une unité ≫. Selon la même source, le terme contexte représente aussi ≪ un ensemble de circonstances liées à un événement qui se produit ≫.

Dans le domaine informatique, il existe deux types de définition du contexte : le premier type définit le contexte en énumérant des entités spécifiques qui représentent le contexte associé à une application, comme la localisation par exemple, alors que le second type définit le contexte

[20]Le terme dans la litterature anglosaxon est Context-Aware Applications

d'une manière conceptuelle et se focalise sur la structure des informations de contexte (une approche formelle).

La définition la plus générale et la plus utilisée est celle donnée par Dey [Dey 2000] qui décrit le contexte comme étant ≪ toutes les informations pouvant être utilisées pour caractériser la situation d'une entité. Une entité est une personne, un lieu ou un objet qui peut être pertinent pour l'interaction entre l'utilisateur et l'application, y compris l'utilisateur et l'application. ≫[21]

Les auteurs se basent fortement sur leurs domaines de recherche pour donner des définitions du contexte. En ce qui nous concerne, nous considérons qu'un contexte observable associé à une entité peut être capturé par le système ou calculé à partir d'autres données (interprété), comme la localisation, l'age ou le sexe de l'utilisateur. Un contexte observable est utilisé par une entité sensible au contexte pour un but déterminé.

Dans le cadre de cette thèse, le contexte est utilisé par notre système dans le but de s'adapter aux changements pertinents de l'environnement d'exécution. En effet, le contexte observable peut changer au fil du temps en prenant des valeurs différentes. Le processus métier doit s'adapter à ce changement de contexte observé dans l'environnement d'exécution.

6.2 Présentation de la notion du "Context-aware computing"

Le contexte a été introduit dans différentes spécialités du domaine informatique, comme le traitement du langage, les systèmes de prise de décisions et l'informatique sensible au contexte (Context-aware computing).

L 'informatique sensible au contexte regroupe les applications ou les systèmes qui utilisent le contexte pour adapter leurs comportements. En se basant sur les travaux de Dey[Dey 2001], [Chalmers 2004] six utilisations possibles du contexte dans un environnement sensible au contexte ont été identifiées :

- Présentation et affichage des informations de contexte comme par exemple la localisation.
- Association d'informations de contexte à une donnée de l'application, comme par exemple l'association des personnes présentes à une conférence et leur localisation au programme de la conférence,
- Configuration sensible au contexte, comme par exemple le fait d'effectuer une impression sur l'imprimante la plus proche, sans intervention de l'utilisateur,
- Lancement d'actions, de réactions selon la valeur du contexte, comme le chargement d'une carte géographique d'une zone quand l'utilisateur se déplace vers cette zone,

[21] any information that can be used to characterise the situation of an entity. An entity is a person, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves.

– Médiation contextuelle qui consiste, par exemple, à utiliser le contexte pour modifier un service fourni,
– Présentation sensible au contexte qui consiste à modifier l'interface d'une application selon le contexte.

Donc, nous définissons un système sensible au contexte comme étant un système qui utilise le contexte pour offrir des informations ou des services pertinents à l'utilisateur. Dans le cadre de cette thèse, nous considérons un système sensible au contexte comme étant un système qui adapte son comportement en fonction du changement de contexte. Cette adaptation consiste à sélectionner l'activité du processus métier adéquate au contexte.

6.3 Caractéristiques des applications sensibles aux contextes

Les informations de contexte sont des informations collectées à partir de plusieurs sources hétérogènes. De ce fait, elles ont des caractéristiques variables. Afin d'utiliser les informations de contexte dans des applications informatiques, il est nécessaire qu'une application sensible au contexte doit satisfaire les caractéristiques suivantes [Loghin 2008] [Behlouli 2006] :

6.3.1 Acquisition du contexte

L'acquisition des informations de contexte peut se faire selon plusieurs méthodes, indépendamment de l'application. Ces méthodes sont classées en trois catégories :

Acquisition par profil : cette méthode consiste à récupérer des informations soit à travers une interface graphique soit par l'intermédiaire d'un fichier de profils ;

Acquisition par sonde : cette méthode consiste à utiliser des sondes (capteurs) pour récupérer les informations de contexte. Il existe deux types de sondes : les sondes physiques et les sondes logiques. Les sondes physiques sont des composants électroniques qui permettent de mesurer des paramètres physiques dans l'environnement comme la température et la localisation, alors que les sondes logiques utilisent des logiciels pour récupérer des informations associées à l'hôte où s'exécute l'application comme la bande passante et la charge de la batterie. Le contexte collecté à l'aide de sondes est dynamique, il est donc nécessaire de mettre à jour les observations fréquemment ;

Acquisition par dérivation : la dérivation ou l'interprétation du contexte consiste à utiliser un ou plusieurs observables pour déduire ou calculer à la volée un contexte de plus haut niveau en utilisant des méthodes d'interprétation.

Pour faciliter l'acquisition, le partage, l'utilisation et la réutilisation des informations du contexte, il est nécessaire de fournir un niveau d'abstraction de ces informations. Cette abstraction peut être obtenue par des modèles de description du contexte.

6.3.2 Modélisation des informations de contexte

Pour décrire la sensibilité d'une application à son contexte d'exécution, il faut déterminer les contextes auxquels cette application est sensible et les décrire dans un modèle. Par conséquent, la modélisation du contexte est la première étape dans le processus de création d'applications sensibles au contexte. Cette modélisation permet à l'application de faciliter l'interaction avec le contexte en fournissant une description abstraite des observables. La diversité des informations de contexte et leur utilisation dans divers domaines engendrent différentes façons de les modéliser. Il existe deux grandes catégories :

Approches basées sur les modèles : Les approches orientées modèle utilisent un modèle formel pour décrire le contexte. De plus, cela offre aux concepteurs d'applications la possibilité de réutiliser le modèle pour d'autres applications. Les modélisations basées sur un langage de balises permettent de décrire des profils ou des contextes simples, sans offrir la possibilité de décrire des relations de dérivation et de dépendance entre ces informations. Les modélisations existantes basées sur UML permettent de décrire des relations entre les informations de contexte mais ne prennent pas en compte la description des dépendances entre ces informations, ni la qualité ou la validité temporelle des données décrites. CML est venu remédier à certains de ces manques en proposant un modèle avec visualisation graphique qui permet de typer le contexte et de décrire un ensemble de relations entre plusieurs contextes observables. Le langage CA-IDL est un langage qui permet de décrire non seulement les situations pertinentes de l'application mais aussi les actions d'adaptation et les conditions de leur déclenchement. Ces conditions représentent des expressions régulières entre les situations pertinentes. Le tableau I.1 [Behlouli 2006] résume les caractéristiques des approches orientées modèles en fonction des informations qu'elles permettent d'exprimer et le langage utilisé à cet effet.

D'après ce tableau, nous constatons que la plupart des modèles ne permettent pas de spécifier à partir de quels capteurs sont effectués les observations, ni de décrire les états du contexte (situations pertinentes) qui nécessitent une adaptation de l'application. Ces modèles n'offrent pas non plus le moyen de décrire la politique d'adaptation à utiliser quand des situations pertinentes sont détectées. Les modélisations basées sur ContextUML ou CA-IDL se distinguent par le fait qu'elles permettent la description des situations per-

Modèle	Modélisation basée sur	Type de contexte	Modélisation des relations de dépendance	Modélisation des règles d'adaptation	Modélisation de l'interprétation	Modélisation des sources du contexte	Modélisation des situations pertinentes
ContextUML	UML	Tout type de contexte	oui	Oui	Oui	oui	Oui
CML	ORM	Tout type de contexte	Oui	Non	Non	Non	Non
ConteXtML	ORM	Profil	Non	Non	Non	Non	Non
CC/PP	RDF	Profil	Non	Non	Non	Non	Non
CSCP	RDF	Profil	Non	Non	Non	Non	Non
CARISMA	XML	Profil de l'application	Non	Oui	Non	Non	Oui
CA-IDL	Grammaire	Profil utilisateur, ressources du dispositif	Non	Oui	Non	Non	Oui

tinentes ainsi que les méthodes d'adaptation nécessaires si ces situations sont détectées. Mais, elles ne permettent pas de décrire la manière d'interpréter un contexte de haut niveau. De plus, les modélisations CA-IDL n'offrent pas la possibilité de décrire les sources du contexte.

Approches basées sur la logique : Les modèles basés sur la logique sont caractérisés par un très grand degré de formalité. Ils utilisent l'algèbre booléenne et la logique du premier ordre pour modéliser le contexte. La logique permet de définir des conditions qui nécessitent de déduire des faits ou des expressions à partir d'un autre ensemble d'expressions ou de faits. Par conséquent, dans les modèles basés sur la logique, le contexte est défini comme des faits, des expressions ou des règles.

Les approches basées sur la logique sont des approches formelles dont l'utilisation a été longtemps restreinte au domaine de l'intelligence artificielle. Ces approches permettent de raisonner sur les informations de contexte pour déduire de nouvelles valeurs du contexte ou pour lancer des réactions au niveau de l'application ou du système. Les approches appartenant à cette catégorie permettent de décrire des relations entre les informations de contexte, mais leurs applications aux systèmes existants restent limitées. Cela est dû à la difficulté de description qu'engendre ce type d'approche.

6.4 Discussion

Dans le domaine du développement des processus métier (gestion des processus métier) la sensibilité au contexte est un axe de recherche relativement récent [Bessai 2009]. Une approche dirigée par les roles sensibles au contexte a été développée [Rolland 2010] avec deux avantages majeurs :

1. La souplesse dans l'attribution des fonctions aux rôles car une fonction peut être assignée à plusieurs rôles possibles selon le contexte de performance.
2. L'autonomie donnée aux acteurs leur permettant de développer des stratégies pour effectuer des opérations.

Selon Saidani [Saidani 2007] [Saidani 2006a] [Saidani 2006b] les acteurs sont affectés à des rôles en fonction de leurs capacités dans un contexte particulier dans un modèle de processus. Par exemple, dans une situation d'urgence ("cas urgent") l'expérience et l'urgence ont un impact sur l'attribution de rôle. Il est préférable d'attribuer un rôle à un acteur expert plutôt qu'à un novice.

Cependant, il est difficile, en ce qui concerne ces approches, d'atteindre une autonomie des acteurs, satisfaire les fonctions et exécuter les opérations dans un contexte statique. Une ap-

proche pour les processus métier sensibles au contexte permet une adaptation aux changements du contexte des processus. Ces approches ne se sont intéressées qu'à l'impact que pouvait avoir le contexte sur l'attribution des roles aux acteurs et ceci à un niveau de modélisation [Bessai 2008].

Notons également que ces travaux se base sur un modèle qu'ils ont proposé connu sous le nom de modèle à la carte pour la flexibilité des processus métier. Ce modèle a inspiré également d'autre travaux tel que ce lui de [Bentellis 2010] pour atteindre la flexibilité.

Dans [Rosemann 2006] un ensemble de variables de contexte a été défini ainsi qu'un certain nombre de problématiques dans ce domaine.

7 Conclusion

Au début de ce chapitre, nous avons donné une définition du terme processus métier puis nous avons montré l'importance de sa prise en compte dans les systèmes d'information. Cette importance croît de plus en plus avec l'évolution technologique et l'ouverture croissante et inévitable des entreprises. Le cycle de vie des processus métier a été présenté. Dans ce contexte nous avons également mis en évidence l'obligation des systèmes d'information d'être sensibles à ces processus (les PAISs). Dans la section 3 les catégories des processus métier ont été exposées.

Dans ce contexte la nécessité au processus métier d'être flexible est vite apparue évidente. Ceci a fait l'objet de la quatrième section où nous avons défini la flexibilité. Nous avons également décrit les critères que devraient satisfaire les systèmes de gestion de processus métier flexible. Après avoir analysé ces critères, nous avons conclu que les produits commerciaux ou académiques actuels peinent à satisfaire tous ces critères. L'apprentissage est la première démarche dans le processus de l'amélioration de cette flexibilité.

La découverte et l'analyse des traces d'exécution des processus métier ont fait l'objet de la section 5 qui illustre l'intérêt pour la fermeture de la boucle du cycle de vie de workflow.

Pour exploiter pleinement les avantages que présentent les techniques du process mining la dernière section de ce chapitre démontre la nécessité des BPMS d'être sensibles à leurs contextes où les techniques d'observation et de modélisation des processus ont été brièvement présentées.

C'est sur le process mining, le contexte et la flexibilité que porte le travail que nous allons exposer dans le deuxième chapitre.

Chapitre II

Une solution sensible au contexte pour la flexibilité des processus métier

1 Introduction

Nous nous intéressons dans ce chapitre à la manière dont nous devons atteindre une flexibilité des processus métier dans l'environnement d'exécution. En effet, bien que les systèmes de workflows actuels soient établis par des modèles de processus explicites, une des difficultés est née du fait que les processus ne sont pas tous explicitement définissables. En effet, la diversité des tâches et des intervenants (concepteurs, utilisateurs, gestionnaires, etc.) et également des conditions d'exécution (contexte) ont contribué à cette difficulté.

Le processus décrit dans la phase de modélisation ne reflète pas toujours la réalité. Par conséquent, l'alignement des processus aux conditions réelles d'exécution exige une attention et une action continues. Pour maintenir cet alignement il est important de détecter les changements dans le temps, c'est-à-dire, les déviations du comportement décrit ou prescrit.

Cependant, il existe plusieurs manières de détecter les changements et les déviations dans le comportement d'un processus métier. Cette diversité a engendré une taxonomie de la flexibilité des processus métier. Une multitude de travaux se sont intéressés à chaque type de flexibilité. Cette taxonomie ainsi que les travaux relatifs à chaque catégorie seront présentés dans la section 2.

Dans cette thèse, nous nous intéressons à la flexibilité par spécification partielle[1]. La justification de ce choix sera expliquée dans cette même section.

Nous proposons une approche qui se base sur la prise de décision du comportement à suivre du processus dans son environnement d'exécution. Elle est liée au contexte d'exécution. Nous considérons le processus métier comme un ensemble d'entités encapsulant le métier de chaque

[1] Underspecification

activité. Le choix de l'exécution d'une entité bien précise se fait selon le contexte d'exécution au moment de l'exécution.

2 Travaux antérieurs

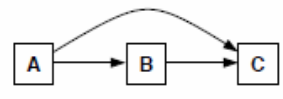
Plusieurs travaux de recherche ont tenté de répondre aux besoins des systèmes de gestion des processus métier d'être flexibles. Ils existent plusieurs types de flexibilité de processus métier. Afin de présenter les travaux et produits commerciaux existants, nous devons tout d'abord exposer cette taxonomie afin de présenter tous les bilans et leurs synthèses [Boukhebouze 2010].

2.1 Taxonomie de la flexibilité des processus métier

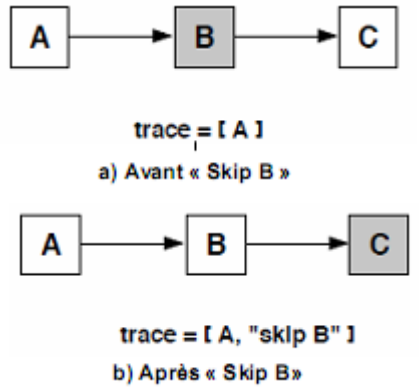
Plusieurs travaux ont été menés pour tenter de proposer une taxonomie générique de la flexibilité des processus métiers. Dans ce contexte, deux grandes taxonomies ont été proposées :

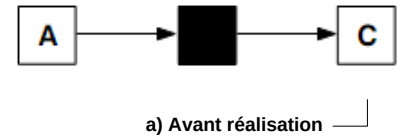
1. Une taxonomie , selon [Regev 2005] [Regev 2006] , qui s'intéresse aux changements qui peuvent survenir durant le cycle de vie d'un processus métier. Elle comprend trois dimensions (le niveau d'abstraction, l'objet du changement, les propriétés du changement).

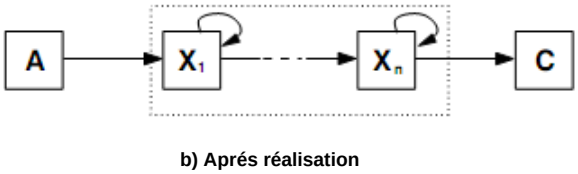
2. Une taxonomie, selon [Schonenberg 2008b] [Schonenberg 2008a], qui s'interesse à la flexibilité du point de vue opérationnel. Ses differents types et concepts sont résumés dans ce qui suit :

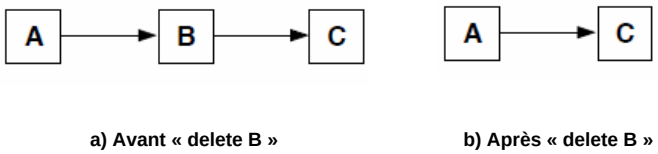


Flexibilité par conception : Elle offre la possibilité d'exprimer des chemins d'exécution alternatifs lors de la modélisation du processus, en permettant l'expression du parallélisme entre activité, le choix, l'instanciation multiple ou encore l'annulation de l'exécution d'une activité.









Flexibilité par déviation : Une instance de processus peut dévier, lors de l'exécution, du chemin défini dans la phase de modélisation. Cette déviation n'influence en rien le modèle de départ et les changements ne sont pas permis sur le modèle.

Flexibilité par spécification partielle : Il s'agit d'offrir la possibilité d'exécuter un processus partiellement spécifié (modélisé). En effet, certaines parties du processus ne sont sélectionnées que tardivement (à la phase d'exécution). Cette sélection tardive d'un fragment du processus se fait entre plusieurs alternatives. Elle est appelée sélection à la volée.

Flexibilité par changement : Dans certains cas, des évènements peuvent paraître à l'exécution sans être modélisés avant. La flexibilité par changement permet de modifier un modèle de processus donné à l'exécution où une ou plusieurs instances émigrent vers le nouveau modèle de processus.

2.2 Classification des méthodes et solutions pour la flexibilité

Plusieurs travaux de recherche ont tenté d'atteindre la flexibilité des processus métiers dans le spectre de la taxonomie précédemment exposée. Dans cette optique, cette section vise à appliquer cette taxonomie à certaines réalisations de PAISs (ADEPT, YAWL,FLOwer et Declare) ainsi que les approches académiques sous-jacentes. La sélection de ces PAISs était basée sur le critère de support de la flexibilité, ce qui exclut la majorité des systèmes workflow et produits commerciaux. Le tableau II.1 présente une évaluation de ces produits en termes de prise en charge d'un type de flexibilité spécifique [Schonenberg 2008a]. En effet, le signe (+)

désigne une prise en charge totale de la flexibilité désignée par un le produit concerné. Le signe (-) reflète la non prise en charge d'un type de flexibilité.

Certains de ces travaux ont proposé des concepts avancés de modélisation de processus, d'analyse et de vérification, dont le projet ADEPT [Günther 2007] [Dadam 2009] [Reichert 2009]. Ces concepts permettent la définition explicite du contrôle et des données. Il permettent également l'assignation des acteurs et des ressources ainsi que l'expression des contraintes temporelles. Dans cette perspective, ADEPT offre un ensemble d'opérations pour la définition des changements des instances[2] de processus métier assurant la propagation des effets des changements au niveau abstrait[3]. ADEPT assure la flexibilité par changement et conception (voir tableau II.1).

D'autres travaux ont abordé la problématique de la flexibilité des processus métier, en utilisant notamment l'approche déclarative basée contraintes plutôt que l'approche procédurale [van der Aalst 2009b] [Pesic 2007] [Pesic 2009]. Declare est un prototype de WfMSs pour la modélisation des processus faiblement structurés. Il supporte la flexibilité par changement, par conception et la violation de contraintes qui est une catégorie de la flexibilité par déviation (voir tableau II.1). Cependant, l'approche basée contraintes supporte la gestion des processus métier simples et peu structurés. L'utilisation de cette approche sur des processus métier complexes réduit et de manière significative l'efficacité et l'utilité du système.

Le case handling est un nouveau paradigme proposé par Van der Aalst dans [van der Aalst 2005c], pour modéliser d'une manière flexible les processus métier. Contrairement aux autres langages impératifs, qui utilisent des structures de contrôle de processus prédéfinis pour déterminer "ce qui doit être" au cours de l'exécution du processus, le case handling se concentre sur "ce qui peut être fait" pour atteindre un objectif métier.

Pour cela, ce paradigme se base sur les données du processus (appelé cas) pour permettre aux concepteurs de choisir d'exécuter (Execute), de passer (Skip) ou de refaire (Redo) les activités du processus en fonction de la disponibilité des cas. Ce paradigme a été concrétisé dans le système de gestion de Workflow FLOWer. Notons que FLOWer est le seul système (basé langage impératif) à supporter la flexibilité par déviation (voir tableau II.1).

Notons également que FLOWer et Declare se complètent dans le support de la flexibilité par déviation. Ce paradigme a été présenté dans le chapitre précèdent afin d'illustrer la nécessité de prendre en charge l'évolution dynamique de l'activité.

Le système YAWL[4], basé sur des fondations formelles, est la plus récente des initiatives dans

[2]Instance change
[3]Type change
[4]Yet An other Workflow Language

ce domaine [der Aalst 2005]. En effet, ce système tente d'atteindre des objectifs prometteurs dans le support de tout type de flexibilité (voir tableau II.1). Le langage YAWL est une extension de la syntaxe des réseaux de Petri. Il supporte tous les patrons de workflow ou flux de contrôle[5]. YAWL est le seul système(YAWL est un langage de modélisation et d'exécution) à supporter la flexibilité par specification partielle. Cependant, il ne supporte le changement qu'au niveau de la modélisation de processus.

Systèmes workflow	Flexibilité par conception	Flexibilité par déviation	Flexibilité par spécification partielle	Flexibilité par changement	Technologies associées
Declare	+	+	-	+	Langages déclaratifs
FLOwer	+	+	-	-	Case handling
YAWL	+	-	+	-	Worklet
ADEPT	+	-	-	+	Expression des contraintes temporelles

Tableau II.1: Evaluation des produits considérés flexibles

2.3 Synthèse

En plus des critiques apportées séparément à chacune des approches décrites ci-dessus, nous avons pu constater le manque, et dans certaines catégories l'absence, de support de la flexibilité par specification partielle, exception faite de YAWL qui ne le fait qu'au niveau modélisation. Notons que la flexibilité par spécification partielle considère que des parties du processus (fragments[6]) ne sont spécifiées qu'à l'exécution où la selection s'effectue.

La plupart des travaux mentionnés dans la section précédente ne considère pas le contexte d'exécution dans leurs quêtes de flexibilité et d'adaptabilité[7].

Dans le cadre de cette thèse, notre travail apporte deux contributions qui couvrent les principaux manques dans la littérature :

– La considération du processus métier comme un ensemble d'entités indépendantes et réutilisables facilitant le changement et la flexibilité.

[5]Terme anglo-saxon : Workflow pattern
[6]Dans l'utilisation de YAWL fragment est appelé Placeholder
[7]les travaux de Nurcan n'étaient pas cités dans la section précédente n'ayant pas de produit ou de prototype à évaluer

– La prise en charge du contexte d'exécution en percevant des règles métier dans l'environnement d'exécution par l'analyse des traces d'exécution (process mining).

Dans le reste de ce chapitre, nous donnons une présentation globale de notre approche en définissant ses vues principales.

Nous définirons par la suite notre représentation du processus métier en entités indépendantes que nous détaillerons. Enfin, nous nous intéresserons au contexte d'exécution à travers la spécification des règles métier.

3 Présentation globale de l'approche proposée

La flexibilité par spécification partielle est la capacité d'exécuter des parties de processus non définies dans la phase de modélisation. Dans cette perspective, nous voulons améliorer la flexibilité des processus métier en les considérant comme un ensemble d'entités coordonnées afin d'atteindre un objectif.

L'évolution vers la décomposition a nécessité la mise en oeuvre de SI flexibles, adaptables à l'évolution des produits et des processus métier, ce qui a motivé un intérêt accru pour les modèles et les méthodes de réutilisation. Par conséquent, la communauté de l'ingénierie des SI a conclu que la réutilisation des composants, et plus particulièrement sur les connaissances de domaine, constitue une approche puissante pour le développement des SI.

La réutilisation des connaissances de domaine, et plus particulièrement de celles issues d'un métier donné et qui constituent un domaine métier, est intuitive, à partir du moment où l'on se rend compte que certains domaines d'activité conduisent à la manipulation fréquente de concepts similaires.

C'est dans ce sens que notre approche a été proposée [Zerari 2010a]. De tels composants sont ainsi utilisés par plusieurs processus métier dans une entreprise.

Par conséquent, notre préoccupation dans ce travail de recherche concerne la considération du processus métier comme un ensemble d'entités. Ces entités encapsulent le métier de chaque activité de ce processus. Elles sont réutilisables, exécutables selon le contexte d'execution, coordonnant pour atteindre un objectif faisant émerger la flexibilité et l'adaptabilité.

Dans cette section, nous présentons l'approche proposée dans un cadre global (voir figure II.1). Ce cadre permet d'illustrer les deux vues de notre système. Il s'agit d'exposer ces deux vues qui expriment deux angles interdépendants et complémentaires d'un même système :

– Une vue de l'action que nous appelons vue locale.

– Une position de cette même action dans une représentation de son contexte que nous

appelons vue globale

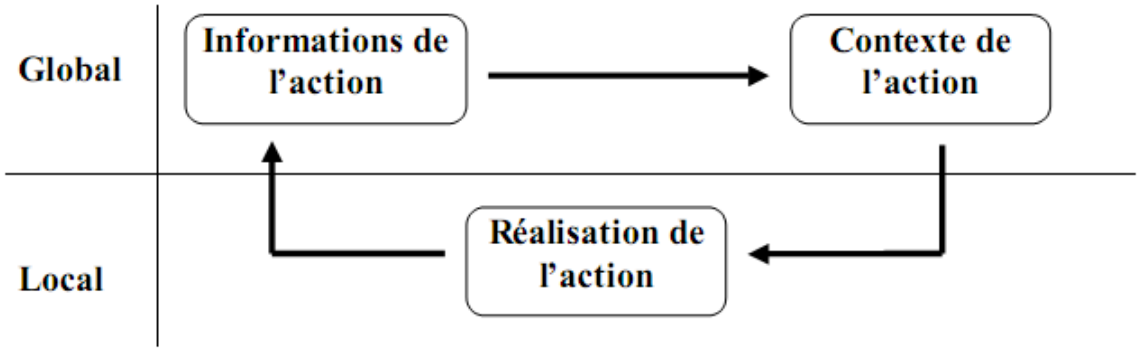


Figure II.1: Différentes vues de l'approche proposée

Les notions de local et de global sont à l'image du pilote et du co-pilote. Le pilote a une vue de la route sur laquelle il conduit l'automobile et le co-pilote a une vue de cette même route sur une carte qui la situe parmi d'autres éléments. On dira que le pilote a une vue locale de la route et le co-pilote une vue globale. Le local est une vue de l'action et le global une position de cette même action dans une représentation de son contexte.

3.1 Vue locale du système

La vue locale est une vue de l'activité dans le sens où elle définit ce qui est réalisé à un moment donné. Une activité est considérée comme un ensemble de tâches identifiables du processus aux entrées et aux sorties clairement définies et dont la valeur ajoutée est mesurable. Si nous considérons l'exemple du processus métier "Commande de produit" composé de plusieurs activités entre autres "Facturer", cette activité aurait des entrées (Pré-condition) et une ou plusieurs sorties (résultats : facturation terminée) sans connaître la finalité d'un tel résultat sur la totalité du processus métier. Dans l'exemple du pilote, ce dernier est en mesure d'agir sur sa conduite en fonction des éléments topographiques de la route qu'il perçoit directement. Par contre, il lui est impossible d'adopter une conduite en fonction d'un élément futur dont il n'a pas la perception directe.

3.2 Vue globale du système

Pour rester dans la même optique du pilote, la vue globale c'est la carte du co-pilote. Elle positionne une action dans une représentation de son contexte, c'est à dire dans l'ensemble des éléments qui conditionnent sa réalisation. La vue globale n'est pas une activité mais le

positionnement d'une activité dans son contexte de réalisation. Dans l'exemple du pilotage d'une automobile, le co- pilote va interpréter l'action de conduite du pilote et la renseigner en fonction de sa position sur la carte.

3.3 Relation entre vue locale et vue globale

La relation qui existe entre le global et le local est qualifiée de "réciproque" car le local communique au global les éléments qu'il perçoit et le global les positionne dans leur contexte pour renvoyer au local des informations sur l'évolution de son action. En positionnant ces éléments, il positionne l'action dans son contexte de réalisation et renseigne cette même action. La relation réciproque entre le local et le global peut être résumée en trois phases qui sont les suivantes :

- **Communication du local au global** : Le local envoie au global des informations qu'il perçoit du fait de sa connaissance du métier de l'activité.
- **Position du local par le global :** Le global interprète les informations qui lui sont transmises par le local et les positionne selon le contexte.
- **Choix du local par le global :** Après avoir positionné le local dans un contexte d'exécution, le global choisit le local à exécuter en mentionnant les éléments auxquels il va falloir faire face et qui vont conditionner son choix.

La relation réciproque du local et du global participe à l'émergence de la flexibilité car elle produit une contextualisation et une formalisation des actions individuelles pour les différentes parties du processus métier.

La suite de ce chapitre portera sur notre représentation du processus métier en un ensemble de locaux, encapsulant le métier de l'activité, positionnés par un global selon un contexte d'exécution.

4 Représentation du processus métier

Une représentation du processus métier décrit la manière dont nous concevons son fonctionnement. L'approche "Composant Métier" a été proposée pour rendre plus facile, plus rapide et moins coûteuse la mise en place de systèmes d'information dans le domaine auquel ils se rapportent.

La recherche dans le domaine des composants métier a donné naissance à de nombreuses technologies à base de composants, qui sont aujourd'hui proposées et utilisées dans tous les secteurs d'activité.

Un composant métier est considéré par [Saidi 2009] comme une représentation d'un concept actif dans un domaine métier. Ainsi, les composants métier sont utilisés pour définir des concepts de type ≪ `produit` ≫ (par exemple, bibliothèque, compte, abonné...), ou pour définir des ≪ `processus` ≫ pour les concepts qu'ils représentent (par exemple, processus d'emprunt, processus de réservation d'un ouvrage, processus d'inscription d'un abonné...). Par conséquent il n'est nullement question de considérer le processus métier comme un ensemble de composants métier.

Dans ce travail, nous nous intéressons à la décomposition du processus métier lui même.

4.1 Vers une décomposition du processus métier

Concevoir un processus métier de façon modulaire revient donc à le décomposer en sous entités indépendantes et d'une taille raisonnable. Notre vision de cette conception modulaire ne considère pas ces modules comme des ≪ `produits` ≫ ou des ≪ `processus` ≫ comme c'est le cas dans l'approche ≪ composant métier ≫. La figure II.2 représente la vision d'une telle approche. Nous considérons plutôt un composant métier comme une activité du processus

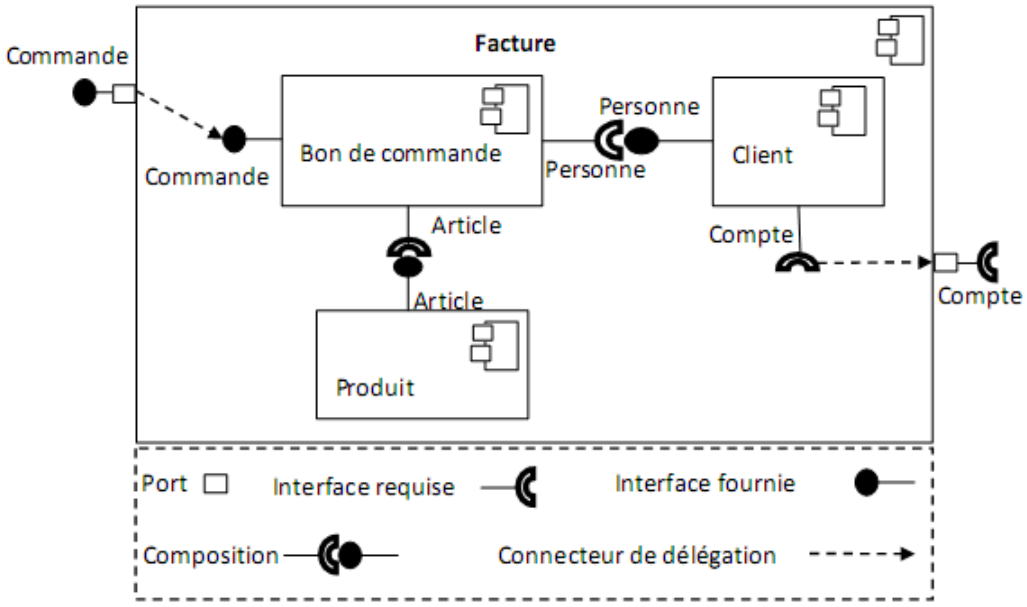


Figure II.2: Exemple d'un composant métier
[Saidi 2009]

métier global. La figure II.3 représente un ensemble de processus métier dans une banque en utilisant le diagramme de caractéristiques. Ce dernier fait la distinction entre ce qui est

obligatoire (commun) et ce qui est optionnel. Nous considérons, par exemple, le processus ≪ *Demande_d'emprunt* ≫ comme un ensemble de composants métier.

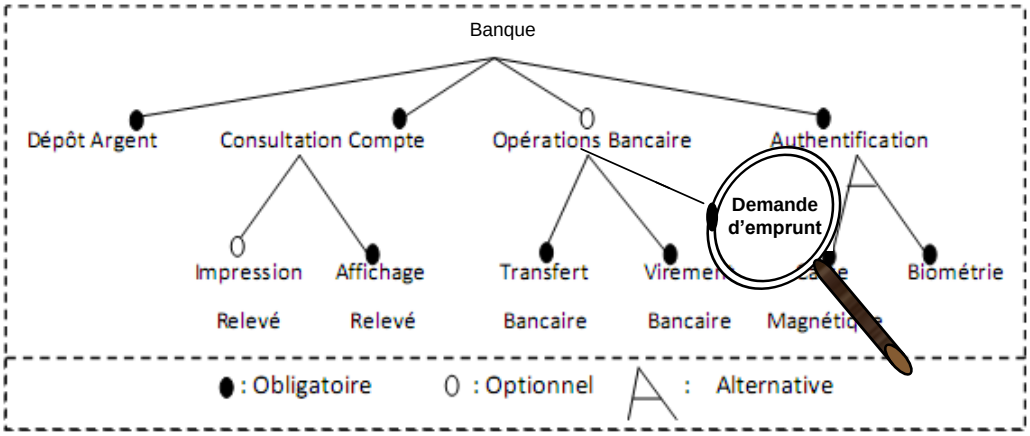


Figure II.3: Représentation d'un ensemble de processus en diagramme de caractéristiques

Une représentation classique d'un tel processus serait un ensemble d'activités dans un language de modélisation (Figure II.4). En ce qui nous concerne ces activités sont considérées comme entités indépendantes, exécutées selon le contexte choisi par le global. Cet exemple n'est donné qu'à titre indicatif. Plus de détails sur ces entités ainsi qu'un processus métier plus démonstratif seront exposés ultérieurement.

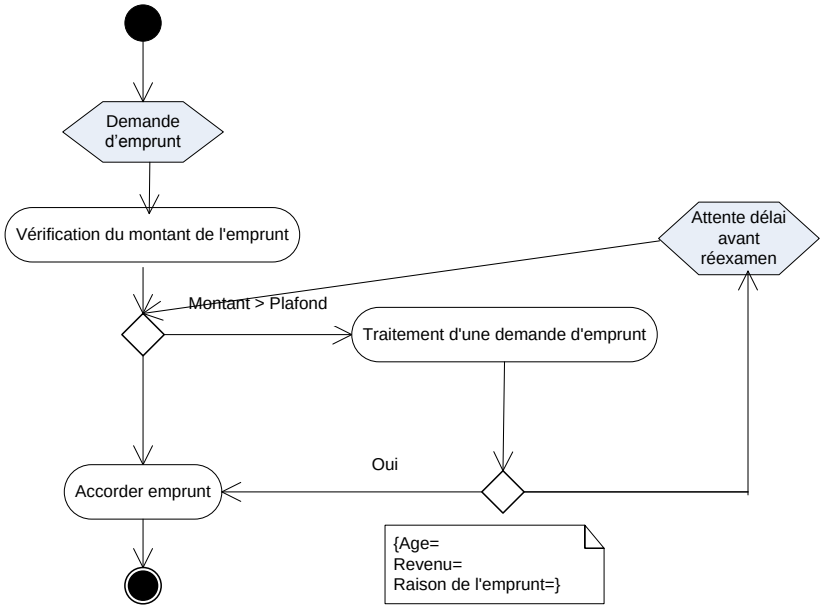


Figure II.4: Représentation d'un processus métier de manière classique

4.2 Processus métier : Regroupement d'entités

Le regroupement des entités pour faire émerger un comportement flexible du processus métier correspond tout simplement à la composition dynamique de ces unités. Cette composition dépend du contexte d'exécution perçu par un module que nous nommons ≪ *Administrateur* ≫ (la vue globale du système) qui, selon le contexte et les modifications apportées par les autres unités, décide quelle entité sera exécutée (Voir figure II.5).

Par ailleurs, les différentes unités (entités) sont connectées les unes avec les autres et communiquent avec ≪ *Administrateur* ≫ via des connexions évènementielles. L'architecture évènementielle du système obtenue réduit les contraintes de couplages entre les différents composants. De ce fait, le système est reconfigurable, dynamique et extensible. Parce que, les interactions de ses composants se faisant via des évènements, les composants deviennent indépendants les uns des autres et sont capables d'opérer sans aucune connaissance préalable des composants avec lesquels ils interagissent. De ce fait, un changement dynamique d'unités peut survenir pendant l'exécution du système sans altérer son fonctionnement.

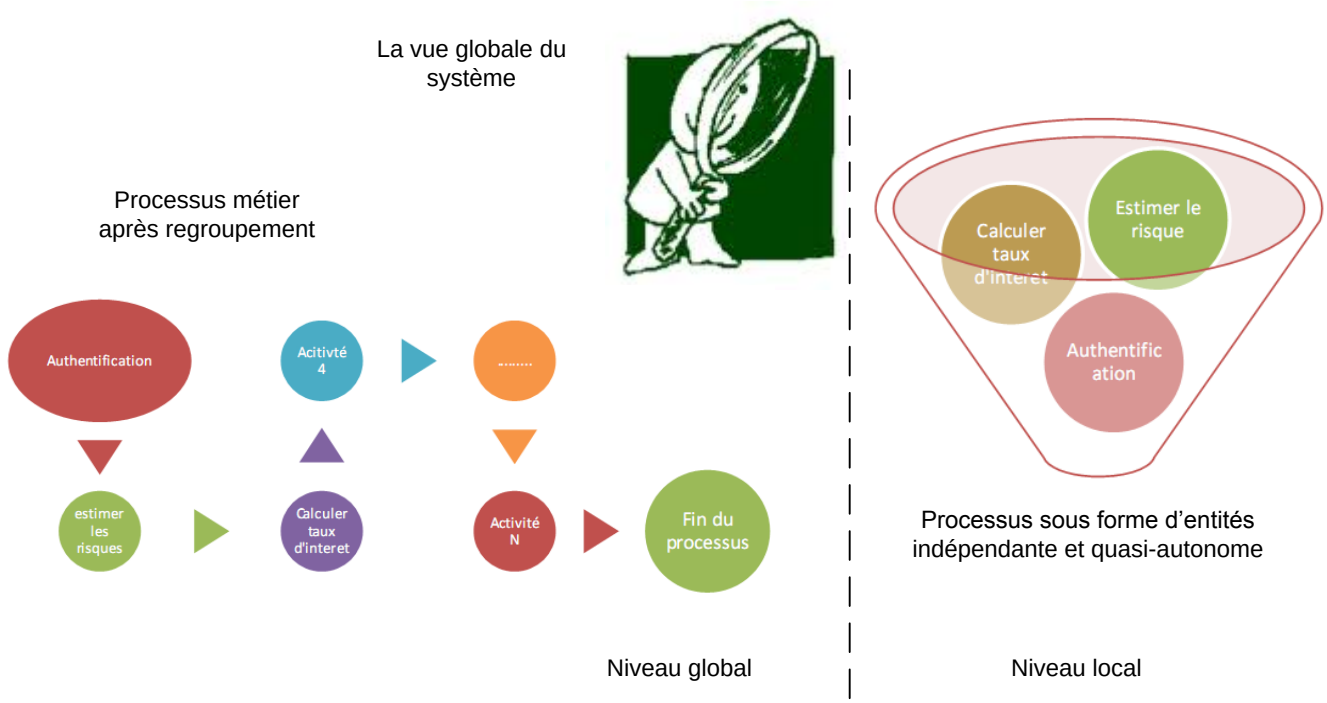


Figure II.5: Représentation du regroupement d'entités pas l'Administrateur

L'architecture de notre système fera l'objet d'une description détaillée dans le prochain chapitre.

5 Récupération du contexte d'exécution

Dans le chapitre I section 6.2, nous avons évoqué les six utilisations possibles du contexte dans un environnement sensible au contexte. Notre système utilise les informations du contexte afin d'effectuer un lancement d'actions et de réaction selon la valeur de celui ci.

Il existe plusieurs définitions d'un système sensible au contexte. [Dey 2001] donne une définition plus générale en incluant même un système qui ne fait que collecter le contexte pour le fournir à une application. Par conséquent, [Dey 2001] stipule qu' ≪ un système est sensible au contexte s'il utilise le contexte pour fournir des informations pertinentes et/ou des services à l'utilisateur. Cette pertinence des données dépend des tâches de l'utilisateur ≫ .

Un système sensible au contexte est caractérisé par des fonctionnalités qui lui permettent d'interagir avec des capteurs, d'analyser les données collectées et d'agir en conséquence (voir figureII.6). Ceci nous a inspiré dans notre démarche de définition des différents modules de notre système.

Un système sensible au contexte peut utiliser un modèle de description du contexte qui lui offre un haut niveau d'abstraction du contexte (Voir chapitre I section6.3.2).

Dans cette section, nous décrivons les fonctionnalités de notre système sensible au contexte. Nous montrons aussi comment adapter les concepts de ce genre de système pour le contexte des processus métier.

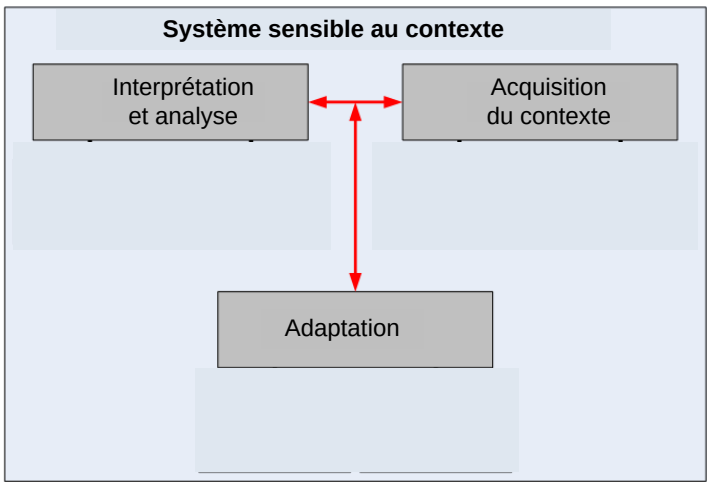


Figure II.6: Représentation des fonctionnalités d'un système sensible au contexte.

5.1 Acquisition du contexte

L 'acquisition du contexte représente le processus de collecte des données à partir de l'environnement qui entoure le système. Pour un processus métier le contexte d'exécution est constitué d'un ensemble de valeurs de variables appelées ≪variables de contexte ≫. Elles sont de plusieurs catégories. Lorsque les valeurs de ces variables changent le processus métier doit s'adapter. Les valeurs collectées sont sauvegardées dans un dépositaire de contexte pour être utilisées ultérieurement par le système. L'acquisition de ces valeurs se fait par l'utilisation des techniques de process mining.

5.2 Interprétation et analyse du contexte

L 'analyse du contexte est un processus qui contrôle continuellement les données collectées. Ce contrôle permet de filtrer les observations et les interprétations du contexte pour détecter les situations pertinentes dans le but d'adapter le système en conséquence ou de notifier les applications sensibles à ces situations pertinentes. Le développement du processus d'analyse nécessite une interaction avec le processus de collecte du contexte via le dépositaire de contexte pour vérifier les valeurs du contexte et détecter leurs changements pertinents. Grâce aux résultats obtenus par cette analyse, des décisions sont prises par le processus d'adaptation afin d'effectuer des changements sur le comportement de l'application selon les nouvelles situations pertinentes.

5.3 Adaptation aux changements de contexte

Une adaptation est une modification d'un système, en réponse à un changement dans son contexte, avec l'objectif que le système résultant soit mieux à même de réaliser sa fonction dans le nouveau contexte. En général, on peut décomposer le processus d'adaptation quelque-soit son type en deux étapes successives :

- Prise de décision concernant les opérations d'adaptation à effectuer,
- Application des actions d'adaptation en utilisant un mécanisme d'adaptation fourni par le système.

Les conditions qui engendrent l'exécution de chaque étape nous permettent de distinguer deux natures d'adaptation : l'adaptation de nature réactive et l'adaptation de nature proactive. Si la prise de décision d'adaptation et son exécution sont conditionnées par la détection d'une situation pertinente, on dit que c'est une adaptation réactive. Alors que si la décision d'adaptation est prise au moment de la détection de la situation pertinente alors que son exécution

est conditionnée par un autre évènement, on dit que c'est une adaptation proactive.Nous nous intéressons à un type d'adaptation proactive qui consiste à prendre la décision d'adaptation avant la détection d'une situation pertinente, simplement en utilisant l'historique (fichier log) des variations du contexte et un mécanisme d'apprentissage (process mining).

Le développement du mécanisme d'adaptation est difficile à mettre en oeuvre. Mais plusieurs techniques peuvent être utilisées pour faciliter cette tâche comme la réflexivité, la programmation par aspect, le paradigme composant conteneur et le moteur d'inférence.

Un moteur d'inférence est un programme qui effectue des déductions logiques d'un système à partir d'une base de connaissance et d'une base de règles. Les règles sont utilisées pour manipuler les connaissances et aboutir à des conclusions. Dans les systèmes sensibles au contexte, la base de connaissances peut être représentée par les informations de contexte observées ou interprétées. Par conséquent, les règles manipulent les informations de contexte pour détecter des situations pertinentes. Lors de la détection d'une situation pertinente, le moteur d'inférence exécute les actions d'adaptation appropriées.

6 Spécification des règles métier

La dimension comportementale d'un processus métier représente le flux de contrôle des éléments à exécuter dans un processus. En effet, les flux de contrôle sont les dépendances entre diverses activités. Ce sont les relations logiques qui contrôlent l'acheminement et l'ordre d'exécution des activités. Ces relations sont la formalisation des décisions à prendre et cela en tenant compte du contexte du processus et d'un certain nombre de contraintes. Ce sont aussi les règles qui permettent de contraindre, contrôler et influencer l'aspect du métier du processus. Ces règles sont appelées les règles métier ou règles de gestion.

Dans ce cas, la logique d'un processus métier peut être décrite, d'une manière déclarative, par un ensemble de règles. En effet, lorsqu'un événement se produit, un moteur évalue l'ensemble des règles, pour exécuter les activités métier décrites dans la partie d'une action en vérifiant certaines conditions. L'exécution de l'action d'une règle peut provoquer un événement qui active une ou plusieurs règles. Ainsi ce déclenchement permet d'exécuter les activités métier en suivant la logique du processus.

Dans un environnement ouvert et qui est en perpétuel changement, définir les règles métier préalablement semble une tâche difficile. Un système sensible au contexte permet aux règles métier d'être adaptatives dans le respect du contexte perçu.

Dans notre système le contexte est considéré comme un ensemble de règles sur lesquelles

≪l'Administrateur≫ s'appuie pour exécuter telle ou telle entité. Elles doivent refléter le contexte afin de permettre de sélectionner l'entité la plus adaptable à l'environnement à cet instant.

Par conséquent, dans notre système, nous avons une représentation initiale du processus métier à base de règles. Ce modèle basé règles est représenté par un des languages de modélisation pour formaliser la définition des règles.

Le choix d'un modèle de règles dépend de la catégorie des règles métier. Ils existent cinq catégories de règle (d'intégrité, de dérivation, de production, de transformation et de réaction). Dans le cadre de ce travail nous considérons que nos règles sont des règles de réaction car elles se déclenchent par des occurrences d'événements et qui exigent une satisfaction de conditions pour exécuter des actions. Exemple : à la réception d'une commande, si les matières premières sont disponibles alors lancer la production.

Les règles de réaction sont les plus adaptées pour décrire la logique du processus par un ensemble de règles. Elles sont supportées par le formalisme ECA[8] [Boukhebouze 2010].

D'une manière générale, une règle ECA se formalise par :

ON	<**E**vent>
IF	<**C**ondition>
DO	<**A**ction>

L'évènement détermine quand une règle doit être évaluée. La condition est un prédicat dont dépend l'exécution de l'action. L'action spécifie le code à exécuter si la condition est vraie.

La sémantique attachée à une règle est la suivante : lorsqu'un événement se produit, la partie condition est vérifiée. Si la condition est satisfaite, alors la partie action est exécutée en tenant compte des attributs associés à la règle.

En d'autres termes, la partie Évènement renseigne sur *Quand* une activité se termine. La partie Condition permet de connaître *Quel* chemin prendre selon les valeurs des variables par le workflow. La partie Action permet de connaître la prochaine activité qui sera exécutée.

Exemple :

$$WhenEndOf(A)Thenif(a == true)Start.Activity(B)if(b == true)Start.Activity(C)$$

D'un autre coté la décomposition de la structure d'un processus métier en règles métier est une procedure basée sur l'identification des patrons workflow [Bernal 2010]. Dans ce contexte, YAWL présente une grande expressivité des patrons workflow que nous détaillerons dans les prochains chapitres.

[8]ECA pour Event Condition Action

Utilisation du framework ProM

L'un des objectifs de ce travail est de découvrir et d'extraire certaines des règles métier non définies vu le changement que subissent les processus métier. Nous exploitons les techniques du process mining pour extraire ces règles métier. Cette démarche consiste en la selection et l'application des techniques du process mining et l'implémenter sur l'infrastructure du framework ProM (voir chapitreI section5.4). La démarche comprend la découverte des règles métier et leur représentation en ECA.

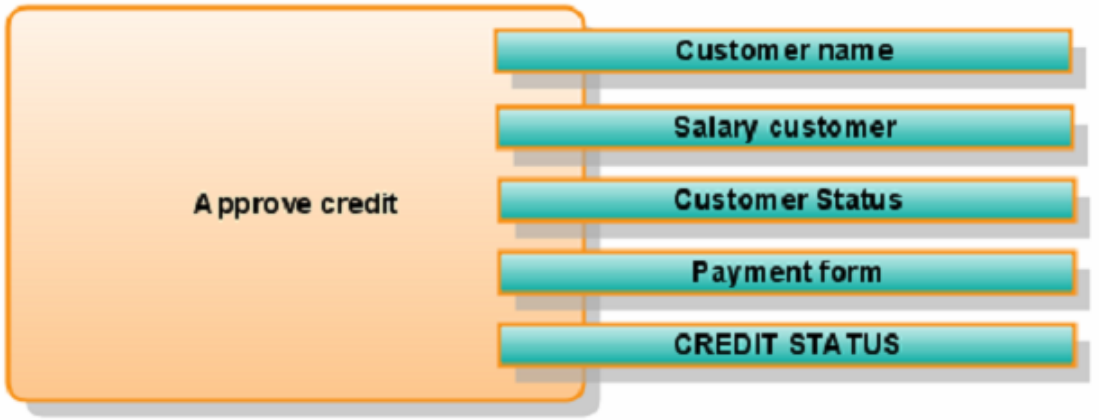


Figure II.7: Exemple de variable de contexte d'une actitvité selon[Crerie 2009].

Notons que dans les travaux de [Rozinat 2006] des points de décision dans un modèle de processus métier ont été exhibés et considérés comme des règles métier. Cependant, cette analyse ne concerne que les activités qui ont plus d'un successeur dans le flux d'activité . Dans ce travail nous considérons chaque activité individuellement avec ses variables de contexte (Voir figureII.7). Un travail de découverte de règles métier basé sur les techniques du process mining à des fins de classification et de representation de règles dans les legacy systems a été proposé dans [Crerie 2009].

7 Conclusion

Nous avons proposé dans ce chapitre notre approche de manière globale pour l'amélioration de la flexibilité par spécification partielle.

En effet, nous avons présenté d'abord la taxonomie de la flexibilité des processus métier en présentant une évaluation des produits existants afin de motiver notre choix sur ce type de flexibilité.

II.7 Conclusion

Nous avons défini par la suite les deux vues de notre système permettant une flexibilité émergente de la coordination d'entités englobant une activité (vue locale) dirigées par un administrateur (vue globale) en interagissant avec l'environnement(contexte).

Nous avons également exposé notre représentation du processus métier en un ensemble d'entités en faisant le parallèle avec l'approche "composant".

Nous avons ensuite défini la démarche à suivre pour la réalisation de système sensible au contexte pour démontrer la nécessité d'approcher la notion de règles métier dans la représentation de notre contexte d'exécution par des variables de contexte.

Le prochain chapitre présente notre architecture détaillée. Nous décrirons les concepts derrière chaque fonctionnalité de notre système à savoir l'acquisition du contexte, la représentation du processus en entités indépendantes et la supervision de l'administrateur.

Chapitre III

Une architecture basée artifact : exploitation du contexte d'exécution

1 Introduction

Depuis "toujours", en Génie Logiciel comme dans toutes les ingénieries, afin de réduire la complexité et améliorer la réutilisation, le produit à construire est divisé en parties construites indépendamment et sont ensuite assemblées. Notre approche fait de même, en proposant "simplement" que les parties à construire et à assembler soient des activités et non des programmes.

En effet, la composition est toujours inhérente à la phase de décomposition précédente. Cette décomposition permet de diviser un système en entités appelées dans [Tam 2008] "grain de composition". La nature des grains dépend de la vision de modularisation de chaque approche. Les approches définissent ce "grain de composition" en fonction :

- de la frontière qui sépare naturellement les objets de la réalité (modularisation orientée-objet),
- des préoccupations (modularisation orientée-aspect),
- des fonctionnalités fournies (modularisation orientée-modulaire, -composant, -service),
- de la vision des acteurs qui participent dans le système (modularisation orientée-vue) etc.

Dans ce chapitre, nous nous intéressons à la définition de notre vision de la modularisation dans l'approche présentée dans le chapitre précèdent. Le développement de système basé sur la décomposition et le contexte d'exécution nécessite que certaines propriétés soient vérifiées par ce système.

Cette problématique nous a conduits à concevoir et implémenter une architecture basée artifact pour améliorer la flexibilité en exploitant le contexte d'exécution.

Ce chapitre est donc dédié à la présentation de cette architecture. Nous commençons la description de notre architecture par la motivation de l'utilisation du concept artifact. Ensuite,

nous présentons les structures internes des différents modules impliqués, ainsi que leurs rôles dans notre approche.

2 Les artifacts d'exécution

Dans le domaine de composition chaque approche formalise les grains par ses propres notions de modularisation (e.g, module, classe, composant, service etc.). Dans une approche, il peut exister plusieurs vocabulaires pour la même notion. Cependant, selon[Tam 2008] un grain de composition doit satisfaire les caractéristiques suivantes :

- **Granularité :** La tendance actuelle est d'utiliser des grains plus gros pour pouvoir manipuler le système au niveau d'abstraction élevé avec moins d'entités à manipuler.
- **Grande cohésion :** C'est la propriété mesurant le degré de proximité fonctionnelle des opérations exposées par l'unité. Les grains cohésifs peuvent se focaliser rigoureusement sur l'implantation d'une logique métier. Les grains fortement cohésifs favorisent la compréhension, la réutilisabilité, la maintenance de système.
- **Couplage faible :** C'est la propriété mesurant le degré d'attachement de deux unités.
- **Avoir une interface**.
- **Avoir un haut niveau d'abstraction :** L'abstraction consiste à retenir uniquement les informations pertinentes dans un but particulier en ignorant les détails moins pertinents. Les unités n'exposent pas tous leurs détails mais seulement les parties qu'elles veulent publier. Dans la plupart des cas, le grain de composition, correspond aux éléments publiés dans leurs interfaces.

2.1 Besoin d'une nouvelle entité

Dans le cadre de ce travail, nous voulons que le grain de composition soit l'activité du processus métier. De plus, pour les raisons citées précédemment et celles citées dans le chapitre I section 4.2 nous ne pouvons considérer l'activité comme composant ou module tel qu'il a été utilisé dans le domaine de l'ingénierie des logiciels. Il faut donc utiliser une entité qui assure :

- L'abstraction.
- La compréhensibilité.
- La réutilisation.
- La facilité d'adaptation et d'évolution.
- La facilité de la localisation du changement.

La question à laquelle il faut répondre maintenant est : ≪ *sous quelle forme doit-on faire apparaître cette activité du processus métier ?* ≫

2.2 Le concept d'artifact

Il est apparu pendant l'étude que les travaux de Dinont portant sur une problématique différente, s'adaptent bien à notre situation. [Dinont 2007] [Dinont 2006].

Ces travaux portaient au départ sur la coordination entre agents. Ils se sont aperçus qu'il y avait un gain du point de vue conceptuel et du point de vue génie logiciel à externaliser toute la mécanique qui sert à la coordination entre les agents. Ils ont ainsi introduit la notion d'artifact de coordination [Omicini 2004] et l'ont placée comme un autre type d'entité de premier plan à côté des agents. Ils ont ensuite cherché à étendre le concept à d'autres utilisations que la coordination et à identifier les modifications que l'utilisation des artifacts apporte à la modélisation et à la programmation des systèmes multi-agents [Ricci 2005] [Viroli 2005].

Nous décrivons dans la suite le concept d'artifact[1] tel que nous l'utilisons dans nos systèmes. Il y a quelques différences par rapport à la définition initialement introduite par l'équipe italienne dans le cadre de ses travaux sur la coordination.

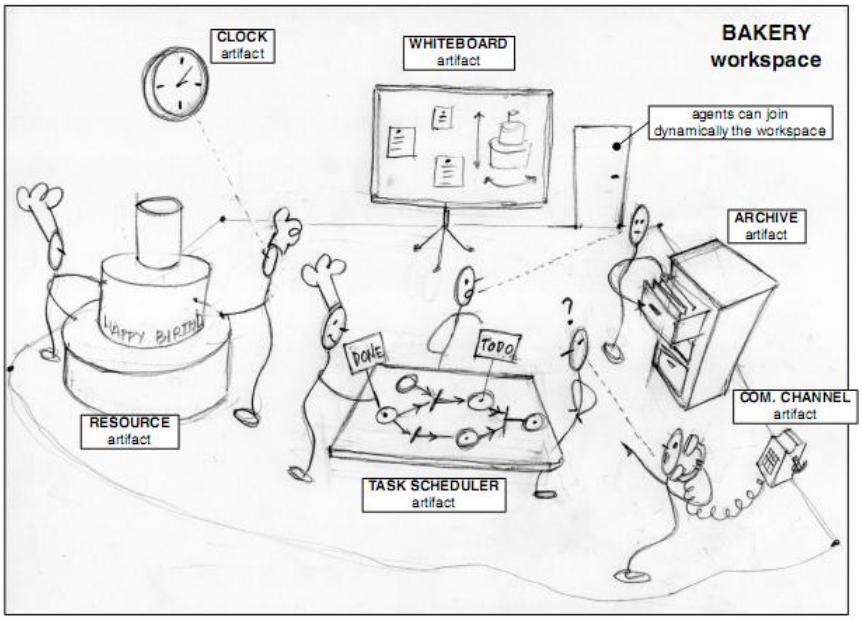


Figure III.1: Représentation métaphorique du concept d'artifact

[1]Nous conservons ici l'orthographe ≪artifact≫ pour éviter la confusion avec l'acception usuelle du mot artefact de la langue française : phénomène d'origine humaine.

Définition

Omicini remarque dans [Omicini 2004] que l'utilisation d'outils a toujours accompagné l'évolution de l'espèce humaine. Il est même certain aujourd'hui que le développement de l'intelligence comme caractéristique de l'espèce humaine est très fortement lié à la disponibilité et au développement des outils (Voir figureIII.1).

Si l'on transpose cela aux processus métier et leurs environnements d'exécution nous pouvons dès lors faire l'hypothèse que placer la notion d'activité comme type d'entité de premier plan au sein des systèmes de gestion des processus métier ne peut être que bénéfique. En effet, si un BPMS vu comme un système dirigé ou orienté par des buts, un artifact est une entité qu'un moteur utilise pour atteindre ses buts. Un artifact persiste au sein du système indépendamment du cycle de vie des instances de processus métier. De manière plus précise, un artifact est défini par le quadruplet :

$A =< F, Att, UI, M >$ où :

- F : Sa fonction qui est la description du ou des services rendus par l'artifact. Elle peut être utilisée pour rechercher et découvrir dynamiquement des artifacts.
- Att : Un ensemble d'attributs. Ce sont des variables internes que le module qui utilise l'artifact (administrateur) exhibe à l'extérieur.
- UI (User Interface) : Interface d'usage. Elle indique la nature des échanges entre l'administrateur et les artifacts. Ceux-ci sont dissymétriques puisque l'administrateur peut exécuter des actions et les artifacts peuvent fournir des perceptions de la terminaison d'une action. L'administrateur qui décide d'exécuter une action oblige l'artifact à l'exécuter immédiatement.
- M : Un ensemble d'instructions opératoires. C'est la description (formelle) de la manière dont le composant administrateur de notre architecture peut utiliser l'artifact. Cela constitue le mode d'emploi de l'artifact.

La figure III.2 montre comment on représente un artifact et ses caractéristiques associées.

A nos yeux, la caractéristique la plus importante d'un artifact réside dans ses instructions opératoires. Celles-ci permettent de définir de manière formelle le mode d'emploi de l'artifact. Ainsi, le comportement de l'artifact est prévisible. L'artifact s'engage implicitement à respecter son mode d'emploi. Nous consacrons la section 2.4 à l'étude des instructions opératoires.

2.3 Utilisations possibles

Il existe un certain nombre d'utilisations possibles des artifacts. Remarquons qu'un artifact peut être abordé selon différents angles et donc appartenir à plusieurs catégories.

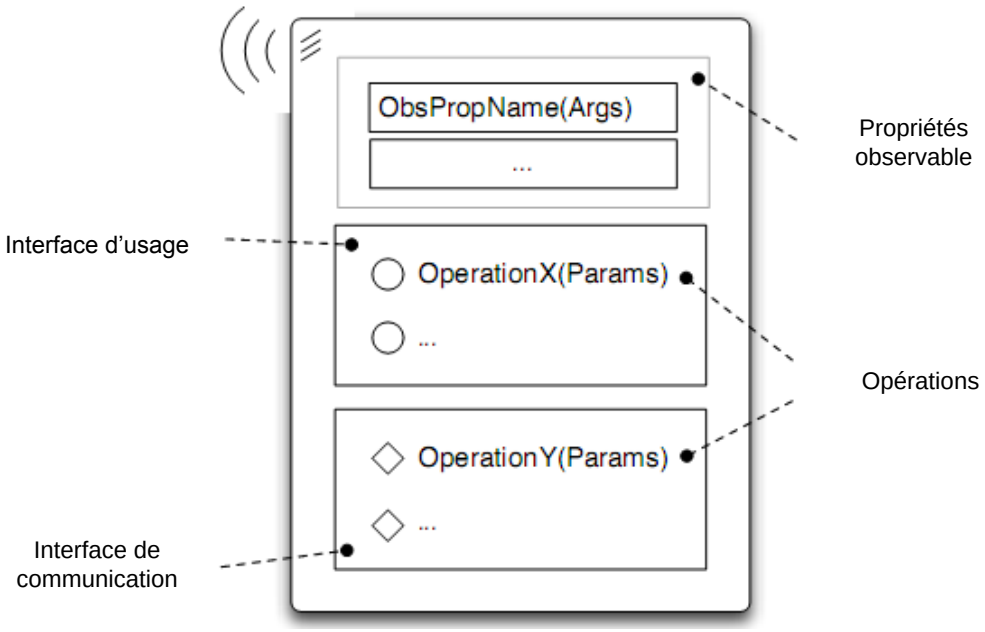


Figure III.2: Représentation abstraite d'un artifact

Les artifacts ont d'abord été utilisés pour la coordination entre les agents. Les artifacts de coordination peuvent être utilisés à différents niveaux d'abstraction pour gérer des fonctions de communication (blackboard, boîtes de messages, services de gestion d'événements), des fonctions de synchronisation (ordonnancement, sémaphores,...) ou des fonctions de coordination de haut niveau (systèmes d'enchères, d'infrastructures de phéromones,...).

Un artifact frontière est utilisé pour caractériser et contrôler la présence d'un agent dans un contexte organisationnel, en réifiant un contrat entre un agent et une organisation. Dans les environnements basés sur les rôles, un artifact frontière embarque le contrat pour les rôles que l'agent joue dans l'organisation. Il est fourni à l'agent quand celui-ci démarre une session de travail dans l'organisation. Il permet de contraindre ce que l'agent peut faire dans l'organisation. Il peut être vu comme une interface frontière entre l'agent et l'environnement.

Les artifacts de ressource sont des moyens de médiation d'accès aux ressources physiques ou pour donner une existence à une partie de l'environnement du SMA[2] correspondant à une ressource partageable. Ce type d'artifact est important pour apporter au niveau d'abstraction des agents toutes les entités physiques et de calcul qui peuvent être utilisées par les agents. Les artifacts de ressource remplacent avantageusement les agents wrappers couramment utilisés pour résoudre ce problème.

[2]Système Multi Agent

2.4 Instructions opératoires

Nous présentons dans cette section le langage formel qui est utilisé pour décrire les instructions opératoires. Il s'agit d'un formalisme à base d'algèbre de processus introduit dans [Viroli 2004].

Dans les algèbres de processus, ces derniers sont vus comme des entités ≪ autonomes ≫ donc asynchrones, et l'on se donne les moyens d'organiser la cohérence du système de processus.

Algèbre de processus

Un processus p est conçu abstraitement comme un objet qui accomplit certaines actions a et ce faisant, se reconfigure en un autre processus p'. Cette relation est dénotée par :

$$p \xrightarrow{a} p'$$

Un processus se termine avec succès (état $\surd$) s'il n'est plus constitué que d'une action atomique et que celle-ci est exécutée :

$$p \xrightarrow{a} \surd$$

La plupart des algèbres de processus contiennent des opérateurs de base pour construire des processus finis, des opérateurs de communication pour exprimer la concurrence et une certaine notion de récursion pour représenter les comportements infinis.

Les algèbres de processus permettent un raisonnement formel sur les processus et les données et peuvent être utilisées pour faire de la vérification des systèmes. On pourra se reporter à [Fokkink 2000] pour plus d'informations théoriques et pratiques sur les algèbres de processus.

Expression des instructions opératoires ≪ Mode d'emploi ≫

Nous allons maintenant décrire le formalisme à base d'algèbre de processus proposé dans [Viroli 2004] pour décrire les instructions opératoires des artifacts.

Soit Δ l'ensemble des actes d'interaction possibles et $\mathcal{I}$ l'ensemble des instructions opératoires. Les définitions d'un acte d'interaction $\delta \in \Delta$ et d'une instruction opératoire $I \in \mathcal{I}$ sont :

$$\delta ::= \alpha \mid \pi$$

$$I ::= 0 \mid !\alpha \mid \underline{\alpha} \mid ?\pi \mid I; I \mid I + I \mid (I \parallel I) \mid \mathcal{D}(t_1, ..., t_n)$$

Un acte d'interaction δ peut être une action α ou une perception π. Une instruction I peut être une instruction atomique : le comportement nul (0), le déclenchement de l'exécution

de l'action α ($!\alpha$), une action débutée mais non encore terminée ($\underline{\alpha}$) et la perception π de la terminaison d'une action ($?\pi$). Une instruction peut également être structurée grâce à différents opérateurs : $\ll;\gg$ pour la composition séquentielle de deux instructions, $\ll + \gg$ pour le choix et $\ll\|\gg$ pour la composition parallèle. Enfin, pour permettre la récursion, une instruction peut être l'invocation $\mathcal{D}(t_1, ..., t_n)$ d'une autre instruction opératoire avec $\mathcal{D}$ le nom de l'instruction opératoire à appeler et $t_1, ..., t_n$ la liste de ses paramètres.

Une sémantique opérationnelle est donnée à ce langage résumé dans [Dinont 2007].

Nous pouvons citer comme exemple d'instructions opératoires le cas général suivant :

$$((!a; ?fin\_a) + (!b; ?fin\_b)) \parallel (!c; ?fin\_c)$$

Ou celui d'un lave_vaisselle :

$$\begin{aligned} lave\_vaisselle\_simple := \\ & ((!laver; ?fin\_lavage); (!laver; ?fin\_lavage); ...; \\ & (!laver; ?fin\_lavage); (!remettre\_du\_sel; ?fin\_remplissage); \\ & lave\_vaisselle\_simple \end{aligned}$$

Concernant le mode d'emploi d'un lave-vaisselle, il serait intéressant de pouvoir y spécifier qu'$\ll$ il faut remplir le bac à sel tous les dix lavages $\gg$. Cela peut être réalisé en utilisant un état de l'instruction opératoire par lavage : après le premier lavage, on se retrouve dans un état identique au précédent, mais correspondant au second lavage. Après le dixième lavage, il n'est plus possible de remettre en route un nouveau lavage. Il faut remplir le bac à sel et ensuite on revient à l'état initial pour un nouveau cycle de dix lavages.

2.5 Discussion

Dans le but d'éclaircir la motivation de l'utilisation d'artifact au lieu de composant ou objet nous présentons une brève comparaison entre ces différentes entités de système. Les artifacts se placent à un niveau d'abstraction différent de celui des objets. Les composants se placent au même niveau d'abstraction que les artifacts. Leur structuration se rapproche très fortement de celle des artifacts. Un composant permet d'encapsuler des traitements. Il dispose d'une enveloppe (le conteneur) fournissant un certain nombre de services de base comme les possibilités de communication, de sécurité ou de persistance. On retrouve cette enveloppe dans les artifacts.

Les attributs représentent les propriétés configurables du composant. Ils se rapprochent des attributs définis pour les artifacts, mais leur sémantique est moins précise. Dans les composants,

ils pourraient ne pas exister et être remplacés par des méthodes appelées en mode synchrone pour lire ou modifier un attribut.

Ce qui différencie clairement un artifact d'un composant, ce sont les instructions opératoires qu'il exhibe. Les autres différences sont somme toute assez mineures. En définitive, et pour résumer, un artifact peut être considéré comme un composant disposant d'un mode d'emploi formel réifié que d'autres entités peuvent manipuler.

Dans notre travail, il s'agit d'introduire une spécialisation de la notion d'artifact. En effet, contrairement aux premiers travaux de l'équipe de Bologne sur les artifacts [Ricci 2005] [Viroli 2005] qui portaient principalement sur les possibilités de coordination entre agents au travers des artifacts, nous nous intéressons aux artifacts comme des entités encapsulant l'activité d'un processus métier disponible dans l'environnement et utilisé par un administrateur pour exécuter un processus métier d'une manière flexible.

Nous allons décrire les différents composants de notre architecture dans les sections suivantes.

3 Description architecturale

Généralement une architecture exprime la structure fondamentale du système à analyser et à concevoir. L'architecture définit un ensemble de composants fonctionnels, des sous-systèmes ou de modules décrits en termes de leurs comportements et interfaces. Elle définit aussi comment ces composants interagissent et comment ils doivent être inter-connectés pour accomplir correctement les buts du système. Ainsi, une description architecturale est principalement requise pour la spécification de la structure du système.

En principe, le terme composant inclut les éléments fonctionnels qui peuvent être des modules ou des artifacts. Tel est le cas dans notre travail. En prenant en considération ces définitions, nous allons présenter dans les sections suivantes de ce chapitre les spécifications des différents composants ainsi que les concepts liés à leur fonctionnement.

3.1 Aperçu de l'architecture proposée

L'architecture de flexibilité que nous proposons dans ce travail est basée sur un framework constitué de composants pour la sensibilité de contexte et de raisonnement et d'une bibliothèque de variables de contexte.

La figure III.3 représente cette architecture qui se compose des parties suivantes :

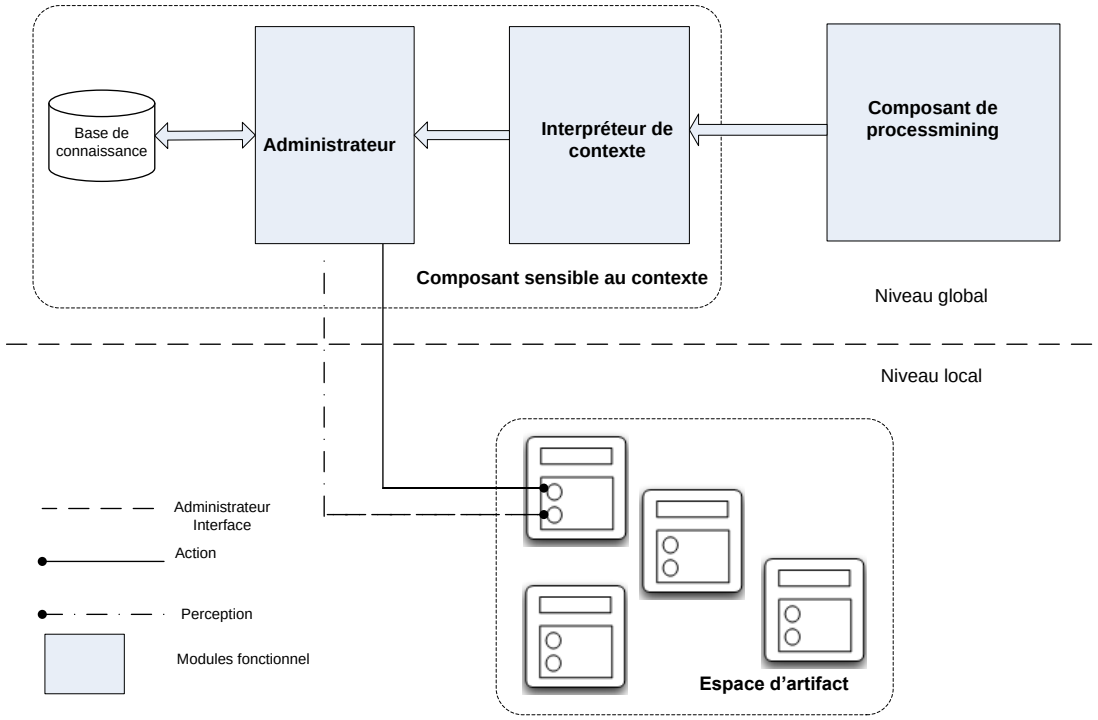


Figure III.3: Architecture générale du système proposé

– L'espace d'artifact où figurent toutes les activités d'un processus métier encapsulées chacune par un artifact dont l'utilisation est décrite par un manuel.
– Un composant sensible au contexte, suite à l'interprétation du contexte, décide de l'action à entreprendre sur l'espace artifact en interagissant avec celui-ci par une action ou une perception(voire figure III.3.
– Un composant de process mining chargé de capturer le contexte pour en extraire des variables de contexte sauvegardées dans une bibliothèque.

Comme mentionné précédemment, nous considérons le modèle du processus métier comme un ensemble de parties s'exécutant ensemble pour atteindre un objectif commun [Zerari 2010b]. Chaque partie s'occupe d'une activité du processus métier global. La composition de ces différentes parties s'effectue au moment de l'exécution (réalisation). Le choix de la partie à exécuter dépend des conditions d'exécution et du contexte d'exécution du processus. En effet, contrairement à la méthode YAWL qui existe et qui traite ce genre de flexibilité basé sur les worklet, nous nous intéressons au contexte d'exécution de ces processus métier pour décider quel module sera exécuté.

Nos hypothèses de travail sont les suivantes :
– Un processus est un ensemble d'activités ;

– La flexibilité est de pouvoir exécuter une activité selon son contexte d'exécution ;
– Le "Quoi" (l'activité) ne change pas c'est le "quand" (sous quelles conditions) qui change.

Notation

Nous définissons un processus métier (BP[3])de manière formelle inspirée des travaux de [HAMRI 2010] sur l'annotation de modèle de processus métier que nous avons adapté à nos besoins.

BP est un quintuplet défini comme suit :

$$BP =< AV, AR, AE, AS, AT, RS >$$

Où :

AV est un ensemble d'activités composant le processus et AV_i est une activité ;
AR est un ensemble de rôles qui interagissent dans le processus et AR_i est un rôle ;
AE est un ensemble de paramètres d'entrées (inputs) et AE_i représente les inputs de l'activité AV_i ;
AS est un ensemble de paramètres de sorties (outputs) et AS_i représente les outputs de l'activité AV_i ;
AT définit les conditions de transition pour l'exécution du processus et AT_i dénote les conditions de transition de l'activité AV_i ;
RS est un ensemble de ressources qui sont invoquées pendant l'exécution du processus et RS_i dénote les ressources utilisées par l'activité AV_i ;

Une activité est une étape d'un processus qui peut être atomique (ou composite). Elle est réalisée par un rôle et exécutée via les conditions de transition. Une activité quelconque (AV_i) est alors décrite comme suit :

$$AV_i = (name, AR_i, AE_i, AS_i, RS_i, AT_i)$$

Considérons maintenant $\mathcal{A}$ l'ensemble d'activités, $\mathcal{A}^+$ est l'ensemble de toutes les séquences finies non vides de $\mathcal{A}$. Une trace $\mathcal{T}$ est un élément de $\mathcal{A}^+$. Un évènement log $\mathcal{L}$ est un ensemble multiple de traces $\mathcal{T}$ de $\mathcal{A}^+$.

Prenons l'exemple suivant :Soit $\mathcal{A} = \{a, b, c\}$ l'ensemble d'activités. $\mathcal{T} = abcabb$ une trace de longueur 6. $\mathcal{L} = \{aba, aba, abba, baca, acc, cac\}$ représente un événement log. Ces notions de trace et d'événement sont en rapport avec le domaine du process mining.

[3]Business Process

Par conséquent, pour considérer l'activité comme un artifact et donc un processus comme un ensemble d'artifacts, il nous faut traiter certaines correspondances que nous allons présenter dans la section suivante.

3.2 Structure de l'artifact

Si nous considérons la définition de l'artifact présentée dans la figure III.2section 2 et celle que nous venons de présenter dans la section précédente une activité est :

$$AV_i = A$$

Où :

$$\begin{cases} F & Le\ traitement\ de\ l'activité. \\ AE_i\ et\ AS_i & Att \\ RS_i & Une\ ressource\ qu'utilise\ un\ artifact \\ AT_i & M \end{cases}$$

Dans le contexte d'annotation de processus métier une condition de transition est représentée par des expressions pour contraindre les inputs et les outputs qui sont définis comme des paramètres de l'activité (pré-conditions et post-conditions). Les conditions de transition permettent de spécifier le contrôle de flux qui s'exprime à travers les connecteurs AND, OR, XOR qui sont des opérateurs logiques de base.

Dans notre contexte de travail ces contraintes sur les paramètres et et le contrôle de flux entre activités sont à la charge de l'administrateur qui exécute le bon artifact. Comment cette interaction doit se faire est décrite dans le manuel (Instructions opératoires).

3.3 Instructions opératoires de l'artifact métier

Nous savons qu'une instance d'activité passe par plusieurs états. L'état d'activité est une représentation des conditions internes définissant l'état d'une instance d'activité à un moment particulier. L'état de transition est un mouvement d'un état interne de l'activité vers un autre à l'intérieur d'un processus métier. La transition traduit un changement dans l'état de l'instance de processus. Une instance d'activité peut passer par les états suivants(voir figure III.4) :

- Initialisée : l'instance de processus vient d'être initialisée mais elle peut ne pas être active.
- Active : l'instance de processus a démarré. Une ou plusieurs activités peuvent être actives.

- Suspendue : l'instance de processus est suspendue et aucune autre activité n'est démarrée tant que l'instance n'est pas reprise.
- Complétée : l'instance de processus est complètement réalisée(avec succès).
- Terminée : l'exécution du processus a été stoppée à cause d'un incident(Abandonnée ou interrompue).

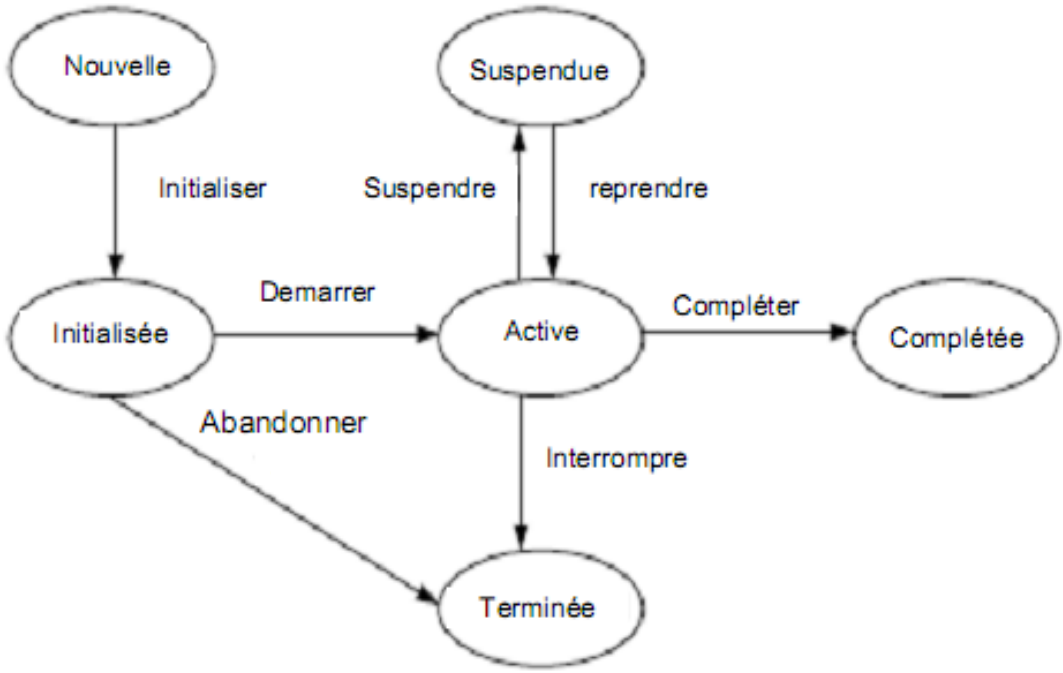


Figure III.4: Diagramme d'état transition représentant les différents états d'une instance.

Un artifact d'activité est un artifact chargé d'encapsuler l'exécution d'une activité métier, au sens où nous l'avons décrit précédemment.

Un artifact d'activité dispose au minimum des actions suivantes :

- `schedule` qui permet d'initialiser l'artifact.
- `start` qui permet de démarrer une activité.
- `suspend` qui permet de suspendre une activité en cours.
- `resume` qui permet de redémarrer l'activité là où elle s'était arrêtée lors de l'appel à `suspend`.
- `abort` qui permet d'interrompre l'exécution de l'artifact qui est en cours d'exécution.
- `withdraw` qui permet d'abandonner l'exécution de l'artifact avant qu'il ne soit actif.

Les perceptions minimales sont :

- `Completed(result)` : Cette perception est reçue lorsque le travail de l'activité initié par

`start` a été terminé avec succès. `result` est unifié avec le résultat donné par l'activité lancée.
- `suspended` : Cette perception est émise au moment où l'artifact est suspendu.
- `resumed` : Cette perception est émise au moment où l'artifact reprend l'exécution.
- `withdrawn` : Cette perception est émise au moment où l'artifact est abandonné.
- `aborted` : Cette perception est émise au moment où l'artifact est interrompu.
- `scheduled` : Cette perception est émise au moment où l'artifact est initié.
- `error(info_error)` : Cette perception permet de transmettre les erreurs d'exécution de l'activité à l'administrateur.

Les actions et perceptions précédentes constituent une base minimale qu'il est possible d'étendre en fonction du type particulier d'artifact d'activité auquel on a affaire. Par exemple, pour un artifact qui encapsule une activité de test sanguin , `test_blood` introduira l'action `treat_blood`.

4 Roles et comportements des différents composants

Nous allons décrire, dans ce qui suit, les différents modules qui composent notre architecture en détails. La définition des différents modules composant notre architecture implique que nous devons également décrire l'interaction et la participation de ces composants dans le processus de l'amélioration de la flexibilité.

4.1 Composant process mining

Ce composant est chargé de découvrir les règles métier en utilisant les principes du process mining. Il s'agit d'identifier les règles métier et de les représenter sous la forme ECA dans une bibliothèque de règles(voir chapitre II section 6). Pour pouvoir extraire ou découvrir ces règles nous avons besoin de certaines informations dans les event log représentées sous le format de Mxml. L'architecture de la figure III.5 représente ce composant. Les fichiers log du système sont capturés et transformés en format Mxml. Une technique est appliquée pour extraire les règles métier dans la bibliothèque.

Ce composant possède une entrée et une sortie pour atteindre un objectif. Pour ce faire, ce composant poursuit un certain nombre de phases que nous allons détailler dans ce qui suit :

Phase de pré-analyse

Pour l'extraction des connaissances (règles métier dans notre cas) certaines hypothèses sur les traces de processus métier (fichier log) doivent être respectées. En effet, un fichier log doit

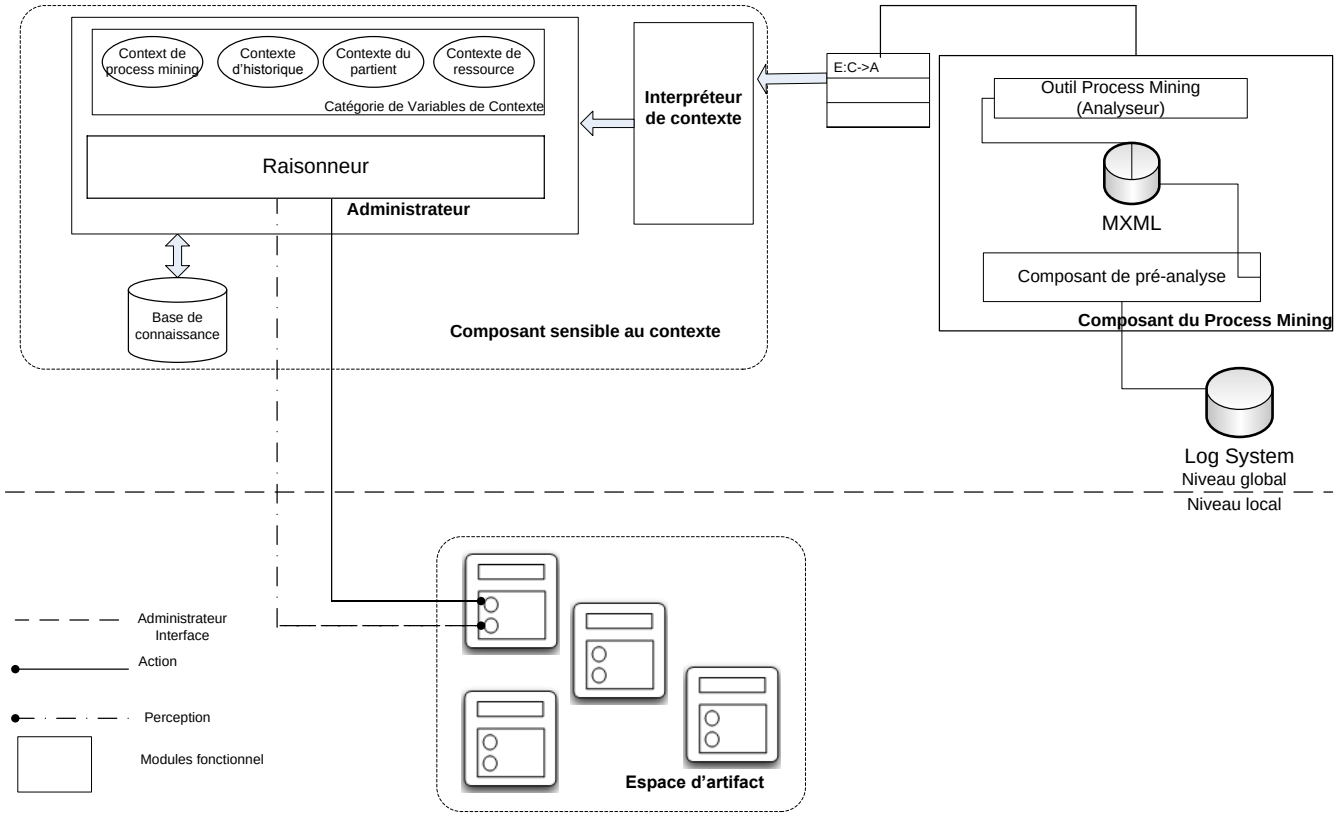


Figure III.5: Un framework sensible au contexte pour la flexibilité des processus métier.

contenir les informations suivantes :

- Chaque événement dans le fichier log exprime une seule activité.
- Chaque événement exprime également un seul cas d'exécution (instance de processus)
- Chaque événement doit contenir un seul rôle qui exécute l'instance de processus.

Dans notre travail, nous voulons avoir le maximum d'informations possibles à l'exécution qui reflètent le contexte d'exécution. Pour cela, nous exploitons dans la représentation des fichiers log la notion d'attribut. Dans notre cas ils seront des variables de contexte. Un exemple de telles variables a été présenté dans le chapitre précèdent section 6.

Par conséquent, la phase de pré-analyse s'assure que les fichiers log du système en entrée respectent ces conditions dans sa phase d'exécution (voir figure III.6). Ensuite ils seront convertis en un format unique qui est le Mxml (voir chapitre I section 5.4).

Le résultat d'une telle phase est la génération d'un fichier pré-traité et converti en un format Mxml.

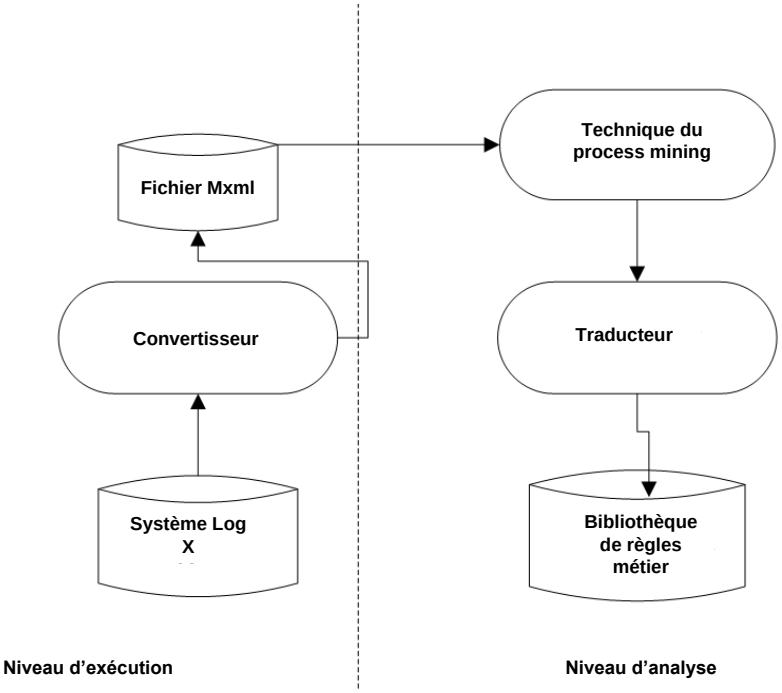


Figure III.6: Les phases de pré-analyse et d'analyse

Phase d'analyse

A ce stade de la découverte une technique d'extraction est appliquée aux fichiers Mxml pour la découverte de règles métier et des valeurs des variables de contexte (voir figureIII.6). Les étapes suivantes doivent être suivies :

1. **Regroupent d'informations relatives à la même activité** : le fichier Mxml en entrée possède une structure qui permet de lier plusieurs instances d'activité entre elles. La première étape est donc de rassembler toutes les occurrences apparues en rapport avec la même activité.

2. **Analyse des variables de contexte** : pour chaque activité sont identifiées les variables de contexte avec leurs valeurs possibles.

3. **Création de la bibliothèque de variables de contexte** : Après le regroupement des informations relatives à une activité et après identification des variables de contexte et de leur valeur un fichier listant l'ensemble de ces variables ainsi que leurs valeurs est créé.

4. **Extraction des règles métier** : les règles métiers sont obtenues en appliquant l'arbre de décision défini par une des techniques du process mining [Rozinat 2006].

Bibliothèque des variables de contexte

Dans l'architecture du système que nous proposons, se trouve une bibliothèque des variables de contexte. Cette bibliothèque est gérée par le composant du process mining et l'administrateur.

4.2 Composant sensible au contexte d'exécution

Il s'agit du composant qui prend les décisions dans notre architecture. En effet, c'est à ce niveau que le choix de l'activité à exécuter est pris en se basant sur la bibliothèque de règles et sur les variables de contexte. Ce composant contient d'autres modules afin de prendre en charge les différentes tâches de prise de décision de raisonnement et de mise à jour de la base de connaissances.

Interpréteur de contexte

Ce module est chargé d'interpréter le contexte en utilisant les valeurs des variables de contexte dans la bibliothèque et les classer selon des catégories. Ceci permet de relever les contextes pertinents selon des facteurs de contexte.

D'après les travaux de [Saidani 2008] et ceux en rapport avec le data mining [Vajirkar 2003] nous avons déduit les catégories suivantes de variables de contexte(facteur de contexte) :

- **Variables de contexte «ressource» :** Elles concernent les propriétés matérielles et humaines des ressources (matériels, acteurs). Par exemple : la performance d'un acteur, son experience ou matériel comme le manque de ressources.
- **Variables de contexte «temps» :**Elles reflètent des caractéristiques liées au temps. Elles peuvent décrire un jour férié ou une urgence.
- **Variables de contexte «données» :** Elles décrivent le domaine spécifique au contexte. Par exemple les informations pertinentes sur l'utilisateur d'une application peut varier selon le domaine.

Il s'agit pour notre étude de cas de l'appliquer aux systèmes de santé vu leur grand besoin de flexibilité et d'adaptabilité dans leurs domaines d'exécution. Nous y reviendrons dans le chapitre suivant. Néanmoins nous aimerions décrire les variables de contexte relatives à ce genre d'application dans le tableauIII.1

Variables de contexte ressource	Dans le processus de diagnostic l'expérience de l'acteur est importante dans certains cas. Dans la phase de test sanguin par exemple, en cas de panne de la ressource (laboratoire) il est souhaitable de s'adapter en donnant le groupe sanguin universel.
Variables de contexte temps	Dans le cas d'une urgence il serait souhaitable de passer certaines activités comme l'admission du patient par exemple.
Variables de contexte données	Il s'agit dans ce cas d'avoir des variables de contexte à propos du patient. Toutes les informations relatives au patient sont importantes afin d'établir un diagnostic. Exemple : Age,adresse, sexe, antécédents familiaux, historiques ...etc.

Tableau III.1: Catégories de variables de contexte

Communication avec les artifacts

Nous avons vu dans la section 3.3 une base minimale des actions et perceptions d'un artifact d'activité. Cependant aucune interaction d'aucune sorte n'a été donnée (instruction opératoire). Dans cette section nous voulons donner un exemple d'instruction opératoire et comment l'administrateur communique (agit et perçoit) avec un artifact toujours dans le domaine des systèmes de santé.

Prenons l'exemple de test sanguin appelé `Test_blood`. Cet artifact pourrait avoir le mode d'emploi suivant :

`Test_blood` := !`start` ; (((?`completed(RESULT)` + ?`error(ERROR_INFO)`) + (!`abort` ; ?`aborted`)) || `suspend_restart`) ; `Test_blood`

Tout d'abord, l'administrateur n'a qu'une action possible : démarrer le test. Ensuite, il sait qu'il recevra soit le résultat du test soit un rapport d'erreur. A tout moment, il a la possibilité d'arrêter le test, d'effectuer des cycles de mise en pause et de redémarrage selon l'un des deux modes proposés. Quand un test est terminé, on reboucle pour pouvoir en relancer un nouveau.

Administrateur

Nous avons précédemment évoqué ce module (chapitre II) en précisant que c'était ce module qui agit et perçoit les artifacts selon le contexte d'exécution qui lui parvient.

En effet, nous pouvons considérer ce module comme notre système de connaissances. Il raisonne et met à jour la base de connaissances en inférant sur les règles métier et le contexte qu'il possède. L'intérêt des informations de contexte est de permettre à ce module de prendre la décision de quelle action (artifact) doit-il lancer et de comprendre les perceptions qui lui parviennent.

Car les instructions opératoires, données seules, ne sont d'aucune utilité à l'administrateur. Il faut lui donner des informations supplémentaires lui permettant de décider des actions qu'il va lancer sur les artifacts et de comprendre les perceptions qu'il va recevoir en retour.

Prenons l'exemple d'un mode d'emploi rudimentaire pour un `lave_linge`. On peut lancer un cycle de lavage et percevoir sa terminaison puis recommencer autant de fois que l'on veut. L'instruction opératoire correspondante est la suivante :

`laver_linge` := !`lancer_cycle` ; ?`fin_cycle` ; `laver_linge`

Nous voulons pouvoir indiquer à l'administrateur quand il pourra décider d'exécuter l'action `lancer_cycle`. Cela s'exprime ainsi :

Action	Pré-condition	Perception	Effet
Lancer_cycle	B linge_sale	fin_cycle	¬ linge_sale

L'administrateur décidera d'exécuter l'action `lancer_cycle` s'il croit (opérateur `B`) que son linge est sale. Les effets liés à la perception `fin_cycle` permettent de mettre à jour la base de connaissances pour que l'administrateur sache maintenant que son linge est propre.

Nous présentons dans ce qui suit le cycle de raisonnement de notre architecture quand tous les composants interviennent :

1. Percevoir l'environnement par le module process mining et interpréteur de contexte.
2. Mettre à jour la base des faits (croyances)
3. Sélectionner un évènement
4. Retirer tous les plans possibles.
5. Déterminer les plans applicables.
6. Sélectionner un plan applicable.
7. Sélectionner un artifact.
8. Exécuter.

5 Conclusion

Le développement de système flexible et adaptable pour les processus métier nécessite une grande maîtrise de leurs contextes d'exécution dans la phase de modélisation ou d'exécution. De ce fait, l'implémentation des systèmes sensibles au contexte est complexe, de même pour l'implémentation des composants qui les exploitent.

III.5 Conclusion

Nous avons développé dans ce chapitre, une architecture basée-artifact permettant l'amélioration de la flexibilité des processus en étant sensibles à leurs contextes d'exécution.

Notre architecture offre plusieurs avantages. C'est ainsi qu'elle permet :

- La réutilisation de fragments de processus (artifact) dans divers contextes d'exécution.
- L'expressivité par la richesse du langage de description des instructions opératoires
- L'évolutivité de la bibliothèque des règles métier car elle permet la découverte de nouvelles règles métier selon les fichiers logs.
- La vérification car les artifacts utilisent des langages formels.

Dans le chapitre suivant, nous illustrons l'application de notre approche avec une étude de cas. Nous décrirons tout d'abord l'utilisation des différentes étapes et modèles utilisés dans notre travail puis l'architecture proposée.

Chapitre IV

Etude de cas et implémentation

1 Introduction

Nous venons de spécifier tout au long des précédents chapitres les concepts, les modules et les formalismes permettant de concevoir une approche basée artifact et sensible au contexte pour l'amélioration de la flexibilité des processus métier. Dans le présent chapitre, nous présentons quelques aspects d'implémentation.

En effet, dans une première partie, nous présentons notre étude de cas sur les systèmes médicaux [1] en motivant les raisons d'un tel choix. Dans cette partie nous décrivons également l'utilisation du framework ProM pour la définition d'un nouveau plug-in pour l'extraction de règles métier. Dans une seconde partie, nous précisons la plate-forme d'implémentation JASON et comment les concepts de base utilisés dans notre architecture système peuvent être implémentés en utilisant cette plate-forme.

2 Énoncé de l'étude de cas

Au coeur de toute organisation médicale (Health-care) existe un ensemble de processus métier critiques. Ces processus métier peuvent concerner le patient, la gestion de l'assurance ou encore des fonctionnalités administratives. Ces organisations médicales souffrent de manque d'efficacité. Leur SIs[2] peinent à résoudre leurs problèmes car ils ne sont pas intégrés et ne sont pas centrés sur leurs processus. Dans ce contexte, la gestion des processus métier tente d'apporter des solutions.

[1] Terme en Anglais utilisé dans la suite est Health-care systems
[2] Systèmes d'Information

2.1 Caractéristiques des systèmes «Health-care»

Les patients se dirigent vers différentes organisations médicales pour recevoir des services tels que les diagnostics et les traitements de leurs problèmes de santé. L'objectif des systèmes health-care est de fournir des services efficaces à leurs patients en termes de qualité, de respect des délais et de fonctionnalité. Par conséquent, ces systèmes ont les caractéristiques suivantes :

1. *Dynamiques, complexes et multi fonctionnels* : les processus métier dans de tels systèmes concernent un large volume de données d'un grand nombre de patients et de personnels. Il est clair également qu'ils ne concernent pas seulement la santé du patient mais utilisent des procedures d'autres disciplines telles que la gestion ou la finance. En plus, le traitement d'un patient peut être dynamique et peut devenir complexe. En effet, un patient peut être admis dans un hôpital pour le traitement d'une maladie X mais durant le diagnostic ou durant le traitement il peut developer une autre maladie Y. Par conséquent le processus de traitement ne peut être vu comme un processus séquentiel. C'est un processus de traitement de différentes maladies avec des conditions concurrentielles ayant recours aux personnels de différentes disciplines.
2. *Automatiques, collaboratifs et avec coordination :* La majorité des activités de chaque processus ne peut être que partiellement automatisée. Les décisions sont prises par les personnes. Il y a également un besoin de collaboration et de coordination entre les différentes activités.
3. *Données non appropriées pour la gestion :* La gestion des hôpitaux et des patients souffre du surplus de données. Elles sont redondantes, non uniformes et prêtent à confusion. Ce qui rend difficile le contrôle, la disponibilité et l'apprentissage sur les informations des systèmes "health-care".
4. *Intervention de plusieurs applications :* Plusieurs départements utilisent les mêmes données, ce qui accroît le problème de redondance. Par exemple, les détails d'un patient peuvent être stockés simultanément par une application utilisée en cardiologie et en radiologie.
5. *Actions ad hoc et changements de processus :* Dans un hôpital, les actions sont influencées par des évènements médicaux tels qu'une nouvelle procedure administrative, une nouvelle technique médicale ou des informations sur l'environnement de travail. Ces événements forcent le changement, l'extension ou l'arret des procedures habituelles. De telles actions non structurées et ad hoc sont difficiles à modéliser et à automatiser. Il devient nécessaire et critique de prendre en charge ces changements dans l'élaboration des systèmes "health-care".

Il parait évident selon ces caractéristiques que les processus métier (Traitement, diagnostic) dans le domaine médical ne sont pas faciles à automatiser et à comprendre car ils sont dynamiques, ad-hoc, non structurés et multi-disciplinaires.

Par conséquent, nous avons choisi ce type de processus métier pour mettre en oeuvre notre architecture afin de voir l'apport de notre approche sur de tels processus métier.

2.2 Présentation de l'étude de cas : Les systèmes "Health-care"

Tel que nous l'avons mentionné précédemment, nous optons pour une étude de cas qui concerne les systèmes "Health-care". Une architecture de tels systèmes est représentée dans la figure IV.1 [Gupta 2007]. Ceci dit, nous nous intéressons seulement aux processus internes de l'organisation.

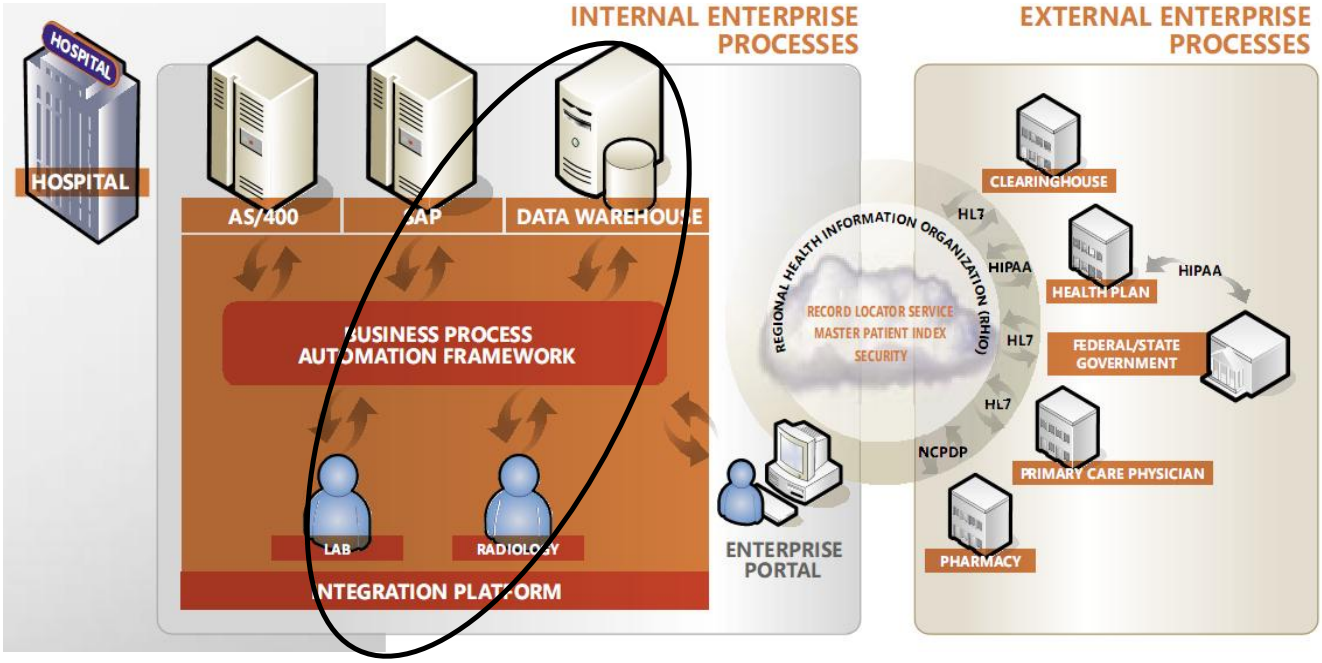


Figure IV.1: Architecture d'un système "health-care".

La mise en oeuvre de cette architecture commence dans la phase d'exécution comme nous l'avons mentionné dans les chapitres précédents. Nous considérons notre étude de cas à partir de traces laissées par l'exécution de notre processus métier (Fichier log). En effet, l'étude de cas concerne les données et les traitements d'un patient dans un hôpital. La base de données contient des informations enregistrées sur le patient lors de son admission pour recevoir des

soins. La structure de ce processus est représentée par le diagramme de la figure IV.2.

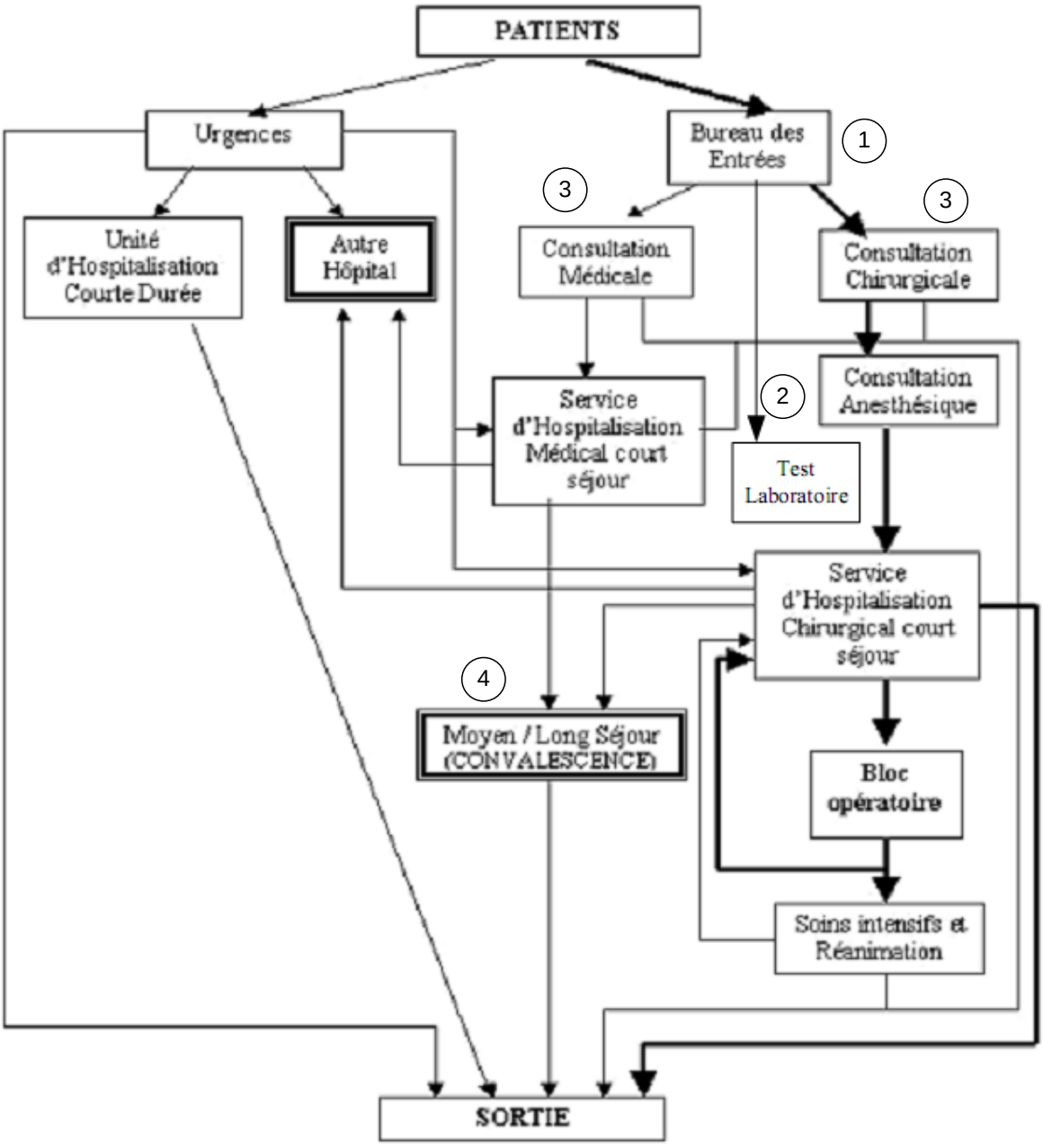


Figure IV.2: Structure de l'étude de cas "Admission d'un patient"

Le diagramme représente dans chacune de ses activités les informations suivantes stockées dans la base de données :

1. Enregistrement des données personnelles du patient ainsi que son historique clinique lors de son admission au bureau des entrées.
2. Des tests de laboratoire sont effectués ainsi que certaines données tels que la tension artérielle, taux de glycémie,...Etc.
3. Les données sur les traitements que le patient a subis durant son hospitalisation.
4. Après la sortie du patient un traitement post-opératoire peut être prescrit.

Ces données sont stockées dans une base de donnés développée à l'aide de MS-Access que nous décrirons ultérieurement.

2.3 Le langage YAWL : Yet Another Workflow Language

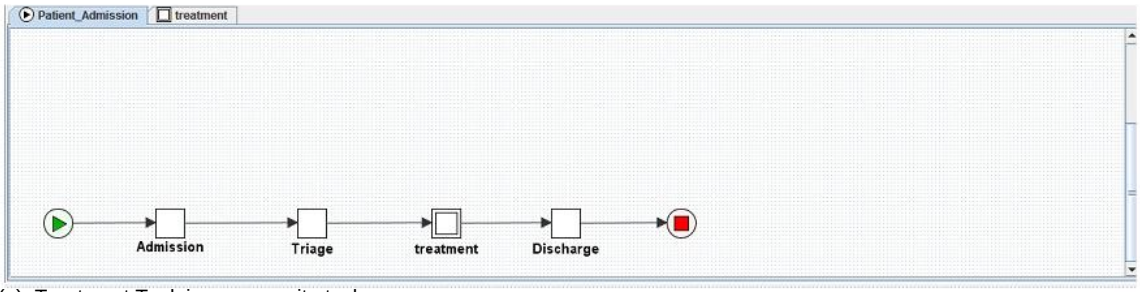
Il est vrai que nous considérons notre processus métier dans sa phase d'exécution et à partir des traces laissées. Cependant, nous aimerions présenter un modèle de notre processus avec le langage YAWL à des fins d'illustration de l'apport de la représentation des processus métier par les artifacts tel que nous l'avons décrit dans [Zerari 2011]. Notre choix s'est porté sur ce langage car :

- YAWL est un langage de représentation des processus métier étendant la syntaxe des réseaux de Pétri. Donc, il hérite de ce langage les mécanismes de vérification et de bon fonctionnement des processus.
- YAWL est surtout, dans notre contexte, le seul langage (système) à traiter la flexibilité par spécification partielle des processus métier (Voir chapitre II section 2.2).

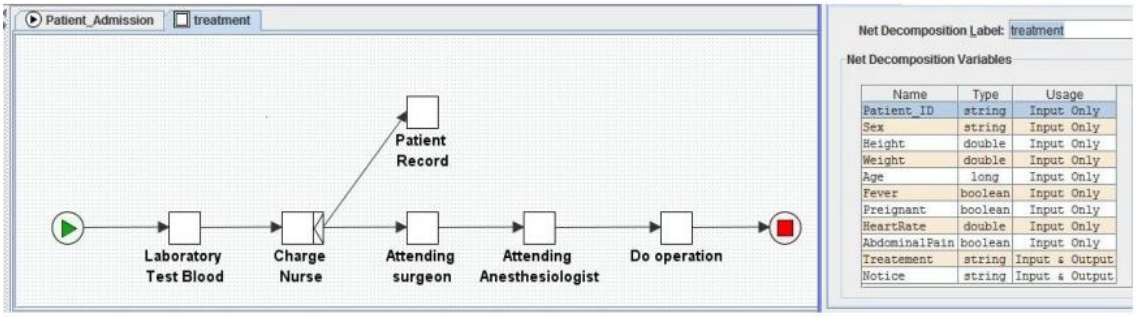
Pour atteindre la flexibilité par specification partielle YAWL utilise la notion de "Worklet" qui définit des sous processus métier appartenant à un processus global. Ces "Worklets" sont stockés dans un répertoire. Le choix de l'exécution du "Worklet" approprié se fait à l'exécution.

2.4 La modélisation du processus métier avec YAWL

La figure IV.3 représente notre étude de cas "Admission d'un patient" modélisé avec YAWL. La figure IV.3 (a) est le modèle global du processus métier (spécifié partiellement) composé d'une tâche composite "Traitement" . Cette tâche est définie comme un ensemble de "Worklets" stocké dans un répertoire. A l'exécution un choix est fait quant au "Worklet" exécuté (Figure IV.3 (b)). Cette méthode exige la définition de toutes les specifications des sous processus (Worklet) au niveau de la modélisation.



(a): Treatment Task is a composite task



(b): Treatment Net with Contextual Variables.

Figure IV.3: Modélisation par YAWL du processus métier "Admission d'un patient"

3 Implémentation de l'architecture proposée

Un système à base d'artifacts inspirés des SMA sensible au contexte exige qu'ils soient déployés dans un environnement qui supporte leur exécution.

Comme nous l'avons décrit dans le chapitre III, plusieurs modules interviennent dans notre architecture. Il y a notamment le module "process mining", les artifacts et l'administrateur. Dans la suite nous présenterons les outils et plate-forme pour la mise en oeuvre de ces différentes parties.

Le déroulement global de notre application implémentant notre architecture est présenté dans la figure IV.4. Il est composé des étapes suivantes :

1. La collecte de traces : Dans notre cas les différentes données et traces des systèmes Healthcare sont sous la forme de base de données MS-Access. Nous avons considéré les données disponibles dans les travaux de [Gupta 2007](Voir 1 de la figure IV.4).
2. Génération du fichier Event logs : Pour appliquer ou developer des plug in pour l'extraction d'informations des traces d'exécution, il faut que ces traces soient dans le format MXML utilisant le framework $ProM_{import}$(Voir chapitre 1 section 5.4)
3. Extraction de règles métier : Pour le développement d'un plug-in pour l'extraction des règles métier à partir du fichier MXML nous utiliserons le framework ProM. Cette appli-

cation est inspirée des techniques de data mining disponibles dans la bibliothèque weka (Voir 3 de la figure IV.4).

4. Implémentation du raisonneur : A ce niveau d'implémentation et de mise en oeuvre de notre travail, nous pouvons évoquer le support technique de notre administrateur en le considérant comme agent exploitant les artifacts selon le contexte recueilli sous forme de règles métier de l'étape précédente. Pour ce faire nous utilisons la plate-forme Jason que nous présenterons dans la section suivante.

5. Implémentation des artifacts : Les artifacts qui encapsulent nos activités métier, sont inspirés comme nous l'avons mentionné dans les chapitres précédents des SMA. Donc, pour leur implémentation nous utiliserons également la plate-forme Jason.

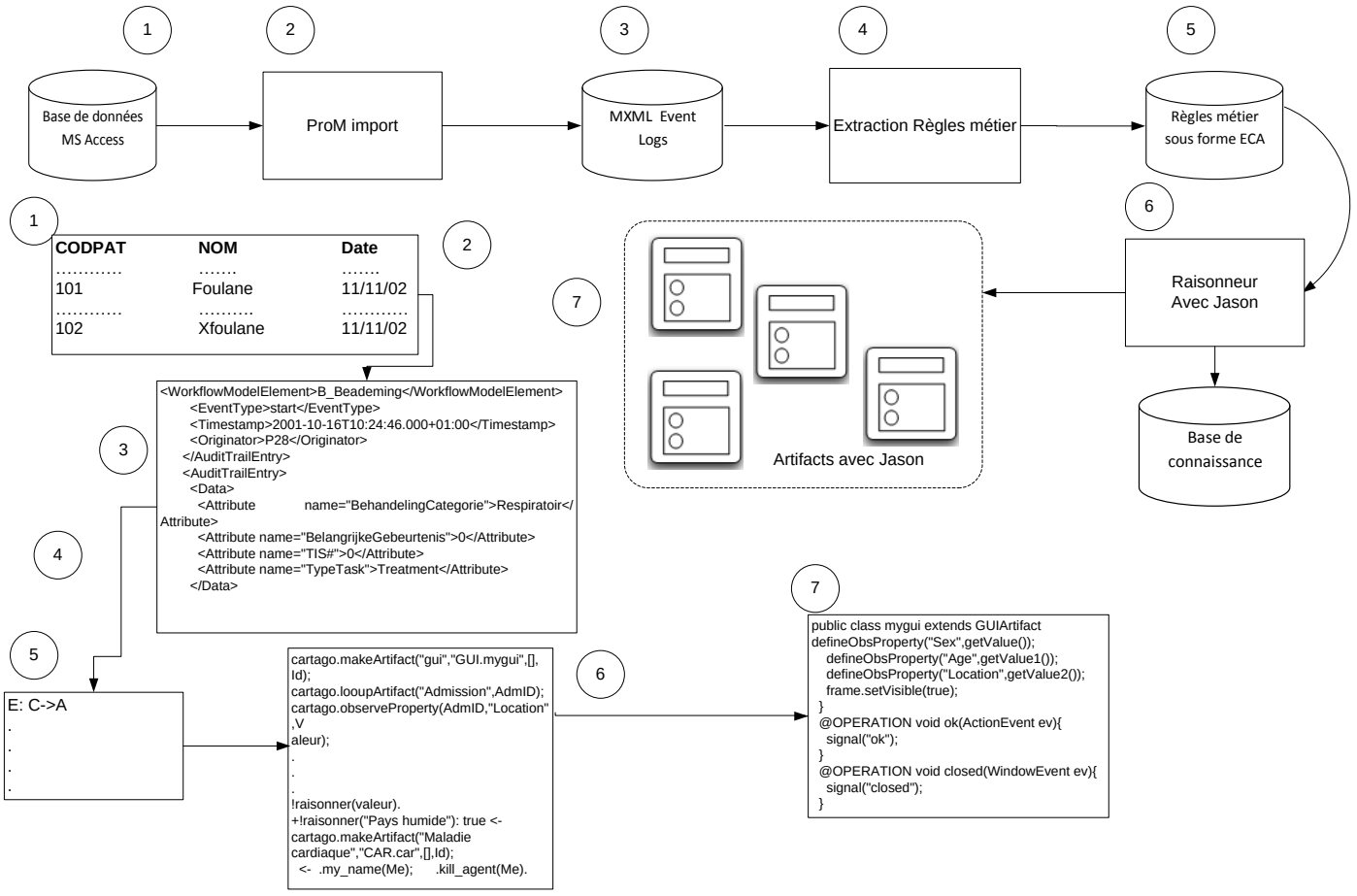


Figure IV.4: Scénario d'exécution du système

3.1 Collecte de traces : Génération des fichiers log

Les données de notre étude de cas sont organisées dans des tables au sein d'une base de données en MS-Access. Dans cette section nous présentons les tables les plus importantes de cette base de données. Les données dans ces tables concernent les informations générales du patient, les complications, les résultats des bilans et les traitements. Chaque table est constituée du nom du champs, type de données et description. Pour transformer une base de données en fichier MXML cette dernière doit contenir certains champs et exécuter certaines macros d'après le tutorial de [Pro] disponible à l'adresse : http://promimport.proessmining.org.

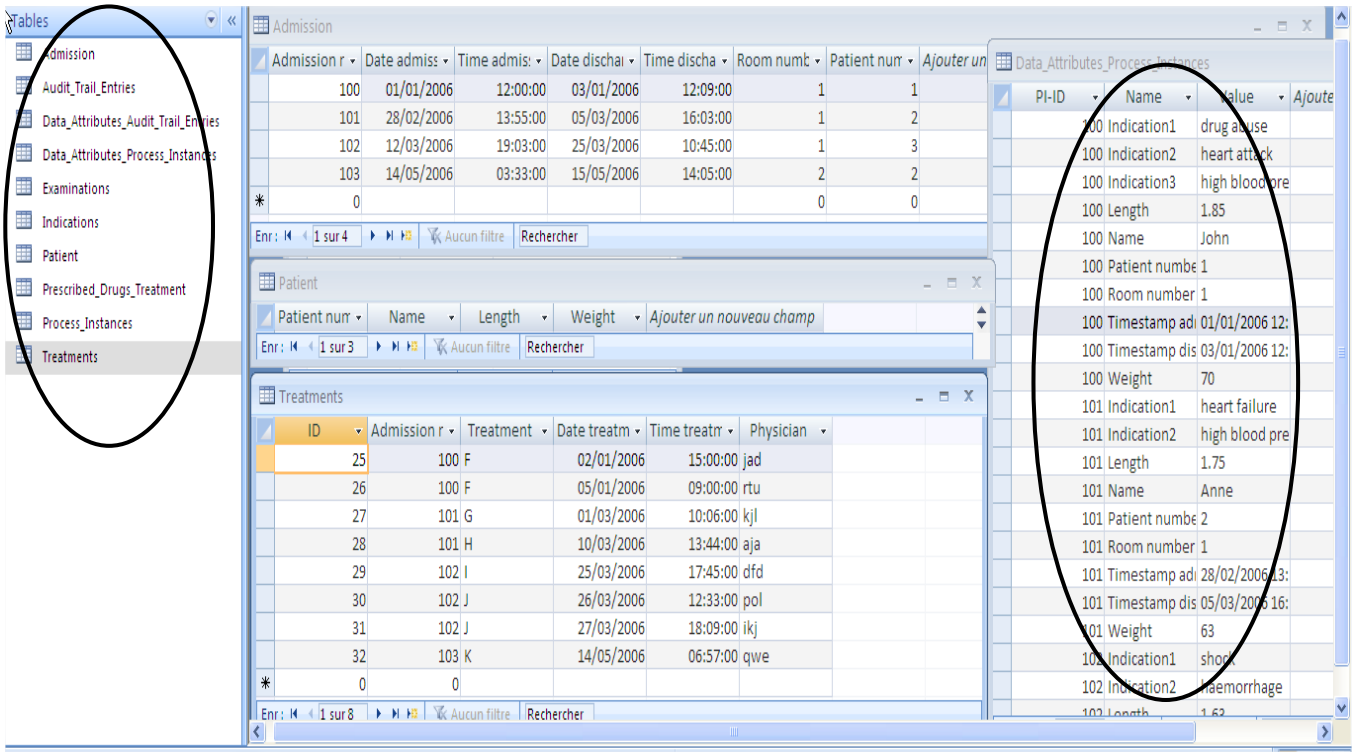


Figure IV.5: L'ensemble des tables de la base de données

La figure IV.5 représente une vue générale des différentes tables constituant la base de données Health-care. Elle représente l'ensemble des tables constituant la base de données traditionnelle ainsi que celles crées par les macros pour les besoins de conversion de MS-Access en MXML.

Cette base de données sera fournie comme entrée au framework $ProM_{import}$ en spécifiant le nom de la base de données ainsi que les tables de transition nécessaires à la conversion.

Le fichier obtenu en sortie est un fichier MXML incluant les attributs, les instances et les événements des différentes exécutions du processus métier.

Nous montrons le code d'un tel fichier dans la figure IV.6. Pour chaque log nous représentons une instance de processus comprenant ses attributs, l'entrée *audit trail* et d'autres informations. La structure de ce fichier log est conforme à la DTD présentée dans le chapitre I à la section 5.4.

```
<ProcessInstance id="100">
<Data>
<Attribute name="Indication1">drug abuse</Attribute>
<Attribute name="Indication2">heart attack</Attribute>
<Attribute name="Indication3">high blood pressure</Attribute>
<Attribute name="Length">1.85</Attribute>
<Attribute name="Name">John</Attribute>
<Attribute name="Patient_number">1</Attribute>
<Attribute name="Room_number">1</Attribute>
<Attribute name="Timestamp_admission">01/01/2006 12:00:00
</Attribute>
<Attribute name="Timestamp_discharge">03/01/2006 12:09:00
</Attribute>
<Attribute name="Weight">70</Attribute>
</Data>
<AuditTrailEntry>
<Data>
<Attribute name="Important_examination">-1</Attribute>
<Attribute name="drug">12055</Attribute>
</Data>
<WorkflowModelElement>A</WorkflowModelElement>
<EventType>start</EventType>
<Timestamp>2006-01-01T15:00:00.000+01:00</Timestamp>
<Originator>jre</Originator>
</AuditTrailEntry>
.....
.....
.....
```

Figure IV.6: Fichier MXML obtenu après conversion

3.2 Extraction du contexte d'exécution

Le fichier MXML précédemment obtenu est l'entrée de notre plug-in pour l'extraction de règles métier. Pour ce faire nous utilisons le framework ProM6 qui dispose d'un certain nombre de plug-in implémentant diverses techniques de process mining ainsi que la possibilité d'en développer de nouveaux.

La vue statique du plug-in est présentée par un diagramme de classe décrit dans la figure IV.7.

Ce diagramme montre les différentes classes qui interviennent dans la structure du plug-in développé. Les relations entre ces classes reflètent leurs interactions futures. Il existe trois catégories de classes :

1. Classes du framework ProM (Voir figure IV.8 (a))

2. Classe du plug-in "extraction de règles métier" (voir figure IV.8 (b)).
3. Classes de la librairie Weka (voir figure IV.8 (c)).

Le résultat est un fichier qui contient des règles métier extraites des fichiers log. Ces règles sont sous la forme ECA (Voir Chapitre II section 6). Elles sont exploitées par le module interpréteur de contexte. Le raisonneur, dans ce cas, peut décider quel artifact sera exécuté. Il coordonne les tâches du processus métier. Un tel fichier ressemblera à la figure IV.9.

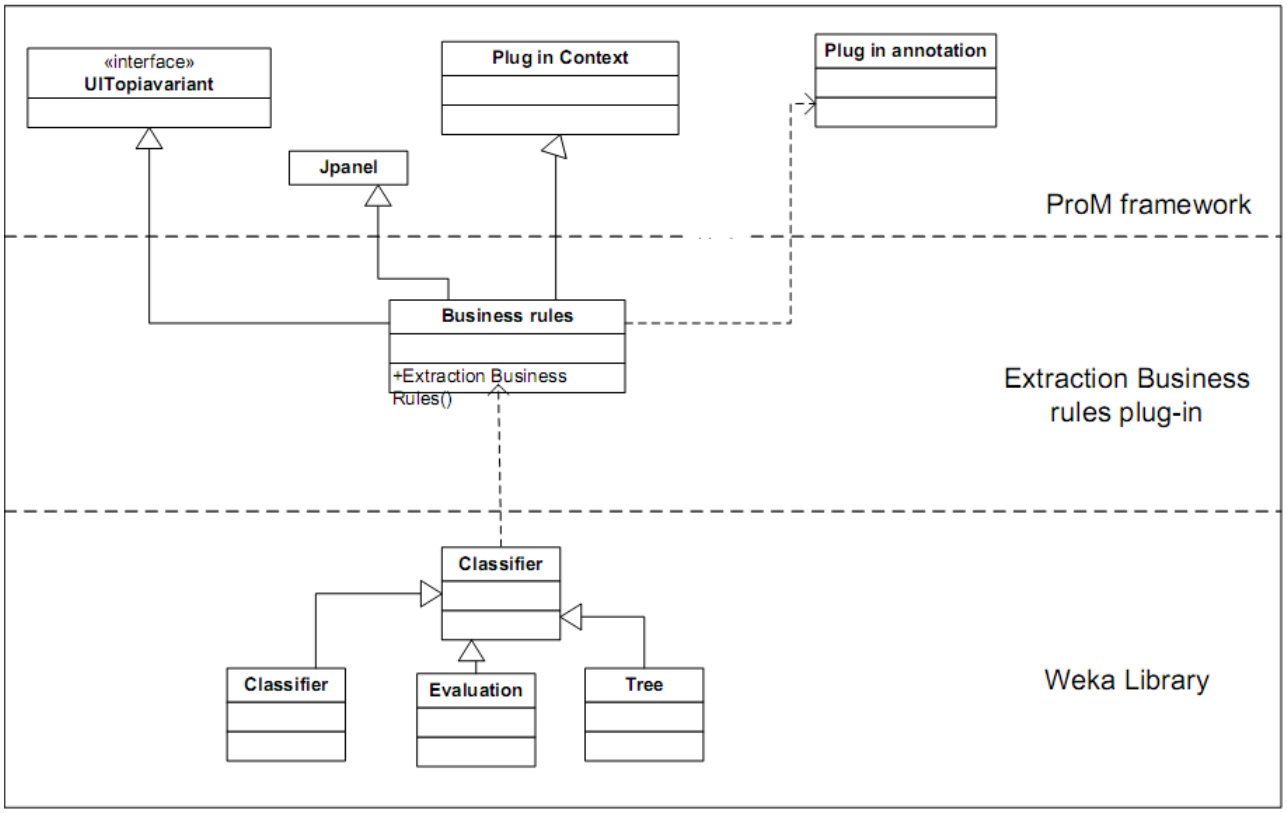


Figure IV.7: Diagramme de classe d'implémentation du plug-in Business rules extraction.

Ce langage est celui proposé comme extension du langage ECA dans les travaux de [Boukhebouze 2010].

3.3 Le langage Jason et le framework CArtAgO

Nous avons mentionné précédemment que la notion d'artifact est inspirée du domaine de la coordination dans les systèmes multi agents.

Par conséquent, nous préconisons que la construction de notre système, qui n'est pas un système à base d'agent, doit correspondre à un processus qui prend en compte le contexte

```
package org.processmining.plugins.gettingstarted;
import java.io.File;
                                                                    (a)
import org.deckfour.xes.factory.XFactoryRegistry;
import org.deckfour.xes.model.XAttributable;
import org.deckfour.xes.model.XLog;
import org.processmining.contexts.uitopia.annotations.UITopiaVariant;
import org.processmining.framework.plugin.PluginContext;
import org.processmining.framework.plugin.annotations.Plugin;

import weka.core.*;
import weka.classifiers.Classifier;
import weka.classifiers.Evaluation;
import weka.classifiers.trees.*;
import weka.gui.treevisualizer.PlaceNode2;
import weka.gui.treevisualizer.TreeVisualizer;                      (c)

public class BusinessRules {
       @Plugin(
            name = "Extraction Business Rules Plug-in",
            parameterLabels = {"log"},                              (b)
            returnLabels = { "Businessrules" },
            returnTypes = {File.class },
            userAccessible = true,
            help = "Extraction de r|gles m-tier ¢ partir d'un fichier MXML en entr-e."
    )


       @UITopiaVariant(
            affiliation = "Mentouri University",
            author = "Zerari",
            email = "Zerari.mounira"
    )
    public static File ExtractBusinessRules(PluginContext context, XLog log) {
……..
….
        DecisionTree(Database db, Vector<Attribute> candidatAttr, Vector<Integer> index,
Attribute goal, int code, int mode, double percent)
            {
                this.db = db;
                this.db.filterAttributes(index, goal);
                this.db.setTrain(mode, percent);
                this.db.train.setClassIndex(this.db.train.attribute(goal.name()).index());
                this.code = code;
                this.goal = goal;
                initTree();
                buildTree();

            }


       return w;

       }

}
```

Figure IV.8: Codification des différentes classes du plug-in Business rules extraction.

d'exécution (environnement) pour percevoir et agir sur les artifacts.

De point de vue de la mise en oeuvre, nous avons considéré le module "Raisonneur" comme un agent. Nous avons utilisé une plate-forme basée-agent pour implémenter cet agent. Nous avons également utilisé en collaboration avec cette plate-forme une infrastructure pour l'implé-

```
<xsd:element name="BusinessRules" type="tBusinessRules"/>
    <xsd:complexType name="tBusinessRules">
            <xsd:sequence>
                    <xsd:element ref="BusinessRule" maxOccurs="unbounded"/>
            </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="BusinessRule" type="tBusinessRule"/>
    <xsd:complexType name="tBusinessRule">
    <xsd:sequence>
                    <xsd:element ref="OnEvent"/>
                    <xsd:element ref="PreCondition"/>
                    <xsd:element ref="Action"/>
                    <xsd:element ref="CompensateAction"/>
                    <xsd:element ref="PostCondition"/>
                    <xsd:element ref="PostEvents"/>
                    <xsd:element ref="ErrorEvents"/>
            </xsd:sequence>
            <xsd:attribute name="name" type="xsd:ID" use="required"/>
            <xsd:attribute name="version" type="xsd:anySimpleType"/>
            <xsd:attribute name="Priority" type="xsd:anySimpleType"/>
    </xsd:complexType>
```

Figure IV.9: Le schéma XML de la partie règles ECAPE extension de ECA.

mentation des artifacts.

Le choix d'une telle plate-forme est basé sur plusieurs critères comme :

- La nature de nos objectifs (utilisation des artifacts) ;
- Le type de communication employée ;
- L'utilisation d'un langage de raisonnement.
- Le degré de standardisation établi dans cette plate-forme.

Cette plate-forme va nous permettre d'implémenter notre module "raisonneur" et nos artifacts et montrer les actions et les perceptions qui régissent la communication entre le "Raisonneur" et les artifacts.

La plate-forme Jason

La plate-forme Jason est une extension du langage AgentSpeak(L) pour la mise en pratique de la programmation agent. Ce langage est basé sur la logique de BDI. Il possède un interpréteur Java. Il est inspiré des systèmes procéduraux de raisonnement. Il est Open source et disponible à l'adresse http://jason.sourceforge.net.

Le langage AgentSpeak(L) interprété par la plate-forme Jason crée un agent en spécifiant les concepts suivants :

Les croyances : Il s'agit de l'ensemble d'informations que l'agent possède du monde qui l'entoure. Ce monde comprend l'etat de son environnement, les autres agents et lui même.

Objectifs : C'est ce que cherche à atteindre l'agent.

Plans : Il s'agit de l'ensemble d'actions (recettes) pour atteindre les objectifs.

Les intentions : C'est l'ensemble des plans exécutables.

Le framework CArtAgO

CArtAgO (Common ARTifact Infrastructure for AGents Open environment) est une infrastructure générique pour la conception et la création d'environnement pour les applications orientées agents utilisant les principes d'une nouvelle approche celle de A&A (Agents and Artifact) [Aricci 2008].

Afin de réduire la complexité de la conception des systèmes de logiciels, le meta-modèle A&A introduit une vue abstraite métaphorique de la théorie de l'activité humaine et la discipline de la connaissance distribuée .

En effet, dans cette optique, les agents sont analogues aux humains qui exécutent les activités et les artifacts analogues aux objets utilisés par ces humains. Dans notre contexte de travail, l'agent est analogue à un module preneur de décisions et les artifacts à des entités encapsulant les activités du processus métier.

CArtAgO fournit une infrastructure et une API pour programmer et exécuter des environnements basés artifact conformes au modèle A&A. Il est orthogonal à une architecture basée agent bien spécifique. Il doit être intégré à différentes plates-formes agents. Dans notre cas il sera intégré à la plate-forme Jason.

Le modèle A&A est basé sur le concept d'agent et d'artifact bien sur mais également sur celui d'espace de travail (workspace) qui regroupe les deux concepts précédents et définit leur environnement.

CArtAgO posséde les caractéristiques suivantes :

- Il se base sur la programmation Java pour la définition d'artifacts.
- IL offre un ensemble d'API pour que les agents puissent utiliser les artifacts.
- Il peut être intégré à un ensemble hétérogène de plate-formes agents.
- Les agents peuvent joindre plusieurs espaces de travail à un moment donné.
- Il est bien évidemment open source et disponible à `http://cartago.sourceforge.net`.

L'intégration de CArtAgO à la plate-forme Jason exige de nous le respect des considérations suivantes IV.10 :

- Actions : les actions externes d'un agent Jason correspondent aux opérations de l'artifact.

– Perceptions : les propriétés observables de l'artifact correspondent aux croyances de l'agent. Les signaux de l'artifact correspondent aux événements observables.
– Modèle de données : Le modèle de données de Jason est étendu pour manipuler des objets Java.

```
public class Myartifact extends Artifact {
  void init(){
  defineObsProperty("proprieté",0);
  }

  @OPERATION void inc(){
  ObsProperty prop = getObsProperty("propriété");
  prop.updateValue(prop.intValue() + );
     signal("tick");
}
}
```

Figure IV.10: La classe artifact

3.4 Utilisation de CArtAgO et Jason pour implémenter les artifacts et le raisonneur

Du point de vue implémentation,le module raisonneur et l'utilisation des artifacts pour exécuter les activités de notre processus métier "Admission d'un patient" correspondent à la déclaration d'un MAS (Multi Agent System) avec un environnement et un agent que nous avons appelé "raisonneur" (Voir figure IV.11).

```
MAS Admission_patient {
environment:
alice.c4jason.CEnvStandalone
agents:
agent0 raisonneur agentArchClass alice.c4jason.CAgentArch;
}
```

Figure IV.11: La définition du MAS avec le module raisonneur

Les artifacts de notre étude de cas présentés dans notre architecture sont dérivés de la classe Artifact de la plate-forme Jason. Chaque artifact correspond à une activité dans le processus métier. La figure IV.12 présente l'artifact "test_blood". La figure IV.12 représente également la

```
public class Test_blood extends Artifact {
  void init(){
  defineObsProperty("groupe_sanguin",O);
  defineObsProperty("taux_glicémie",0.8);
  }

  @OPERATION void test_sanguin(){
  ObsProperty prop = getObsProperty("groupe_sanguin");
  prop.updateValue(prop.string.Value() + );
     signal("tick");
}
@OPERATION void glicémie(){
  ObsProperty prop1 = getObsProperty("taux_glicémie");
  prop.updateValue(prop.int.Value() + );

}
}
```

Figure IV.12: La définition de l'artifact "test_blood"

définition de deux propriétés observables par le raisonneur. Nous avons également défini deux opérations disponibles sur l'artifact activé par le raisonneur.

3.5 Invocation des artifacts lors de la prise de décision

La création, l'utilisation et l'observation de tous les artifacts, déclarés à l'image de celui de la figure IV.12, est à la charge du "raisonneur" : Sa spécification est présentée dans la figure IV.13.

Le raisonneur agit sur l'artifact à travers les actions définies comme des opérations. Il observe également le comportement de l'artifact à travers les propriétés observables et met à jour sa base de croyances. Il utilise cet artifact selon le mode d'emploi. Par exemple, le raisonneur observe la propriété de groupe_sanguin. Si un certain délai est dépassé ou il perçoit une erreur, le groupe sanguin correspondant au groupe universel est pris en considération. Le cycle suivant est respectée dans la plate-forme Jason :

1. Percevoir l'environnement.
2. Mettre à jour la base de croyance (faits).
3. Sélection d'un évènement.
4. Retirer tous les plans applicables.
5. Déterminer le plan applicable.
6. Sélection d'une intention pour une prochaine exécution.
7. Exécuter une étape de l'intention.

```
!create_and_use.
+!start : true
  <- !Test_blood(Id);
     // use
     test_sanguin;
     // second use specifying the Id
     glicémie [artifact_id(Id)].
// create the instance
+!Test_blood(C): true
  <- makeArtifact("c0","Test_blood",C).

!observe.
+!observe : true
  <- ?myTool(C);  // discover the tool
     focus(C).
+groupe_sanguin(V)
  <- println("Groupe sanguin ",V).
+ ?erro(error_info)+  ( !abort ; ?aborted)

<- lookupArtifact("O", « test_blood »).
-?myTool(CounterId): true
  <- .wait(10);
     ?myTool(CounterId).
```

Figure IV.13: La définition de l'agent "raisonneur"

4 Discussion

Dans les sections précédentes, nous nous sommes intéressés à l'implementation de notre architecture. Cette architecture comporte un ensemble de modules qui exploitent des concepts de divers domaines. Ces modules peuvent être utilisés pour les applications opérant sur des sources d'information hétérogènes placées dans un environnement ouvert et dynamique. Cette exigence de "réutilisabilité", nous a été dictée par le fait de décomposer le processus métier en un ensemble d'entités pour atteindre la flexibilité. La réutilisabilité est un moyen pour réduire la charge de travail et par conséquent, le coût de production. C'est ce que nous avons choisi.

Hormis cet avantage crucial, notre architecture offre plusieurs autres intérêts. En effet, elle est :

- portable : car elle est programmée en Java ;
- flexible : facilement adaptable, il suffit de changer le mode d'emploi des artifacts.
- interopérable : c'est une possibilité offerte par l'utilisation de la technologie agent, puisqu'elle supporte plusieurs modes de communication telles que la communication inter-agents et la communication agent-service Web. En effet, considérer qu'un artifact encapsule un service est une de nos perspectives futures.

Critères de solutions	Support du critère	Concept sous-jacents
Réutilisation	✓	La modularité atteinte grace aux artifacts permet la réutilisation
Localisation du changement	✓	La centralisation de l'approche en utilisant l'administrateur facilite la localisation du changement ainsi que le changement du contexte d'exécution
Modélisation	∼	Représenter le processus métier en un ensemble d'artifacts encapsulant les activités.
Vérification	✓	Le mode d'emploi et l'utilisation des algèbres de processus
Compréhensibilité	✓	La décomposition et la modularité permettent une meilleur comprehensibilité
Traitement des exceptions	-	/
Réflexion	✓	L'apprentissage et la mise à jour de la base de connaissance permettent un apprentissage sur les changements survenus

Tableau IV.1: Projection de notre solution sur les critères de solutions flexibles

Les premiers essais effectués ont montré que l'idée d'utiliser des artifacts pour exécuter les activités du processus métier offre beaucoup d'avantages, notamment, la réutilisation et l'adaptabilité à l'environnement d'exécution. De plus, l'approche proposée évite d'encombrer l'administrateur ("raisonneur") par les détails d'exécution relatives à chaque activité. Ceci est du à l'encapsulation offerte par les artifacts. Bien que notre architecture offre plusieurs avantages, il y a toujours place pour son amélioration. C'est ainsi que nous prévoyons d'inclure une nouvelle fonctionnalité : prendre en charge le processus métier depuis sa phase de modélisation.

Nous allons récapituler, dans le tableau IV.1 , les critères de solution présentées dans le chapitre I section 4.3 appliquées à notre solution. La notation utilisée dans ce tableau est ✓, ∼ et - qui signifie que le critère correspondant aux solutions flexibles est respectivement : totalement satisfait, partiellement satisfait, ou n'est pas satisfait du tout. Nous précisons également le concept sous-jacents permettant de satisfaire le critère (quand c'est le cas).

Nous pouvons constater que l'utilisation de l'artifact comme unité de construction du processus métier permet d'atteindre bon nombre de propriétés. En effet, la notion de décomposition du processus ainsi élaborée facilite notamment la compréhensibilité, la localisation du changement et également la réutilisation. L'administrateur dans sa conception permet la réflexion par une mise à jour de la base de connaissance.

5 Conclusion

Dans la première partie de ce chapitre, nous avons présenté les systèmes health-care. Nous avons également motivé le choix de notre étude de cas sur un tel système. Dans la seconde partie de ce même chapitre, nous avons présenté notre spécification du premier module important de notre architecture à savoir le module process mining. Pour ce faire nous avons présenté les outils et framework intervenant pour la réalisation de nos objectifs. Nous avons ainsi décrit les phases de transformation de base de données en fichier MXML et l'extraction de règles métier.

Ensuite nous avons abordé l'implémentation de la seconde partie de notre architecture à savoir le module administrateur et le concept artifact.

Pour implémenter cette architecture, nous nous sommes basés sur des standards. Pour cela, nous avons utilisé les technologies AgentSpeak(L) et XML pour représenter les informations échangées entre l'agent et les artifacts.

L'architecture proposée a été implémentée en utilisant la plate-forme Jason ainsi que le framework ProM. La plate-forme Jason permet de simplifier le développement des artifacts et leurs utilisations avec les agents. Le framework ProM nous a permis de développer un plug-in pour l'extraction des règles métiers à partir des fichiers log.

Conclusion générale

L'évolution continue des marchés électroniques et des nouvelles technologies, tels que : e-gouvernement, e-auctions, web services,...Etc., nécessite l'intégration d'une prise en charge flexible des processus métier. Par conséquent, le développement d'applications sensibles au contexte est un domaine d'actualité. Ceci se reflète par les nombreux travaux sur la composition des Web services d'une manière flexible et dynamique. Les BPMSs offrent alors des perspectives prometteuses grâce à la séparation de la logique du processus du code de l'application. Ce qui permet aux changements survenus dans les processus métier d'être intégrés par une adaptation de sa représentation sans modifier son code. Bien que cette séparation fournisse la base pour supporter les changements d'instances de processus à l'exécution ceci est insuffisamment réalisé dans les systèmes actuels. Les systèmes de gestion de processus métier commerciaux existants empêchent les changements des processus métier ou bien les restreignent. D'où un manque de flexibilité qui devient la principale raison qui freine l'expansion de ces systèmes dans la pratique.

C'est la raison pour laquelle le travail proposé dans cette thèse a traité de la flexibilité de la représentation et de l'exécution des processus métier. L'objectif étant de permettre, d'une part une représentation modulaire souple qui prend en compte la nature dynamique des éléments du processus métier et d'autre part, l'exécution de chaque module constituant ce processus métier selon le contexte d'exécution. Nous nous sommes donc intéressés, dans le cadre de cet objectif aux différentes catégories de flexibilité. La catégorie "flexibilité par spécification partielle" vise à gagner en flexibilité par division du processus métier en sous processus. Nous nous sommes également intéressés à la découverte de processus ainsi qu'aux systèmes sensibles au contexte. Notre approche permet de gagner en flexibilité et en adaptation aux changements dans la mesure où l'élément interchangeable du processus est l'artifact.

Contribution

Nous avons débuté cette thèse en identifiant trois problèmes fondamentaux limitant les technologies workflow actuels. En utilisant la notion d'artifact, les techniques du process mining et la sensibilité au contexte, nous avons atteint les objectifs arrêtés comme suit :

Fléxibilité et réutilisabilité

Notre approche sensible au contexte permet d'atteindre la flexibilité de différentes manières.

Premièrement, en fournissant une bibliothèque de règles métier extraite du contexte d'exécution par les techniques du process mining.

Deuxièmement, en représentant le processus métier comme un ensemble d'artifacts chacun encapsulant une activité qui permet d'atteindre la flexibilité en permettant un choix contextuel quant à l'artifact à exécuter par l'administrateur au niveau de l'exécution. De plus, si un choix particulier d'un artifact s'avère inapproprié l'administrateur peut s'en percevoir, l'arrêter et un autre artifact peut être choisi pour le nouveau contexte.

En utilisant une approche modulaire, une spécification peut être une simple définition d'un ensemble d'activités qui se suivent où elles seront définies au niveau de l'exécution. Ceci permet une adaptation dynamique aux environnements difficilement définissables où l'issue du processus est inconnue au niveau de la modélisation. L'artifact, grace à ses instructions opératoires, son formalisme, son abstraction et quasi- autonomie nous a permis d'atteindre ces objectifs.

Cette modularisation nous a permis également d'atteindre une réutilisabilité évidente. En effet, un artifact peut intervenir dans plusieurs processus métier. L'administrateur décide selon sa base de connaissance de l'exécuter ou non.

Adaptation

Les artifacts représentants une activité et l'administrateur permettent aux plans (fil d'exécution des processus) d'être fluides et extensibles pour anticiper un objectif futur selon l'évolution passée. En effet, l'administrateur voit en chaque instance d'exécution une expérience potentielle d'apprentissage. Après chaque exécution d'un artifact la base de connaissances et de croyances de l'administrateur est mise à jour. Ceci permet de déduire un plan possible dans un certain contexte. Additivement, l'utilisation des techniques de process mining permet cet apprentissage en ajoutant, selon les traces d'exécution, de nouvelles règles métier par extraction.

Evolution dynamique

Les instructions opératoires des artifacts permettent une évolution dynamique de l'activité en combinant grace à l'administrateur un ensemble d'instructions opératoires différentes faisant émerger la flexibilité et l'évolution du processus global.

Compréhensibilité de la représentation du processus métier

Ce travail a démontré qu'un processus peut être représenté par un ensemble de fragments. Cette modularité accroît la compréhensibilité et permet une représentation du processus métier à différents niveaux de granularité.

L'architecture du système qui découle de l'approche proposée est basée sur le concept d'agent comme administrateur. Cette architecture repose sur deux éléments clés : un framework modulaire d'artifacts coordonnant pour faire émerger la flexibilité et un composant process mining pour l'extraction des règles métier pour la sensibilité au contexte d'exécution.

Notre architecture possède des fonctionnalités qui prennent en charge les mécanismes de communication et de coordination tels que la communication entre administrateur (agent) et les artifacts. L'implémentation de cette architecture est basée sur la plate-forme Jason et le framework ProM. Cette plate-forme permet de simplifier le développement des systèmes multi-agents tout en fournissant une définition des artifacts. Aussi, nous avons utilisé les standards MXML pour représenter les informations véhiculées par les traces pendant l'exécution.

Perspectives

Les travaux proposés dans cette thèse nous permettent d'ouvrir plusieurs perspectives à court et à moyen terme.

Perspectives à court terme

- **Proposition d'un outil de représentation des processus** : Il s'agit de pouvoir représenter un processus métier comme un ensemble d'artifacts de manière graphique et conviviale.
- **Optimisation de l'échange entre administrateur et artifact** : Pour ce faire il faudrait promouvoir le concept A&A pour qu'il arrive à un niveau plus mature.

Perspectives à long terme

L'approche que nous avons proposée ne traite pas le cas ou un processus métier est partagé entre plusieurs partenaires. En effet, il serait intéressant d'approfondir la réflexion sur une application complexe, où il faut parfois coordonner des fragments de processus issus de modèles de processus différents, d'environnements hétérogènes et des intervenants issus d'organisations différentes. Donc, il faudrait étudier les différentes possibilités d'intégrer ces fragments définis au niveau de différents partenaires pour avoir un processus métier distribué.

Références bibliographiques

[Adams 2005] Michael Adams, Arthur H. M. ter Hofstede, David Edmond et Wil M. P. van der Aalst. *Facilitating Flexibility and Dynamic Exception Handling in Workflows through Worklets*. In CAiSE Short Paper Proceedings, 2005. 29

[Adams 2007a] Michael Adams, Arthur H. M. ter Hofstede, Wil M. P. van der Aalst et David Edmond. *Dynamic, Extensible and Context-Aware Exception Handling for Workflows*. In OTM Conferences (1)Portugal, November 25-30, 2007, Proceedings, Part I, volume 4803 of *Lecture Notes in Computer Science*, pages 95–112. Springer, 2007. 23, 29

[Adams 2007b] Michael James Adams. *Facilitating Dynamic Flexibility and Exception Handling for Workflows*. PhD thesis, Faculty of Information Technology, Queensland University of Technology Brisbane, Australia, May 2007. 26, 27

[Aricci 2008] Aricci. *CARTAGO Annotated Manual CARTAGO v. 1.3.3*. AliceTeam, DEIS, Universit'a di Bologna, Italy Sede di Cesena (FC), fc édition, 09 2008. 99

[Behlouli 2006] Nabiha Belhanafi Behlouli. *Ajout de mécanismes de réactivité au contexte dans les intergiciels pour composants dans le cadre d'utilisateurs nomades*. PhD thesis, Institut National des Télécommunications dans le cadre de l'école doctorale SITEVRY en co-accréditation avec l'Université d'Evry Val d'Essonne, 2006. 44, 45

[Bentellis 2010] Adla Bentellis. *Intégration des Applications d'Entreprises Une approche basée objectif pour la gestion des processus métier flexibles*. PhD thesis, Université Mentouri Constantine, 2010. 48

[Bernal 2010] Jose F. Mejia Bernal, Paolo Falcarin, Maurizio Morisio et Jia Dai. *Dynamic context-aware business process : a rule-based approach supported by pattern identification*. In SAC, pages 470–474, 2010. 63

[Bessai 2008] Kahina Bessai, Bruno Claudepierre, Oumaima Saidani et Selmin Nurcan. *Context-aware Business Process Evaluation and Redesign*. In bpmds08, pages 81–84, 2008. 48

[Bessai 2009] Kahina Bessai et Selmin Nurcan. *Actor-Driven Approach for Business Process. How to Take into Account the Work Environment ?* In Enterprise, Business-Process and Information Systems Modeling, 10th International Workshop, BPMDS 2009, and 14th International Conference, EMMSAD 2009, held at CAiSE 2009, Amsterdam, The Netherlands, June 8-9, 2009.„ pages 187–196, 2009. 47

[Bose 2009] R. P. Jagadeesh Chandra Bose et Wil M.P. van der Aalst. *Context Aware Trace Clustering : Towards Improving Process Mining Results.* In Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 - May 2, 2009, Sparks, Nevada, USA, pages 401–412. SIAM, 2009. 25

[Boukhebouze 2010] Mohamed Boukhebouze. *Gestion de changement et vérification formelle de processus métier : une approche orientée régles.* PhD thesis, Institut national des sciences appliquées de Lyon, France, 2010. 50, 63, 96

[Bratosin 2010] Carmen Bratosin, Natalia Sidorova et Wil M. P. van der Aalst. *Discovering Process Models with Genetic Algorithms Using Sampling.* In KES (1), Springer Lecture Note in Computer Science, pages 41–50, 2010. 36

[Carlsen 1997] Steinar Carlsen. *Conceptual Modeling and Composition of Flexible Workflow Model.* PhD thesis, Information Systems Group Department of Computer and Information Science Faculty of Physics, Informatics and Mathematics Norwegian University of Science and Technology, 1997. 28

[Casati 2000] Fabio Casati, Ski Ilnicki, Li jie Jin, Vasudev Krishnamoorthy et Ming-Chien Shan. *eFlow : A Platform for Developing and Managing Composite e-Services.* In AIWoRC, Industry Working Conference on Research Challenges (AIWoRC 2000), Next Generation Enterprises : Virtual Organizations and Mobile / Pervasive Technologies, 27-29 April 2000, Buffalo, NY, USA, pages 341–348, 2000. 26

[Celino 2007] Irene Celino, Ana Karla Alves de Medeiros, Gernot Zeissler, Michael Oppitz, Federico Michele Facca et Stefan Zoeller. *Semantic Business Process Analysis.* In SBPM, 2007. 34

[Chalmers 2004] Dan Chalmers, Naranker Dulay et Morris Sloman. *Towards Reasoning About Context in the Presence of Uncertainty.* In Proceedings of Workshop on Advanced Context Modelling, Reasoning And Management at UbiComp 2004, 2004. 43

[Chiu 2000] Dickson K. W. Chiu, Qing Li et Kamalakar Karlapalem. *Web Interface-Driven Cooperative Exeption Handling in ADOME Workflow Management System.* In WISE, Proceedings of the First International Conference on Web Information Systems Engineering, Volume I (Main Program), Hong Kong, China, June 19-21, 2000, pages 174–182. IEEE Computer Society, 2000. 26

[Cook 1998] Jonathan E. Cook et Alexander L. Wolf. *Discovering Models of Software Processes from Event-Based Data.* ACM Trans. Softw. Eng. Methodol., vol. 7, no. 3, pages 215–249, 1998. 34

[Crerie 2009] Raphael Crerie, Fernanda Araujo Baião et Flávia Maria Santoro. *Discovering Business Rules through Process Mining*. In BMMDS/EMMSAD, pages 136–148, 2009. viii, 64

[Dadam 2009] Peter Dadam et Manfred Reichert. *The ADEPT project : a decade of research and development for robust and flexible process support*. Computer Science - R&D, vol. 23, no. 2, pages 81–97, 2009. 27, 52

[Davenport 1990] Thomas H. Davenport et J.E. Short. *The new industrial engineering : Information technology and business process redesign*. Sloan Management Review, pages 4–10, 1990. 32

[de Medeiros. 2006a] Ana Karla Alves de Medeiros. *Genetic Process Mining*. PhD thesis, Eindhoven : Technische Universiteit Eindhoven, 2006. vii, 40

[de Medeiros 2006b] Ana Karla Alves de Medeiros et A.J.M.M. (Ton) Weijters. *ProM Framework Tutorial*. Technische Universiteit Eindhoven Eindhoven, The Netherlands, November 2006. 39

[de Medeiros 2007a] Ana Karla A. de Medeiros, A. J. M. M. Weijters et Wil M. P. van der Aalst. *Genetic process mining : an experimental evaluation*. Data Min. Knowl. Discov., vol. 14, no. 2, pages 245–304, 2007. 36

[de Medeiros 2007b] Ana Karla Alves de Medeiros, Carlos Pedrinaci, Wil M. P. van der Aalst, John Domingue, Minseok Song, Anne Rozinat, Barry Norton et Liliana Cabral. *An Outlook on Semantic Business Process Mining and Monitoring*. In On the Move to Meaningful Internet Systems OTM Workshops (2), Springer, Lecture Notes in Computer Science, Vilamoura, Portugal, November 25-30, pages 1244–1255, 2007. 39

[der Aalst 2005] Wil M. P. Van der Aalst et Arthur H. M. ter Hofstede. *YAWL : yet another workflow language*. Inf. Syst., vol. 30, no. 4, pages 245–275, 2005. 15, 53

[Dey 2000] Anind K. Dey. *Providing Architectural Support for Building Context-Aware Applications*. PhD thesis, Georgia Institute of Technology, 2000. 43

[Dey 2001] Anind K. Dey. *Understanding and Context*. Personal and Ubiquitous Computing, vol. 5, no. 1, pages 4–7, 2001. 43, 60

[Dinont 2006] Cédric Dinont, Philippe Mathieu, Emmanuel Druon et Patrick Taillibert. *Artifacts for time-aware agents*. In AAMAS, pages 593–600, 2006. 69

[Dinont 2007] Cédric Dinont. *Calculs longs et partage des ressources processeur dans les systèmes multi-agents cognitifs*. PhD thesis, Université des Sciences et Technologies de Lille,France, 2007. 69, 73

[Fokkink 2000] Wan Fokkink. Introduction to process algebra. Texts in theoretical computer science. Springer, 2000. 72

[Gaaloul 2006] Walid Gaaloul. *La Découverte de Workflow Transactionnel pour la Fiabilisation des Exécutions*. PhD thesis, Université Henri Poincaré -Nancy 1, France, 2006. vii, 12, 13, 14, 15, 16, 31, 32, 33

[Giaccari 2002] Philippe Giaccari. Gestion des processus business et modélisation des processus business d'une start-up de type cybermédiaire. Master's thesis, Université De Lausanne. École Des Hautes Études Commerciales, 2002. 10

[Günther 2006a] Christian W. Günther et Wil M. P. van der Aalst. *A Generic Import Framework for Process Event Logs*. In Business Process Management Workshops, pages 81–92, 2006. 35, 39

[Günther 2006b] C.W. Günther et W.M.P. van der Aalst. *Process Mining in Case Handling Systems*. In In F. Lehner, H. Nosekabel, and P. Kleinschmidt, editors, Proceedings of the Multikonferenz Wirtschaftsinformatik 2006 (MKWI '06). GITO-Verlag, Berlin„ 2006. 42

[Günther 2007] Christian W. Günther, Stefanie Rinderle-Ma, Manfred Reichert, Wil M.P. van der Aalst et Jan Recker. *Using Process Mining to Learn from Process Changes in Evolutionary Systems*. Int. J. Business Process Integration and Management, vol. 1, no. 1/2/3, pages 111–131, 2007. 23, 35, 41, 52

[Gupta 2007] Shaifali Gupta. Workflow and process mining in healthcare. Master's thesis, TECHNISCHE UNIVERSITEIT EINDHOVEN Department of Mathematics and Computer Science, 2007. 89, 92

[Hammer 1990] Michael Hammer. *Reengineering Work : Don't Automate, Obliterate*. Harvard Business Review, pages 70–91, 1990. 32

[HAMRI 2010] Salah HAMRI. *Interopérabilité des Modèles de Workflow*. PhD thesis, Université Mentouri, Constantine, Algerie, 2010. 76

[Herbst 2000] Joachim Herbst. *A Machine Learning Approach to Workflow Management*. In ECML, pages 183–194, 2000. 34, 35

[Kindler 2006] Ekkart Kindler, Vladimir Rubin et Wilhelm Schäfer. *Incremental Workflow Mining for Process Flexibility*. In Proceedings of the CAISE*06 Workshop on Business Process Modelling, Development, and Support,BPMDS, Luxemburg, June 5-9„ 2006. 41

[Kobayashi 2005] Marina Kobayashi, Susan R. Fussell, Yan Xiao et F. Jacob Seagull. *Work coordination, workflow, and workarounds in a medical context*. In CHI Extended Abstracts, pages 1561–1564, 2005. 25

[Kumar 2006] Kuldeep Kumar et Murali Mohan Narasipuram. *Defining Requirements for Business Process Flexibility*. In Proceedings of the CAISE*06 Workshop on Business Process Modelling, Development, and Support,BPMDS, Luxemburg, June 5-9„ 2006. 41

[Leymann 2000] Frank Leymann et Dieter Roller. Production workflow - concepts and techniques. Prentice Hall, 2000. 11, 13

[Loghin 2008] Gligor-Calin Loghin. *Observer un Environnement Numérique de Travail pour réguler les activités qui s'y déroulent.* PhD thesis, Université Technique de Cluj-Napoca (Roumanie) et de l'Université de Savoie (France), 2008. 44

[Marustera 2001] Laura Marustera, Wil van der Aalsta et Ton Weijtersa. *Automated Discovery of Workflow Models for Hospital Data.* In Proceedings of the 13th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC 2001)„ pages 183–190, 2001. 34, 36

[Montali 2010] Marco Montali, Maja Pesic, Wil M. P. van der Aalst, Federico Chesani, Paola Mello et Sergio Storari. *Declarative specification and verification of service choreographiess.* TWEB, vol. 4, no. 1, 2010. 37

[national de la recherche scientifique.] Centre national de la recherche scientifique. Dictionnaire de la langue française du xixeme et xxeme siècle. centre national de la recherche scientifique. 42

[Omicini 2004] Andrea Omicini, Alessandro Ricci, Mirko Viroli, Cristiano Castelfranchi et Luca Tummolini. *Coordination Artifacts : Environment-based Coordination for Intelligent Agents.* In Nicholas R. Jennings, Carles Sierra, Liz Sonenberg et Milind Tambe, editeurs, 3rd international Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), volume 1, pages 286–293, New York, USA, 19–23 Juillet 2004. ACM. 69, 70

[Pesic 2007] Maja Pesic, Helen Schonenberg et Wil M. P. van der Aalst. *DECLARE : Full Support for Loosely-Structured Processes.* In IEEE, editeur, EDOC, pages 287–300. IEEE, 2007. 52

[Pesic 2009] Maja Pesic, Helen Schonenberg et Wil M. P. van der Aalst. *DECLARE Demo : A Constraint-based Workflow Management System.* In Business Process Management Demonstration Track (BPMDemos), Ulm, Germany, September 8, 2009, 2009. 52

[Pro] *User Manual for converting data from MS Access database to the ProM MXML format.* 94

[Regev 2005] Gil Regev et Alain Wegmann. *A Regulation-Based View on Business Process and Supporting System Flexibility.* In CAiSE'05, 2005. 24, 50

[Regev 2006] Gil Regev, Pnina Soffer et Rainer Schmidt. *Taxonomy of Flexibility in Business Processes.* In roceedings of the CAISE*06 Workshop on Business Process Modelling, Development, and Support BPMDS '06, Luxemburg, June 5-9, 2006, 2006. 25, 40, 50

[Reichert 2009] Manfred Reichert, Stefanie Rinderle-Ma et Peter Dadam. *Flexibility in Process-Aware Information Systems.* T. Petri Nets and Other Models of Concurrency, Springer, Lecture Notes in Computer Science, vol. 2, pages 115–135, 2009. 25, 52

[Ricci 2005] Alessandro Ricci, Mirko Viroli et Andrea Omicini. *Programming MAS with Artifacts*. In Rafael H. Bordini, Mehdi Dastani, Jürgen Dix et Amal El Fallah-Seghrouchni, editeurs, Programming Multi-Agent Systems, Third International Workshop(PROMAS), volume 3862 of *Lecture Notes in Computer Science*, pages 206–221, Utrecht, The Netherlands, July,26 2005. Springer. 69, 74

[Rinderle 2004] STEFANIE BEATE Rinderle. *Schema Evolution in Process Management Systems*. PhD thesis, UniversitÄat Ulm Abt. Datenbanken und Informationssysteme, 2004. vii, 1, 2, 3, 5, 19

[Rolland 2010] Colette Rolland et Selmin Nurcan. *Business Process Lines to deal with the Variability*. In Proceedings of the 43rd Hawaii International Conference on System Sciences, IEEE Computer Society - 2010, pages 1–10, 2010. 47

[Rosemann 2006] Michael Rosemann et Jan Recker. *Context-aware Process Design Exploring the Extrinsic Drivers for Process Flexibility*. In BPMDS, 2006. 48

[Rozinat 2005] Anne Rozinat et Wil M. P. van der Aalst. *Conformance Testing : Measuring the Fit and Appropriateness of Event Logs and Process Models*. In Business Process Management Workshops, pages 163–176, 2005. 37

[Rozinat 2006] Anne Rozinat et Wil M. P. van der Aalst. *Decision Mining in ProM*. In Business Process Management, Springer, Lecture Note in Computer Science, pages 420–425, 2006. 36, 38, 39, 64, 81

[Russell 2007] Nicholas Charles Russell. *Foundations of Process-Aware Information Systems*. PhD thesis, Faculty of Information Technology, Queensland University of Technology Brisbane, Australia, December 2007. vii, 11, 19, 20, 22, 29

[Saidani 2006a] Oumaima Saidani et Selmin Nurcan. *FORBAC : A Flexible Organisation and Role-Based Access Control Model for Secure Information Systems*. In ADVIS, pages 364–376, 2006. 47

[Saidani 2006b] Oumaima Saidani et Selmin Nurcan. *A Role-Based Approach for Modeling Flexible Business Processes*. In BPMDS, 2006. 47

[Saidani 2007] Oumaima Saidani et Selmin Nurcan. *Multi-Level Delegation for Flexible Business Process Modeling*. In IRMA-BPM 2007, 2007. 47

[Saidani 2008] Oumaima Saidani et Selmin Nurcan. *Towards Situational Business Process Meta-Modelling*. In CAiSE Forum, pages 93–96, 2008. 82

[Saidi 2009] Rajaa Saidi. *Conception et usage des composants métier processus pour les systèmes d'information*. PhD thesis, Institut polytechnique de Grenoble et de l'Université Mohammed V Agdal Rabat, 2009. 57

[Schimm 2002] Guido Schimm. *Process Miner - A Tool for Mining Process Schemes from Event-Based Data*. In Logics in Artificial Intelligence, European Conference, JELIA2002, Cosenza, Italy, September, 23-26, volume 2424 of *Lecture Notes in Computer Science*, pages 525–528. Springer, 2002. 34, 35

[Schonenberg 2008a] Helen Schonenberg, Ronny Mans, Nick Russell, Nataliya Mulyar et Wil M. P. van der Aalst. *Process Flexibility : A Survey of Contemporary Approaches*. In CIAO ! / EOMAS, pages 16–30, 2008. 25, 50, 51

[Schonenberg 2008b] Helen Schonenberg, Ronny Mans, Nick Russell, Nataliya Mulyar et Wil M. P. van der Aalst. *Towards a Taxonomy of Process Flexibility*. In CAiSE Forum, pages 81–84, 2008. 25, 50

[Schonenberg 2008c] Helen Schonenberg, Barbara Weber, Boudewijn F. van Dongen et Wil M. P. van der Aalst. *Supporting Flexible Processes through Recommendations Based on History*. In Business Process Management, 6th International Conference, BPM 2008, Milan, Italy, September 2-4, 2008., volume 5240 of *Lecture Notes in Computer Science*, pages 51–66. Springer, 2008. 41

[Tam 2008] NGUYEN Thi Thanh Tam. *Codèle : Une approche de composition de modèles pour la Construction de Systèmes à Grande Échelle*. PhD thesis, Université Joseph Fourier de Grenoble, 2008. 67, 68

[Tanguy 2003] Christian Tanguy. Business process management : De la modélisation à l'exécution, positionnement par rapport aux architectures orientées services. Intalio, 2003. 10, 15

[Vajirkar 2003] Pravin Vajirkar, Sachin Singh et Yugyung Lee. *Context-Aware Data Mining Framework for Wireless Medical Application*. In Vladimír Marík, Werner Retschitzegger et Olga Stepánková, editeurs, Database and Expert Systems Applications, 14th International Conference, DEXA 2003, volume 2736 of *Lecture Notes in Computer Science*, pages 381–391, Prague, Czech Republic, September 1-5 2003. Springer. 82

[van der Aalst 2003a] Wil M. P. van der Aalst, Arthur H. M. ter Hofstede et Mathias Weske. *Business Process Management : A Survey*. In Business Process Management, International Conference, BPM 2003, Eindhoven, The Netherlands, June 26-27, 2003, Proceedings, Springer, LNCS,, pages 1–12, 2003. 18, 31

[van der Aalst 2003b] Wil M. P. van der Aalst, Boudewijn F. van Dongen, Joachim Herbst, Laura Maruster, Guido Schimm et A. J. M. M. Weijters. *Workflow mining : A survey of issues and approaches*. Data Knowl. Eng., vol. 47, no. 2, pages 237–267, 2003. 34, 35, 36

[van der Aalst 2004] Wil M. P. van der Aalst, Ton Weijters et Laura Maruster. *Workflow Mining : Discovering Process Models from Event Logs*. IEEE Trans. Knowl. Data Eng., vol. 16, no. 9, pages 1128–1142, 2004. 35

[van der Aalst 2005a] Wil M. P. van der Aalst et Ana Karla A. de Medeiros. *Process Mining and Security : Detecting Anomalous Process Executions and Checking Process Conformance.* Electr. Notes Theor. Comput. Sci., vol. 121, pages 3–21, 2005. 37

[van der Aalst 2005b] Wil M. P. van der Aalst, Hajo A. Reijers et Minseok Song. *Discovering Social Networks from Event Logs.* Computer Supported Cooperative Work, vol. 14, no. 6, pages 549–593, 2005. 36

[van der Aalst 2005c] Wil M. P. van der Aalst, Mathias Weske et Dolf Grünbauer. *Case handling : a new paradigm for business process support.* Data Knowl. Eng., vol. 53, no. 2, pages 129–162, 2005. 27, 52

[Van der Aalst 2006a] W. Van der Aalst, Gunther.C, J Recker et M Reichert. *Using Process Mining to Analyze and Improve Process Flexibility -Position Paper -.* pages 168–177, 2006. 41

[van der Aalst 2006b] Wil M. P. van der Aalst, Marlon Dumas, Chun Ouyang, Anne Rozinat et H. M. W. Verbeek. *Choreography Checking : An Approach based on BPEL and Petri Nets.* In The Role of Business Processes in Service Oriented Architectures, 2006. 37

[van der Aalst 2008a] Wil M. P. van der Aalst. *Discovery, Verification and Conformance of Workflows with Cancellation.* In ICGT, Springer, Lectures Note in Computer Science, pages 18–37, 2008. 37

[van der Aalst 2008b] Wil M. P. van der Aalst et H. M. W. (Eric) Verbeek. *Process Mining in Web Services : The WebSphere Case.* IEEE Data Eng. Bull., vol. 31, no. 3, pages 45–48, 2008. 36

[van der Aalst 2009a] Wil M. P. van der Aalst. *Process-Aware Information Systems : Lessons to Be Learned from Process Mining.* T. Petri Nets and Other Models of Concurrency, vol. 2, pages 1–26, 2009. vii, 37

[van der Aalst 2009b] Wil M. P. van der Aalst, Maja Pesic et Helen Schonenberg. *Declarative workflows : Balancing between flexibility and support.* Computer Science-RD, vol. 23, no. 2, 2009. 52

[van der Aalst 2010a] Wil M. P. van der Aalst. *Process Discovery : Capturing the Invisible.* In IEEE Computational Intelligence Magazine, 5(1), pages 28–41, 2010. 36

[van der Aalst 2010b] Wil M. P. van der Aalst, Maja Pesic et Minseok Song. *Beyond Process Mining : From the Past to Present and Future.* In CAiSE, Springer, Lecture Notes in Computer Science, Hammamet, Tunisia, 7-9„ pages 38–52, 2010. 41

[van der Aalst 2010c] Wil M. P. van der Aalst, Vladimir Rubin, H. M. W. Verbeek, Boudewijn F. van Dongen, Ekkart Kindler et Christian W. Günther. *Process mining : a two-step approach to balance between underfitting and overfitting.* Software and System Modeling, vol. 9, no. 1, pages 87–111, 2010. 34, 35

[van Dongen] Boudewijn F. van Dongen et Wil M. P. van der Aalst. *A Meta Model for Process Mining Data*. In INTEROP'05, Enterprise Modelling and Ontologies for Interoperability, Proceedings of the Open Interop Workshop on Enterprise Modelling and Ontologies for Interoperability, EMOI-INTEROP Co-located with CAiSE'05 Conference, Porto (Portugal), 13th-14th June,2005. 27, 35

[van Dongen 2005] Boudewijn F. van Dongen, Ana Karla A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters et Wil M. P. van der Aalst. *The ProM Framework : A New Era in Process Mining Tool Support*. In Applications and Theory of Petri Nets 2005, 26th International Conference ICATPN,Springer Lecture Note in Computer Science, Miami, USA, June 20-25,, pages 444–454, 2005. 38

[van Dongen 2009] Boudewijn F. van Dongen et A. Adriansyah. *Process Mining : Fuzzy Clustering and Performance Visualization*. In Business Process Management Workshops, Springer, Lecture Notes in Business Information Processing, pages 158–169, 2009. 36

[Viroli 2004] Mirko Viroli et Alessandro Ricci. *Instructions-Based Semantics of Agent Mediated Interaction*. In 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), pages 102–109, New York, NY, USA, 19-23 August 2004. 72

[Viroli 2005] Mirko Viroli, Andrea Omicini et Alessandro Ricci. *Engineering MAS environment with artifacts*. In H. Van Dyke Parunak Danny Weyns et curatori Fabien Michel, editeurs, 2nd International Workshop Environments for Multi-Agent Systems (E4MAS 2005), AAMAS 2005, volume 3862 of *Lecture Notes in Computer Science*, pages 1–16, Utrecht, The Netherlands, July,26 2005. Springer. 69, 74

[Weber 2008] Barbara Weber, Manfred Reichert et Stefanie Rinderle-Ma. *Change patterns and change support features - Enhancing flexibility in process-aware information systems*. Data Knowl. Eng., vol. 66, no. 3, pages 438–466, 2008. 25

[Weber 2009] Barbara Weber, Manfred Reichert, Stefanie Rinderle-Ma et Werner Wild. *Providing Integrated Life Cycle Support in Process-Aware Information Systems*. Int. J. Cooperative Inf. Syst., vol. 18, no. 1, pages 115–165, 2009. 25

[Weijters 2002] Ton Weijters et Jan Paredis. *Genetic rule induction at an intermediate level*. Knowl.-Based Syst., vol. 15, no. 1-2, pages 85–94, 2002. 34

[Zerari 2010a] Mounira Zerari et Mahmoud Boufaida. *An artifact-based architecture for a better flexibility of business processes*. In Proceedings of the 12th International Conference on Enterprise Information Systems (ICEIS 2010), Volume 1, DISI, Funchal, Madeira, Portugal, June 8 - 12, 2010, pages 359–365, 2010. 54

[Zerari 2010b] Mounira Zerari et Mahmoud Boufaida. *Context-Aware Process Mining Framework for Business Process Flexibility*. In ACM iiWAS 2010, Paris, France, November 8-10, 2010, pages 419–424, 2010. 75

[Zerari 2011] Mounira Zerari et Mahmoud Boufaida. *Dynamic Context-Aware Business Process Flexibility : An Artifact-based Approach using Process mining.* International Journal of Business Intelligence and Data mining, vol. 6, no. 4, pages 345–361, 2011. 91

Titre : Une approche pour l'amélioration de la flexibilité des processus métier basée sur les techniques du process mining.

Résumé

Une évolution continue des paramètres, des contraintes,des conditions d'exécution et des besoins du processus métier, non complètement prévisible initialement, exige des systèmes de gestion de processus une représentation fiable et une adaptation aux conditions d'exécution. Dans cette thèse, nous nous intéressons à assurer une exécution réactives par l'analyse des traces d'exécutions assurant une sensibilité au contexte d'exécution et donc une flexibilité des processus métier.

Pour ce faire, nous proposons dans ce travail de recherche une approche pour améliorer la flexibilité des processus métier en utilisant les techniques du process mining pour découvrir le contexte d'exécution ainsi qu'une nouvelle représentation des processus métier en un ensemble de fragments qui s'exécutent selon le contexte pour faire émerger la flexibilité.

Cette démarche est basée conjointement sur le concept d'artifact pour encapsuler chacune des activités des processus métier ainsi que sur l'extraction des règles metier à partir des fichiers log pour la description du contexte d'exécution. Nous avons également proposé une architecture pour la mise en oeuvre de ces différents concepts.

La contribution majeure de notre proposition est qu'elle permet de tenir compte des besoins d'évolution du processus observés dans la phase d'exécution et d'adopter une approche modulaire du processus métier. Ceci nous permet d'assurer des exécutions correctes et fiables, d'un coté et l'avantage de réutilisabilité d'expressivité et de simplicité des processus métier d'un autre coté.

Mots clés : Artifact, flexibilité par spécification partielle, process mining, règles métier, sensibilité au contexte.

Title : An approach for business processes flexibility using process mining techniques.

Abstract

A continuous evolution of business process parameters, constraints and needs, hardly unforeseeable initially, requires from the business process management systems a reliable process representation and an adaptability to execution conditions.

In this thesis, we are interested in developing a reactive execution through a process logs analysis ensuring a context aware business process execution.

For that purpose, our approach starts by collecting workflow logs. Then, we extract business rules that represent the context execution. We consider, in our approach that a process mining is a collection of fragment that encapsulate an activity of business process. Those fragments are artifact that have a function, interface, and formal manual.

The major contribution of our proposal is its ability to take into account process evolution needs observed at runtime. The aspect of modularity in our approach allows reusability, expressivity and reduce complexity.

Keywords : Artifact, flexibility by underspecification, process mining, business rules, context awareness.

العنوان: منهجية للتحسين من مرونة المعالجات المهنية باستعمال تقنيات الاستخراج.

ملخص :

هذا العمل يقترح نهجا جديدا لتحسين من مرونة المعالجات المهنية (Processus métier). بالفعل المؤسسات الحالية ترى أن نشاطها يتطور بسرعة مذهلة خاصة في سياق العولمة الحالي. لذلك استلزم عليها جعل معالجاتها المهنية أكثر مرونة تتأقلم مع التغيرات الجارية في وسط أين الأسواق العالمية في حاجة إلى ديناميكية مهمة. وعليه مساهمتنا تتمثل في نمذجة و تمثيل المنهج العملي كمجموعة من الكيانات, كل منها تغلف نشاط عملي من أنشطة المعالج. تدعى هذه الكيانات بأدوات (Artifacts). وذلك بأخذ بعين الاعتبار سياق تنفيذ المعالجات باستعمال تقنيات الاستخراج (Process Mining).البنية المقترحة تسمح من تخفيض التعقد في تمثيل المعالجات, الرفع من الشمول هذا كنتيجة للتفكك و تحبب المعالج المهني. تعتبر أيضا قابلة لتكيف لكونها حساسة للسياق.