République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université de 8 Mai 1945 – Guelma -

Faculté des Mathématiques, d'Informatique et des Sciences de la matière

Département d'Informatique

Mémoire de Fin d'études Master

Filière : Informatique

Option : Informatique Académique

Thème :

Un Framework pour l'automatisation de l'identification de service à base des représentations BPMN

Encadré Par :

Mr. Séridi ali

Présenté par :

Grini Ahlem

Bennour Khedidja

Juin 2017

Remerciements

Nous remercions notre dieu(Allah) qui nous a donné la volonté, la patience et la force pour réaliser ce travail.

*Mr **SERIDI Ali**, pour la confiance qu'il nous a témoignée en acceptant de diriger ce travail et pour nous avoir accordé de son temps et avoir mis à notre disposition ses compétences et ses conseils pour une meilleure maîtrise du sujet.*

Nous tenons à présenter nos sincères remerciements aux membres de jury.
Tous les enseignants du département d'informatique, qui ont assisté à nos débuts en informatique, pour leurs précieux conseils.

Nos collègues de fin de cycle, qui nous ont donné leurs encouragements toute la durée de réalisation de ce travail.

C'est avec un grand plaisir que nous apportons ce modeste travail à tous ceux qui nous ont gratifiés de leur soutien et de leur Confiance.

GRINI AHLEM & BENNOUR KHADIDJA

Dédicace

Je dédie ce modeste travail, fruit de mes années d'étude à :

Mes parents *qui n'ont jamais eu de cesse de croire en moi, que ce travail soit pour vous un motif supplémentaire de fierté.*

A mes frères, mes sœurs *qui ont été le support moral durant ma vie pour lesquels je souhaite le meilleur, que dieu nous garde unis à jamais.*

Mes collègues *de fin de cycle, qui nous ont donné leurs encouragements toute la durée de réalisation de ce travail.*

Toutes les personnes qui nous ont aidé et soutenu de prés ou de loin tout le long de ce travail.

Et à toute la promotion 2017

Bennour khadidja

Dédicace

Je dédie ce travail à mes chers parents

Pour leur volonté, leurs énormes sacrifices et pour l'amour qu'ils ont su me donner tout au long de mes études. Je leur suis infiniment reconnaissant.

A mon frère Nacer Eddine sa femme Lamya, ma sœur Souaad, son marié Ali, leurs enfants mes chers Meryem et Amina, et à mon frère Saif Eddine ;

Mes cousins et cousines ;

Mes camarades de promotion ;

Mes amis ;

A tous ceux qui de près ou de loin ont contribué à la réalisation de ce travail;

Enfin, tous ceux que je n'ai pas pu citer.

Grini Ahlem

Résumé :

L'évolution rapide des nouvelles technologies n'a cessé de laisser son impact sur les systèmes d'information des entreprises ce qui incite ces entreprises à changer leurs stratégies, méthodes de conception et architectures car elles leur fournissent plus de souplesse et de simplicité surtout quand il s'agit de communications avec l'environnement extérieur.

Parmi ces nouvelles architectures, l'architecture orientée services (littéralement en anglais SOA pour « Service Oriented Architecture ») qui permet la réutilisation (réutiliser un ou un ensemble de composants), la modularité (remplacement facile d'un service), l'évolutivité (mise à jour d'un service) et une maintenance facile. Cependant, la technologie des services Web représente l'implémentation la plus utilisée dans ce type d'architecture puisqu'ils résolvent le problème majeur d'interopérabilité entre les différents systèmes grâce à leurs standards.

Notre travail s'intéresse à proposer une approche qui permettra d'automatiser la phase d'identification de service qui est une phase inévitable dans n'importe quelle approche de mise en œuvre d'une architecture orientée service.

Nous avons choisi une approche Top down qui prend comme point de départ le point de vue de l'entreprise représenté par des diagrammes BPMN pour arriver à la fin à travers notre Framework à extraire et à identifier automatiquement des services candidats qui peuvent être programmés suivant un standard de la SOA. Cela permettra aux concepteurs de gagner énormément de temps dans la phase d'identification surtout lorsque l'application est assez grande.

Une méthode d'évaluation des services candidats résultants a été aussi proposée afin d'avoir une idée sur le taux de confiance que nous pourrions accorder aux services identifiés.

Les mots clé :

SOA (Service Oriented Architecture), BPMN, Identification automatique des services, approche par les processus métiers.

Chapitre 01 : L'architecture orientée service (SOA) et les services web

Chapitre 02 : Approche processus

Chapitre 03 : Conception

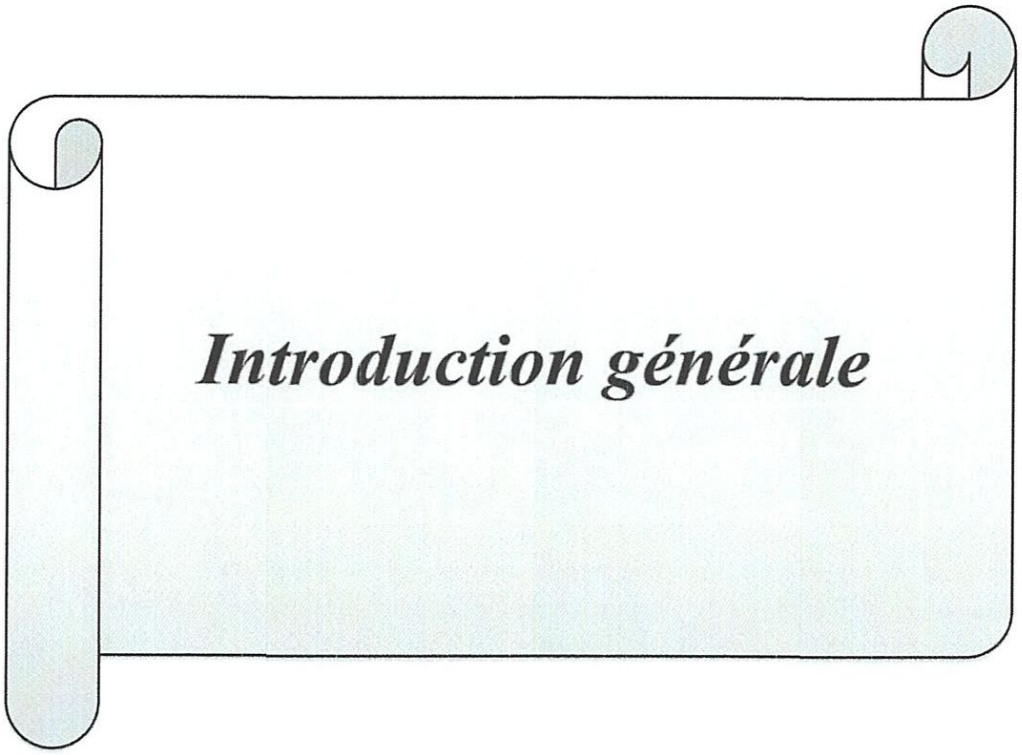
Chapitre 04 : Implémentation

Liste des figures :

Liste des tableaux :

Liste des abréviations :

SOA	Services Oriented Architecture
WS	Web Service
SIM	Service Identification Method
SI	Système Information
BPM	Business Process Management
BPMN	Business Process Modeling Notation
IT	*Information* Technology
ESB	Entreprise Service Bus
BSP	Business System Planning
XML	Extensible Markup Language
W3C	World Wide Web Consortium
WSDL	Web Service Description Language
SOAP	Simple Object Access Protocol
UDDI	Universal Description, Discovery and Integration
IBM	International Business Machines
BPEL	Business Process Execution Language
UML	Unified Modeling Language
DSI	Directeur du Système d'Information
JEE	Java Enterprise Edition
OMG	Object Management Group
BMPI	Business Process Management Initiative
ISO	International Organization for Standardization
XPDL	XML Processing Description Language
WfMC	Workflow Management Coalition
JDOM	Java Modèle d'Objet de Document
ODE	Orchestration Director Engine
URI	Uniform Rcsourcc Idcntificr
RPC	Remote Procedure Call


Introduction générale

Introduction générale :

La diversité des besoins et exigences des clients, la difficulté de répondre à leurs demandes en assurant une bonne qualité à moindre coûts et la concurrence sur marchés ont poussé les entreprises à revoir leurs stratégies de production et à s'orienter vers la collaboration et la coopération. [1]

A cet effet, leurs systèmes d'information implémentés (en particulier orienté objet) deviennent plus lourds et rigides et souffrent d'une multitude de problèmes essentiellement l'hétérogénéité, l'agilité, l'interopérabilité et l'ouverture vers l'extérieur. [1]

Dans les architectures informatiques traditionnelles, les processus métier, les applications et les données sont souvent enfermés dans des silos indépendants et incompatibles, chers à maintenir et obligeant les utilisateurs à naviguer entre divers réseaux, applications et bases de données afin d'effectuer des tâches bien spécifiques. Ces architectures ont provoqué aux SI un blocage vis-à-vis des nouvelles demandes de modifications et d'alignement avec les nouveaux besoins de l'entreprise, ce qui a conduit à un manque d'agilité et d'ouverture vers l'extérieur. [1]

L'architecture orientée services (SOA pour Service Oriented Architecture), en tant que paradigme architectural, constitue un accomplissement de telles approches en permettant l'exposition, l'interconnexion et la réutilisation d'applications à base de services et ce, en dépit de l'hétérogénéité des domaines, des technologies et des fournisseurs impliqués. L'une des mises en œuvre d'applications à base d'architecture orientée services la plus utilisée est la technologie des services Web. Elle se concentre sur la définition d'un système logiciel comme une application distribuée qui implique une composition d'agents logiciels indépendants et interconnectés. Ces agents fournissent leurs fonctionnalités sous forme de services disponibles sur le Web et collaborent ensemble dans le but de réaliser un ensemble d'objectifs offrant des capacités à tendance commerciale. [1]

L'identification des services est l'une des activités difficiles dans le développement de modèles axés sur les services. L'identification des services concerne la détermination des services appropriés à mettre en œuvre dans une architecture axée sur les services et la définition des fonctions qui devraient faire partie de chaque service, la plupart des approches actuelles relais sur les descriptions de processus métier pour identifier les services.

C'est à travers ce projet que nous essayons de répondre à cette problématique en se fixant l'objectif d'automatiser cette phase cruciale qui est l'identification de service à partir du standard de la BPM, le BPMN2.
En outre, il est nécessaire d'évaluer la qualité des services candidats par un ensemble de mesures de qualité. Un ensemble spécifique de ces métriques est le couplage et la cohésion qui mesure les dépendances fonctionnelles et informatives entre les tâches / processus dans un modèle de processus métier.

Dans ce projet, nous fournissons un cadre qui aide à évaluer la qualité des services identifiés à un stade précoce du développement de modèles SOA, ouvrant la voie à un moyen de mesurer chacun des attributs de qualité du service.

Notre mémoire est organisé en quatre chapitres :

Le premier chapitre : nous allons introduire brièvement les concepts de base relatifs à l'architecture orientée service et la technologie de l'implémentation de la SOA qui est les services web et les différents approches d'identification des services.
Le deuxième chapitre : Dans ce chapitre nous présenterons l'approche par processus, la notion de BPM, la norme BPMN, les diagrammes, Les éléments de base de la notation BPMN afin de mettre des propositions pour identifier les services.
Le troisième chapitre : nous allons expliquer le processus que nous avons suivi pour atteindre notre objectif et comment nous avons évalué l'approche d'identification des services candidats.
Le quatrième chapitre : ce chapitre comportera les différents détails de l'implémentation où nous définirons les outils utilisés et présenterons les différentes interfaces ainsi que les résultats obtenus.

Enfin, nous clôturons ce mémoire par une conclusion générale.

Chapitre 01 :

Architecture orienté service(SOA) et Web Service

1. Introduction :

Depuis quelques années, la notion d'**Architecture Orientée Service** (SOA : *Service Oriented Architecture*) s'est rapidement répandue et a été largement acceptée comme une architecture support du système d'information de l'entreprise.

À ses débuts, la SOA s'est imposée comme une approche pour l'architecture logicielle des systèmes. Suite à son succès grandissant, le concept de SOA s'est orienté vers des aspects très variés qui dépassent d'une certaine manière largement le domaine initial qu'est l'architecture logicielle. Aujourd'hui la SOA est considérée comme un style architectural pour le système d'information de l'entreprise grâce à son concept pivot : le service.

Récemment, les **services Web** ont émergé pour proposer des solutions d'intégration des applications d'entreprises. Les services Web constituent l'instanciation la plus importante du modèle SOA dans le domaine industriel. En effet, les entreprises encapsulent les tâches automatiques comme des services logiciels pour les rendre visibles sur Internet. [12]

Dans ce chapitre nous voulons présenter la définition de l'architecture orienté service et les différents concepts qui lui sont liées, ainsi que les éléments de base qui la constituent, et finalement nous détaillons un peu le concept de service.

2. Définitions :

2.1. L'Architecture Orientée Service (Service Oriented Architecture) SOA :

La SOA est un ensemble de principes architecturaux qui permettent de développer des systèmes modulaires basés sur des « services » ou des unités de fonctionnalités informatiques. Ces services, qu'ils soient métiers ou techniques, sont offerts par une entité, le prestataire de services, et consommés par une autre. [13]

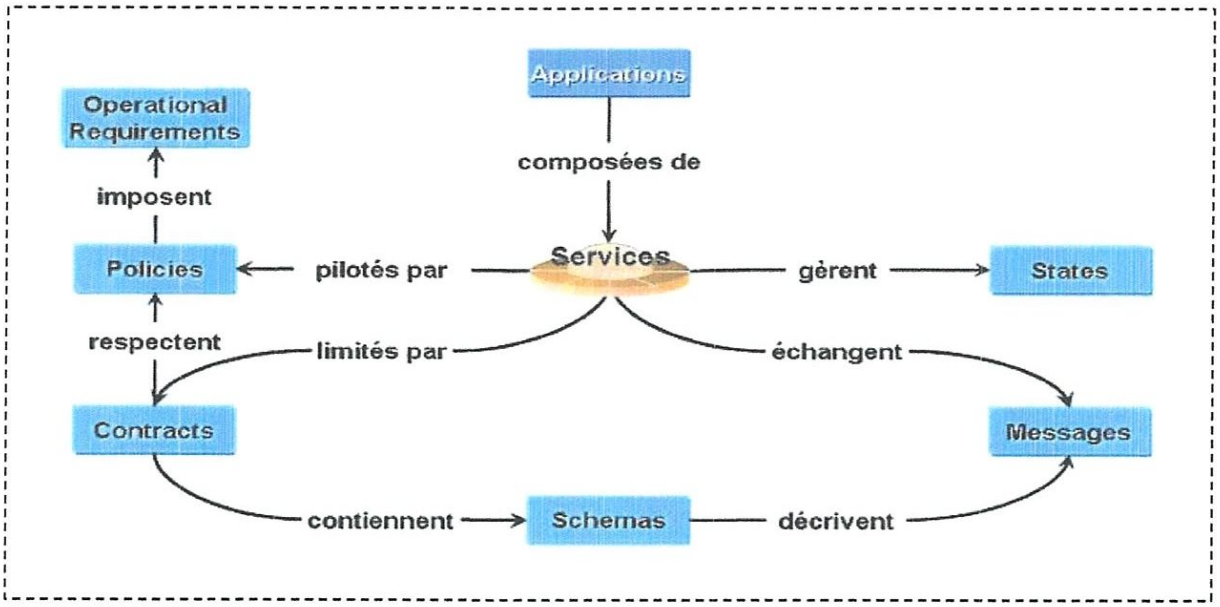


Figure 1.1 : *Concepts relatifs à la SOA.* [13]

2.2. Service :

Le service est l'unité atomique d'une architecture SOA. Les Services sont des fonctions clairement définies, indépendantes de la plateforme technique et de l'état des autres services. Cette collection de services hétérogènes, interopérables et faiblement couplés permet de créer et d'exécuter des applications composites selon les processus métiers de l'entreprise. [3]

2.3. Service Web :

La technologie des services Web est un moyen rapide de distribution de l'information entre clients, fournisseurs, partenaires commerciaux et leurs différentes plates-formes. Les services Web sont basés sur le modèle SOA. [W7]

Les Web Services représentent l'implémentation la plus utilisée pour appliquer l'architecture SOA [W5]. Les WS constituent la meilleure solution standardisée disponible [W6]

2.3.1. Définition de service web:

Selon W3C [1]: « Un Service web est un système logiciel conçu pour permettre l'interopérabilité machine-à-machine sur un réseau. Un service web est un système logiciel identifié par URI, dont les interfaces publiques sont définies et décrites en utilisant un langage dérivé de XML. Sa définition peut être découverte par d'autres systèmes logiciels. Ces systèmes peuvent alors interagir avec le service web d'une

[1] **W3C** : Consortium géré par le MIT, l'INRIA et Keio University au Japon. Ce consortium a un rôle leader en ce qui concerne la définition des standards de l'Internet. www.w3c.org

manière prescrite par sa définition, en utilisant des messages basés XML véhiculés par des protocoles d'Internet». [8]

3. Propriétés de la SOA :

Le principe d'une architecture orientée services consiste à structurer le système d'information de l'entreprise comme un ensemble de services qui exposent leur interface fonctionnelle et qui communiquent par messages.

Parmi les caractéristiques que nous jugeons fondamentales de la SOA nous citons :

Couplage faible ou lâche	Le but du principe de couplage faible entre les services est de garder une faible liaison entre les services, pour atteindre un haut degré de flexibilité dans l'architecture. [1]
La cohésion forte	La cohésion dans les services adresse le degré de relation fonctionnelle et sémantique qui existe entre les opérations accomplies par un service. Une forte cohésion d'un service implique que les opérations de ce dernier sont fortement reliées entre elles. [1]
Interopérabilité	être codé dans n'importe quel langage ; s'exécuter sur n'importe quelle plate-forme (matérielle et logicielle). [1]
Autonome	Le terme « autonome » se réfère à la capacité à prendre des décisions ou exécuter des tâches sans appui extérieur. [1]
Stateless (Sans état)	ne dépend d'aucun contexte ou service externe. [1]
Contrat standardisé	respecter un ensemble de contrats (règles de fonctionnement) Accord sur contrat légal entre le fournisseur et le client d'un service, notion de Courtage, Négociation. [1]
Granularité	définie la complexité et le nombre de fonctionnalités fournies par un service, il faut faire un compromis entre granularité fine et grosse. [1]
Découverte dynamique	Un service est complété par un ensemble de métas donnés de communication au travers desquelles il peut être découvert et interprété de façon effective [w9].
Réutilisabilité	La réutilisation des services indique la possibilité de réutiliser des composants dans plusieurs scénarios pour atteindre plusieurs buts. La caractéristique de la réutilisation des services permet de baisser les coûts de maintenance de l'architecture. [1]
Composabilité	Un service doit être conçu de façon à participer à des compositions de services [w9].

Tableau 1.1 : *Les caractéristiques du service.*

4. Les avantages et les limites de la SOA :

4.1. Les avantages :

- L'obligation d'avoir une modélisation poussée.
- Possibilité de découpler les accès aux traitements.
- Localisation et interfaçage transparents (ouverture accrue).
- Possibilité de mise en place facilitée à partir d'une application objet existante.
- Réduction des coûts en phase de maintenance et d'évolution.
- Facilité d'amélioration des performances pour des applications importantes (répartition des traitements facilitée). [3]

4.2. Les inconvénients :

- Coûts de conception et de développement initiaux plus conséquents.
- Nécessité d'appréhender de nouvelles technologies.
- Existant non SOA dans les entreprises.
- Performances réduites pour des traitements simples (couche supplémentaire). [3]

5. Les acteurs de la SOA :

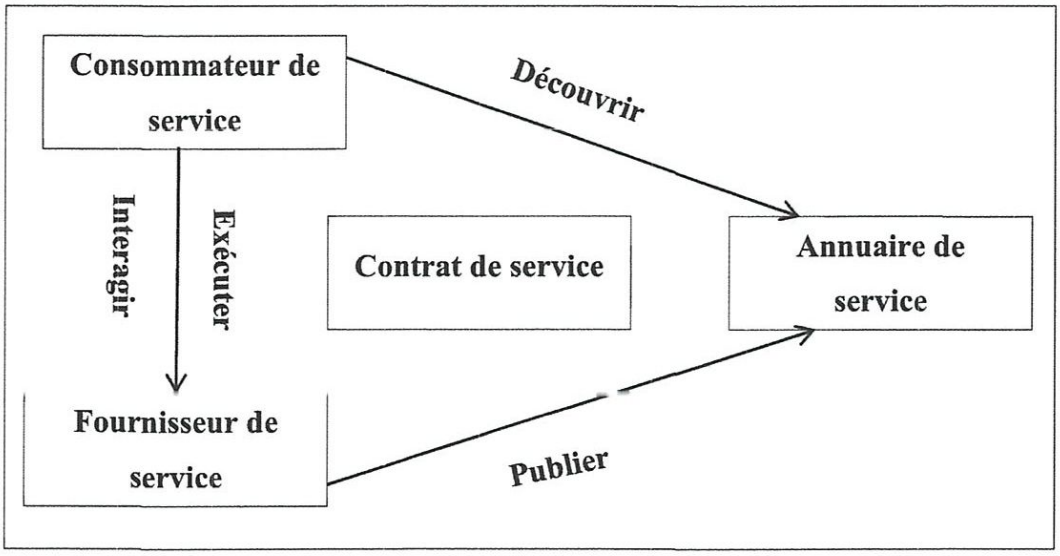


Figure 1.2 : *les acteurs de la SOA. [3]*

5.1. Le consommateur de service :

Le consommateur de service est une application qui requière un service. C'est l'entité qui initie la localisation du service dans l'annuaire, interagit avec le service à travers un protocole et exécute la fonction exposée par le service. [3]

5.2. Le fournisseur de service :

Le fournisseur de service est une entité adressable via un réseau, il accepte et exécute les requêtes venant d'un client. Le fournisseur de service publie le contrat de service dans l'annuaire pour qu'il puisse être accédé par les clients. [3]

5.3. L'annuaire de service :

L'annuaire de service est un annuaire qui contient les services disponibles. C'est une entité qui accepte et sauvegarde les contrats du fournisseur de service et présente ces contrats aux éventuels clients. [3]

5.4. Le contrat de service :

Le contrat spécifie la manière dont le client de service va interagir avec le fournisseur de service. Il spécifie le format de la requête et la réponse du service. [3]

6. Les principes de base de la SOA:

- ***Diviser pour régner :***

Substituer la découpe applicative strictement par une structuration en composants plus réduits et potentiellement plus simples à faire évoluer. [3]

- ***Alignement métier :***

Construire et organiser le système à partir des réalités métiers, qui doivent se retrouver dans ses constituants. [3]

- ***Neutralité technologique :***

Assurer une indépendance totale entre les interfaces et les implémentations. L'élément qui utilise un service ne doit pas être contraint ni par la technologie d'implémentation, ni par sa localisation (potentiellement distribué). [3]

- ***Mutualisation :***

Favoriser la réutilisation de services métiers par plusieurs lignes métiers ou applications. Permettre la construction de services de haut niveau par combinaison de services existants. [3]

- ***Automatisation des processus métier*** :

Isoler la logique des processus métiers sur des composants dédiés qui prennent en charge les enchaînements de tâches et les échanges de flux d'information. [3]

- ***Echanges orientés Document :***

Les informations échangées par les services possèdent une structure propre, guidée par les besoins métiers. On privilégie la transmission de contenus complets et utilisables au profit d'accès direct aux structures de type objet ou relationnel. [3]

7. Les Standards de la SOA :

La SOA s'appuie sur des standards techniques aujourd'hui matures :

7.1. Langage XML (Extensible Markup Langage) :

XML constitue la technologie de base des architectures Web services. Il s'agit d'un format générique qui est extensible dans des syntaxes propres à n'importe quelle utilisation. XML est un standard, format universel qui permet de décrire des documents structurés transportables sur les protocoles d'Internet. Il apporte à l'architecture des Web services l'extensibilité et la neutralité vis à vis des plates-formes et des langages de développement. [3]

7.2. Protocole SOAP (Simple Object Access Protocol):

- ✓ SOAP est un protocole de transmission de messages.
- ✓ Il définit un ensemble de règles pour structurer des messages principalement pour exécuter des dialogues requéte-réponse de type RPC (Remote Procedure Call). [W8]

7.2.1. Structure d'un message SOAP :

Un message SOAP est un document XML ordinaire qui contient les éléments suivants :

- ***L'élément Envelope*** : qui identifie le document XML comme étant un message SOAP.
- ***L'élément Header*** : qui est optionnel et qui contient des informations d'entête.
- **L'élément *Body*** : qui contient l'appel ainsi que la réponse retournée.
- ***L'élément Fault*** : qui est optionnel et qui fournit des informations sur d'éventuelles erreurs survenues lors de l'analyse du message. [3]

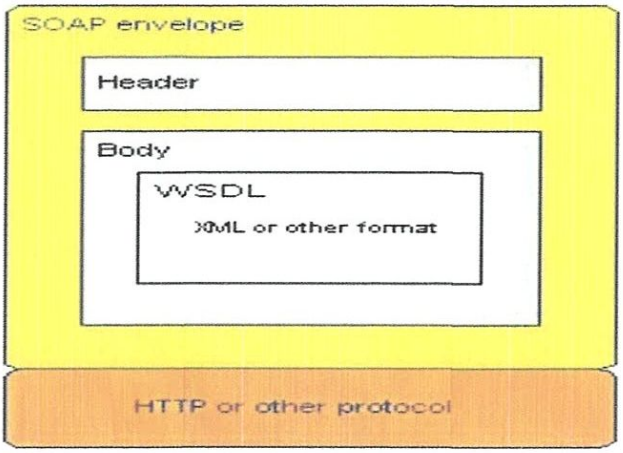


Figure 1.3 : *Architecture SOAP.* [3]

7.3. Norme UDDI (Universal Description, Discovery and Integration):

Un registre **UDDI** est un annuaire basé sur XML qui permet de publier des services et faciliter leur découverte par d'autres services en définissant comment ils interagissent. Un scénario d'utilisation possible est donc la publication d'un fournisseur de service (donc de son WSDL) auprès d'un registre en créant tout d'abord une entreprise et une catégorie de service. Un client demande ensuite à un registre UDDI la localisation d'un service qui correspond au service venant d'être ajouté à l'annuaire. Le WSDL du service demandé est alors reçu par le client qui peut communiquer avec le fournisseur de services en SOAP. [W5].

7.4. Langage WSDL -Web Service Description Language :

Une interface qui cache le détail de l'implémentation du service, permettant une utilisation indépendante [W8] :

- De la plate-forme utilisée.
- Du langage utilisé.

Le fichier WSDL est au format XML et regroupe toutes les informations nécessaires pour interagir avec le Web Service [W8] :

- Les méthodes.
- Les paramètres et valeurs retournées.
- Le protocole de transport utilisé.
- La localisation du service.

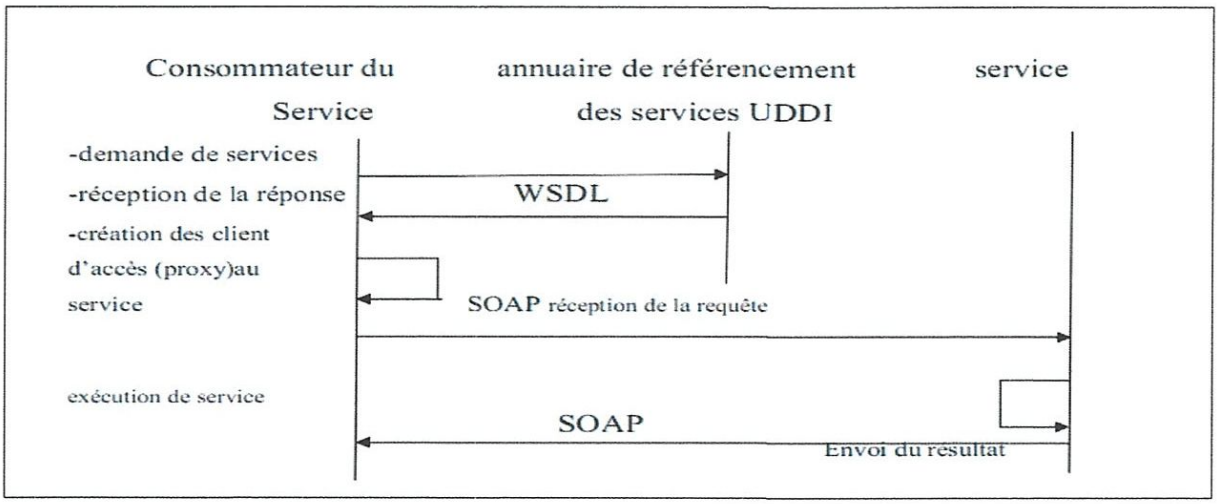


Figure 1.4 : *flux WSDL et SOAP dans l'utilisation de services [3].*

8. Le fonctionnement dans une architecture de services web :

Le fonctionnement des services Web s'articule autour de trois acteurs principaux illustrés par le schéma suivant [W7]:

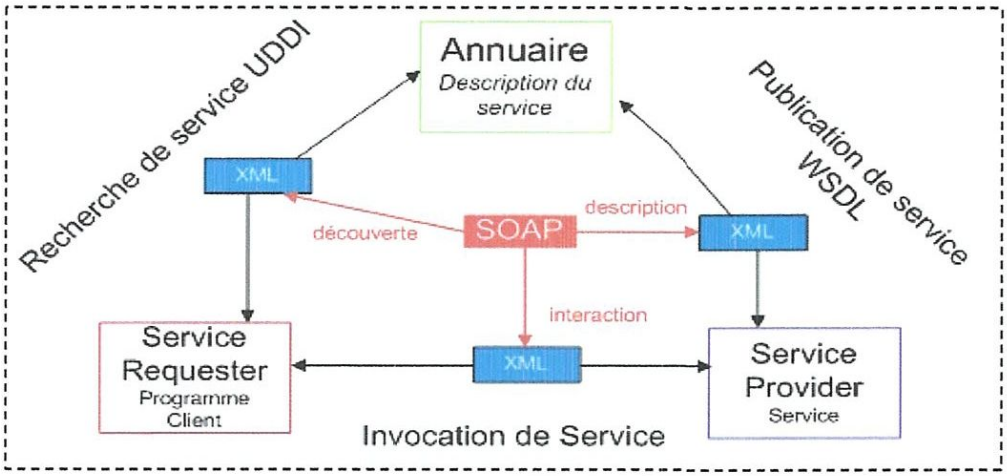


Figure 1.5 : *Le fonctionnement des services Web [W7].*

Une architecture de services web fonctionne de la manière suivante [14]:

- Le client envoie une requête à l'annuaire de Service pour trouver le service Web dont il a besoin.
- L'annuaire cherche pour le client, trouve le service Web approprié et renvoie une réponse au client en lui indiquant quel serveur détient ce qu'il recherche.
- Le client envoie une deuxième requête au serveur pour obtenir le contrat de normalisation de ses données.
- Le serveur envoie sa réponse sous la forme établie par WSDL en langage XML.

- Le client peut maintenant rédiger sa requête pour traiter les données dont il a besoin.
- Le serveur fait les calculs nécessaires suite à la requête du client, et renvoie sa réponse sous la même forme normalisée.

9. Les différentes approches d'identification de service:

Le processus de modélisation et de conception axée sur le service se compose de trois étapes générales : l'identification, la spécification et la réalisation des services [16]. L'étape d'identification vise à déterminer quels services sont appropriés pour être implémentés dans une architecture axée sur les services. Dans cette phase, les services sont nommés services candidats. Au cours de l'étape de la spécification, l'architecture du service est conçue et son interface, ses messages et ses événements sont détaillés. Enfin, le service est codifié et testé dans l'étape de réalisation. L'étape d'identification, qui est la portée de ce travail, est l'un des principaux défis dans la conception et la mise en œuvre d'un système axé sur les services [17]. Ce défi consiste à prévoir quels services une entreprise aura éventuellement besoin et à définir les fonctions qui devraient faire partie de chaque service. [19]

Au cours des dernières décennies, plusieurs méthodes d'identification des services ont été proposées, mais il n'y a pas de consensus sur la «meilleure méthode» ni sur une approche prédominante pour identifier les services candidats. Les méthodes d'identification des services (SIM) sont classées dans trois stratégies d'identification possibles: Top down, Bottom-up et Meet in the middle [18] [19].

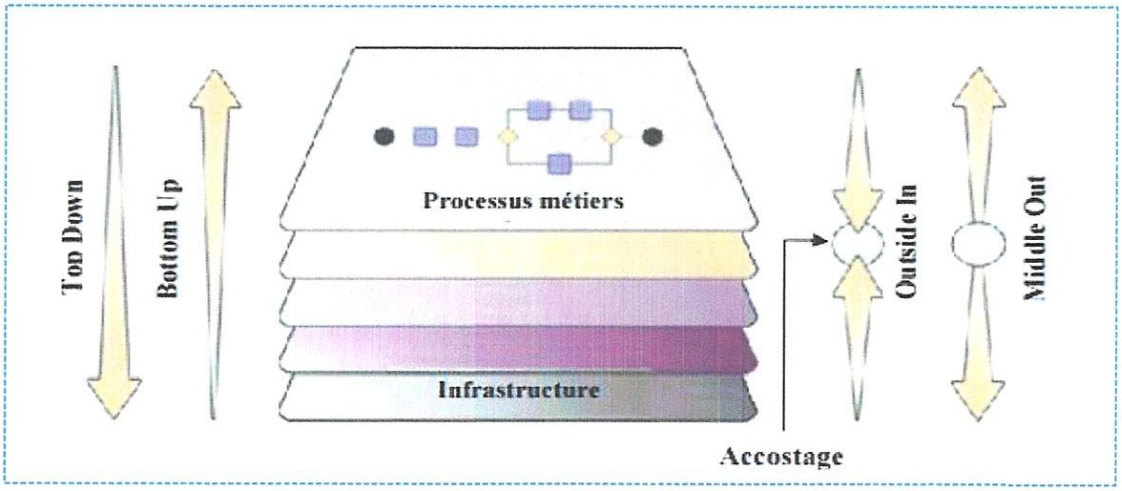


***Figure 1.6** : Différentes approches d'identification de services.* [6]

9.1. Approche Top Down :

Le point de départ de cette approche est la définition et la formalisation des processus métiers, qui représentent les objectifs fonctionnels de l'entreprise, pour descendre ensuite au travers des différentes couches du système afin de définir les services nécessaires à la réalisation de ces processus.

Cette approche représente **la voie royale** :

- Elle permet de piloter la SOA par les besoins métiers.
- Elle minimise la redondance de services.

Elle n'est cependant que **rarement possible** car :

- Elle est très coûteuse, très longue (plusieurs années) et trop risquée, puisque elle cause la reconstruction de tout ou d'une partie de système d'information.

9.2. Approche Bottom Up :

L'approche "**Bottom Up**" démarre par une phase d'analyse de l'existant afin de déterminer les fonctions transverses existantes dans le système. Ensuite, ces fonctions sont mises en mode service, pour qu'ils soient exploitables au sein d'un autre service et/ou un processus métier. [6]

Cette approche est intéressante :

- puisque elle contraint à réaliser une cartographie du système, ce qui facilite la publication et la réutilisation de ses fonctionnalités intéressantes. [6]

Cette approche présente un inconvénient majeur :

- Est qu'elle ne prend pas en charge les nouveaux besoins de l'entreprise, dans la mesure où elle se limite à encapsuler les fonctions existantes au niveau du système dans des services, qui restent très fortement couplés à leur application d'origine. [6]

9.3. Approche Outside In :

L'approche "**Outside In**" propose de démarrer en parallèle, l'approche Top Down "sans se préoccuper de l'existant", pour définir les besoins métiers en processus métiers et les services nécessaires à leur réalisation.

Et l'approche Bottom Up "en ne considérant que l'existant", afin de cartographier l'existant applicatif dont dispose l'entreprise pour supporter les services métiers à forte valeur ajoutée métier.

Une fois ces deux chantiers en phase finale, commence l'étape de l'accostage. Son objectif est de réconcilier les résultats des deux approches afin de déterminer comment seront réalisés les processus métiers. Il faut, pour cela, comparer les besoins en services exprimés par l'approche **Top Down** avec ceux remontés de l'approche **Bottom Up**. [6]

9.4. Approche Middle Out :

Par opposition à l'approche **Outside In**, cette méthode propose de commencer au milieu, c'est-à-dire le métier et les technologies d'information IT (Information Technology) parlent le même langage.

Elle s'attaque donc d'emblée à ce qui reste un des facteurs limiteurs à l'adoption des SOA : La compréhension du métier de l'entreprise par les maîtrises d'œuvre et inversement, la compréhension des contraintes IT par les maîtrises d'ouvrage. Une fois les différentes parties sont d'accord sur un premier socle de services "métiers" nécessaires : [6]

- Les maîtrises d'ouvrage engagent un chantier "**Middle Up**" pour spécifier les processus métiers.
- Les maîtrises d'œuvre engagent un chantier "**Middle Down**", pour spécifier le socle de services de plus bas niveau permettant la réalisation des services métiers. Cette approche limite par contre le pilotage de la SOA par les besoins métiers, puisque le point de départ est l'identification des services métiers nécessaires et non la définition des processus réalisant le métier de l'entreprise.

10. Conclusion :

L'utilisation de la SOA a donc pour but d'avoir une vision globale du système d'information d'une entreprise.

Elle représente la capitalisation des ressources métier d'une entreprise avec la réutilisation des services et le stockage des données sur une base de données unique. L'utilisation d'une telle architecture facilite l'évolution des applications mais demande beaucoup d'efforts au niveau conception.

Dans le contexte de notre travail nous nous somme intéressé à l'approche Top-Down où nous essayons d'automatiser l'identification des services à partir du point de vu de

l'entreprise qui est représenté par les diagrammes BPMN. Donc nous prenons les processus métier décrits par l'entreprise comme point de départ.

L'approche processus est l'opération qui consiste à la modélisation et l'automatisation des processus d'entreprise.

L'émergence de l'approche processus, et l'architecture orientée services (Service Oriented Architecture – SOA-) a été accompagnée par la promesse de la mise en place d'une entreprise efficace, agile et interopérable.

Le chapitre suivant discute des différentes notions attenantes à l'approche processus.

Chapitre 02 :

Approche processus

1. Introduction :

L'approche de l'organisation est profondément modifiée par la vision processus. Celle-ci a d'abord touché la production industrielle avec la théorie du juste à temps et s'est ensuite étendue à toutes les activités avec les principes de la qualité totale. L'approche par les processus, telle que normalisée par ISO9000 ou appliquée par ISO100006 pour la gestion des processus uniques, conduit à une représentation d'une organisation ou d'un projet sous forme d'un système de processus. [9]

Le terme ***processus*** vient du latin et signifie littéralement «aller de l'avant ». Dans un usage général, il évoque l'idée d'une marche progressive, souvent selon un plan déterminé à l'avance. [9]

Cette idée de plan a été reprise depuis longtemps dans les systèmes d'information. Dès les années 70, dans sa **méthode** BSP (Business System Planning) pour la planification des systèmes d'informations, IBM définissait un **processus** de gestion comme un ensemble organisé d'activités pour la production d'un résultat. La **méthode** Merise a considéré le **processus** comme la réaction de l'entreprise à un type d'événement. Plus récemment, le courant de la reconfiguration des **processus** définit le **processus** comme un ensemble finalisé d'activités, orienté vers la production d'un résultat représentant une valeur pour un client.

Le terme « **processus** » occupe de plus en plus une place importante dans le discours traitant des systèmes d'informations, de la modélisation et de l'ingénierie des systèmes en général.

Les **processus** deviennent l'élément fondamental pour l'analyse, et ce, quel que soit le domaine traité. [9]

Dans ce chapitre, nous allons d'abord définir un processus d'une manière générale, évoquer les différents types de processus qui existent. Ensuite, nous allons présenter l'approche processus, le rôle qu'elle joue au sein de l'organisation, et enfin les modèles de représentation des processus.

2. Approche Processus :

2.1. Définition :

- **L'approche processus** désigne l'application d'un système de processus au sein d'un organisme, ainsi que l'identification, les interactions, le pilotage et le management de ces processus.

Approche processus = une organisation maîtrisée. [10]

- **L'approche processus** est une méthode de modélisation des activités de l'entreprise. Décrire son fonctionnement, c'est en comprendre les mécanismes et les interactions ; préalable nécessaire à toute démarche de progrès. [11]

2.2. Pourquoi l'approche processus ? [W4]

- Satisfaire aux exigences des clients et/ou autres parties intéressées
- Avoir une vision transversale et globale
- Maitriser également les interfaces entre les activités
- Inciter les acteurs à travailler vers un objectif commun et partagé
- c'est un outil de modélisation qui facilite le management et le pilotage
- c'est une approche qui part des besoins du client
- c'est un des éléments centraux de la norme ISO 9001 version 2000 :

La norme demande :

- d'identifier les processus nécessaires au fonctionnement de votre entreprise
- d'assurer le bon fonctionnement *et l'amélioration continue* des processus identifiés (Dont la capacité à atteindre les objectifs liés à la satisfaction des clients)

Réussir son approche processus = réussir sa démarche qualité.

2.3. Principales étapes d'une approche processus:

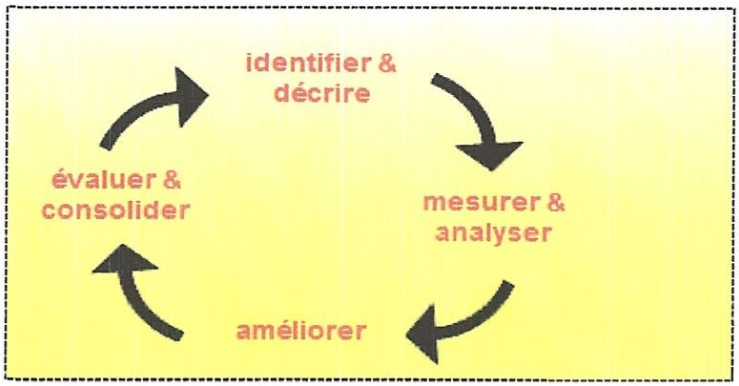


Figure 2.1 : *Les étapes de l'approche processus. [W4]*

4 étapes de l'approche processus [W4]:

2.3.1. Identifier et décrire les processus :

- faire la cartographie des processus
- décrire chaque processus

2.3.2. mesurer et analyser les processus :

- mettre en place des indicateurs
- réalisé la surveillance
- évalué la maturité

2.3.3. Amélioration :

- fixer des priorités.
- identifier des actions d'amélioration.
- réaliser les actions d'amélioration.

2.3.4. évaluer et consolider :

- évaluer *l'efficacité* des actions
- généralisé les solutions

2.3. Les avantages de l'approche processus: [W4]

- ✓ Prend en compte la valeur-ajoutée d'une activité.
- ✓ Axée sur la performance.
- ✓ Stimule la dynamique fournisseur-client interne.
- ✓ Plus logique, compréhensible et donc mieux acceptée par les collaborateurs.
- ✓ La documentation qualité est bien structurée et light.
- ✓ Elle réfléchit la structure opérationnelle de l'entreprise.
- ✓ Est un outil de gestion de l'entreprise
- ✓ comprendre et de satisfaire les exigences.
- ✓ considérer les processus en termes de valeur ajoutée
- ✓ mesurer la performance et l'efficacité des processus.
- ✓ Améliorer en permanence les processus sur la base de mesures fréquentielles et objectives.

3. Les processus :

3.1. Définition :

- *ISO 9000:2000* : **Un processus** est un ensemble d'activités corrélées ou interactives qui transforment des éléments d'entrée en éléments de sortie. Ces moyens peuvent inclure le personnel, les finances, les installations, les équipements, les techniques et méthodes. [10]
- **Un processus** :est un ensemble d'activités ayant un déclencheur commun, reliées entre elles par des flux d'information ou de matière significatifs et qui se combinent pour fournir un produit matériel ou immatériel, important et bien défini que l'on peut rattacher à un client externe ou interne. [9]

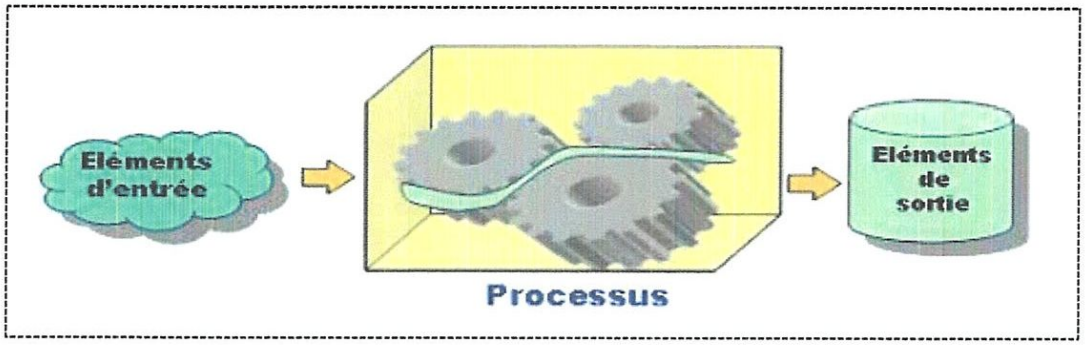


Figure 2.2 : *Représentation graphique d'un processus. [9]*

3.1.1. Activité :

Une activité est " un ensemble de tâches élémentaires réalisées par un individu ou un groupe, par une machine ou un groupe de machines, avec un objectif bien précis (out put commun aux tâches) ".

Une activité transforme des entrées en sorties par l'influence d'objets de contrôle et en utilisant les ressources requises et disponibles pendant une durée bien définie. [9]

3.1.2. Tâche :

Une tâche est "une opération, un travail à accomplir, par exemple, écrire une note, répondre au téléphone, préparer une commande, vérifier un tarif, mettre à jour un fichier … Les tâches élémentaires servent à préciser le contenu d'une activité ".

Il faut préciser donc qu'une tâche est exécutée par un seul acteur. [9]

3.2. Les types de processus :

Certainement nous pouvons définir 3 catégories de processus comme montre la figure suivante :

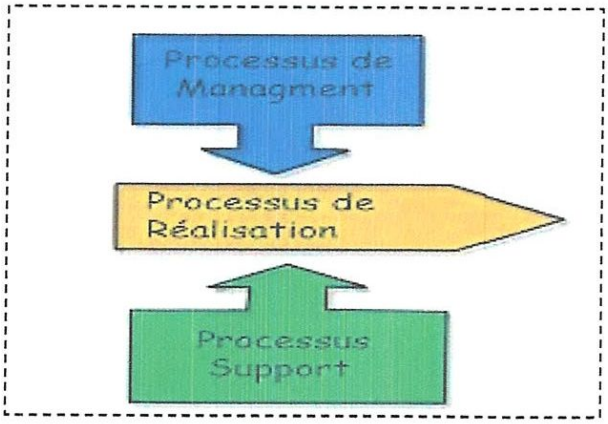


Figure 2.3 : *Les types de processus. [W2]*

3.2.1. Les processus de support :

Les processus de support (soutien) fournissent les ressources nécessaires au bon fonctionnement de tous les autres processus. Ils ne sont pas liés directement à une contribution de la valeur ajoutée du produit mais sont toujours indispensables. [W2]

3.2.2. Les processus de management :

Aussi appelés de **direction**, de **pilotage**, de **décision**. Ils participent à l'organisation globale, à l'élaboration de la politique, au déploiement des objectifs et à toutes les vérifications indispensables. Ils sont les fils conducteurs de tous les processus de réalisation et de support. [W2]

3.2.3. Les processus de réalisation (métier) :

Les processus de réalisation sont liés au produit, augmentent la valeur ajoutée et contribuent directement à la satisfaction du client.

Ils sont composés d'un enchainement d'activités ou d'ensembles d'activités, alimentés par des entrés et consomment des ressources, qui créent des sorties en y apportant une valeur ajoutée. [W2]

3.3. L'interaction entre les trois types de processus :

La figure ci-dessous symbolise l'interaction entre les trois types de processus vue précédemment :

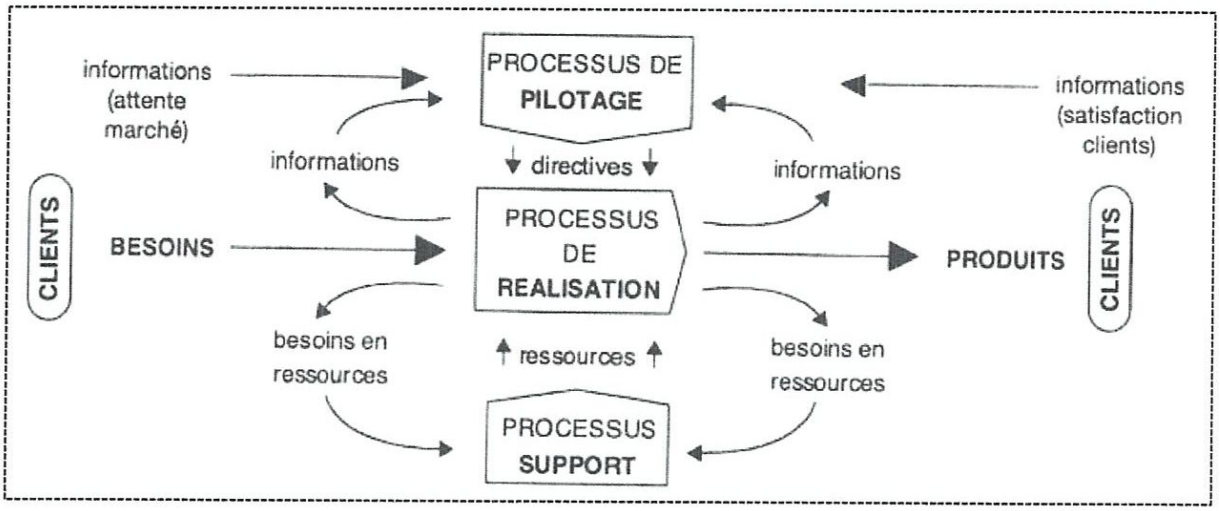


Figure 2.4 : *L'interaction entre les trois types de processus. [11]*

Comme montre la figure précédente, le processus métier est le responsable à la fourniture des produits, en consommant certain besoin, en utilisant les ressources fournies par le processus du support et sous la direction du processus de pilotage. [11]

3.4. La modélisation du processus :

La modélisation des processus métier est au cœur même de la démarche d'analyse dynamique d'une organisation. Que ce soit dans le cadre d'une démarche d'amélioration ciblée ou d'une réorganisation plus globale, la modélisation des processus permet de formaliser le fonctionnement précis d'une organisation en utilisant un langage standard et aisément compréhensible.

La richesse sémantique, offerte par les techniques et outils de modélisation organisationnelle de l'entreprise, facilite ainsi une perception commune des processus métiers orientée "amélioration" ponctuelle ou continue. [W3]

3.4.1. Techniques de modélisation des processus métiers :

Durant les dernières années, la modélisation des entreprises a ete un domaine de recherche très actif. En réalité les gens modélisaient les processus depuis plusieurs années, même s'ils n'appelaient pas leurs modèles des « modèles de processus métiers ». [11]

3.4.2. Notion d'une méthode de modélisation :

Une **méthode** est un ensemble comprenant un langage, souvent présenté sous forme d'un ensemble de modèles et diagrammes associés, ainsi que des préconisations sur la façon d'utiliser ces modèles. [11]

Un **langage de modélisation** est un ensemble de concepts et de règles permettant de construire des modèles. Les éléments composants un langage de modélisation peuvent être représenté par un modèle, appelé méta-modèle. [11]

Un **modèle** est une représentation simplifiée de tout ou une partie d'un système d'information existant ou futur, mettant en évidence certains aspects essentiels. Pour élaborer un modèle, on s'appuie sur un méta-modèle. On peut suivre des règles de construction, lorsqu'elles ont été énoncées. [11]

Un **diagramme** correspond à la forme graphique d'un modèle. [11]

3.5. Les différentes méthodes de modélisation :

Dans le monde des entreprises, il existe de nombreuses méthodes qui permettent la modélisation des processus métier. Devant ce large choix nous avons choisi d'en présenter certain :

3.5.1. Business Process Management (BPM) :

La BPM est défini par une application informatique qui permet l'intégration des données, des gens et des applications à travers un processus métier commun. C'est une approche globale de la gestion des processus d'entreprise. Elle couvre la modélisation, l'informatisation, l'exécution, l'administration et l'optimisation des processus d'entreprises. [5]

Les objectifs des BPM sont: [5]

- Automatiser les processus métier de l'entreprise ;
- Accélérer grandement les étapes de décisions en apportant en temps réel les informations pertinentes aux bonnes personnes ;
- Capitaliser sur les applications du système d'information.

3.5.2. Process Definition Language (XML):

La définition d'un processus comporte les principaux éléments suivants, sous forme de balises: [5]

- ✓ les marques de début et de fin du ou des processus
- ✓ les activités
- ✓ leurs interrelations (les transitions)
- ✓ les attributs qualifiant certains comportements de l'activité
- ✓ les participants / rôles / groupes
- ✓ les interactions / relations entre les acteurs et les activités
- ✓ ...

3.5.3. Business Process Execution Language (BPEL):

BPEL est exécutable sur des moteurs BPEL comme ODE (Orchestration Director Engine) d'Apache. Le langage BPEL – Business Process Execution Language – a été lancé à l'initiative de Microsoft et IBM en réponse à l'initiative de BPMI. Depuis, ce langage a reçu le support de la plupart des acteurs du marché. BPEL est devenu le standard. Il vient en complément de la spécification sur les services Web. Depuis 2003, c'est l'organisme de standardisation OASIS qui a en charge l'évolution du langage BPEL.

Il est possible de modéliser un "Business Process" en utilisant BPMN et de le convertir en BPEL, puis de déployer le BPEL sur un moteur BPEL. [5]

3.5.4. Unified Modeling Language (UML)

Notation permettant de modéliser un problème de façon standard. Ce langage est né de la fusion de plusieurs méthodes existant auparavant, et est devenu désormais la référence en termes de modélisation objet. L'UML propose des diagrammes spécialisés (dont les diagrammes d'activité, de séquence, de classe etc.) ayant chacun une fonction précise. Il n'existe pour le moment pas de diagramme UML spécialisé pour la modélisation des processus. [5]

4. Business Process Modeling Notation (BPMN):

BPMN est constitué d'un ensemble d'éléments de modélisation. Ces éléments sont constitués d'un symbole (objet graphique sur un diagramme) et d'une liste d'attributs (la plupart du temps invisible sur un diagramme).

Remplir les attributs associés à un élément est généralement nécessaire dans une perspective d'implémentation. [5]

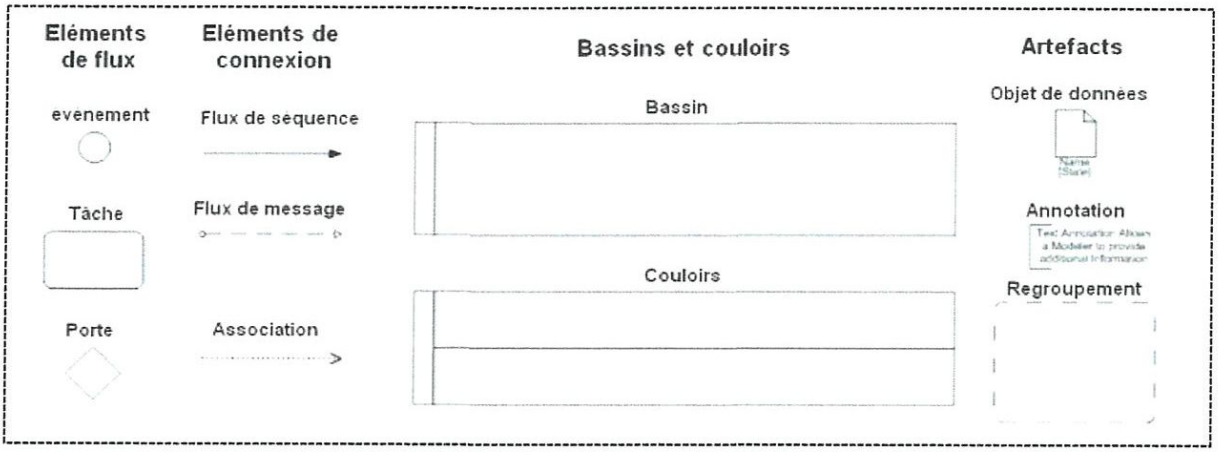


Figure 2.5 : *Les principaux éléments de BPMN. [5]*

4.1. Qu'est-ce que BPMN ?

Business Process Modeling Notation (BPMN) est une représentation graphique permettant de définir des processus métier dans un flux d'informations.

Business Process Modeling Notation (BPMN) est un standard pour la modélisation de processus métier qui fournit une notation graphique permettant de définir des processus métier dans un **diagramme de processus métier (BPM)**, basé sur une technique d'organigrammes très proche de celle utilisée par les diagrammes d'activité UML. [5]

4.2. Objectifs de BPMN :

L'objectif principal de BPMN est de proposer une notation standard facilement compréhensible par les partenaires professionnels. Ces partenaires incluent les analystes métiers qui créent et raffinent les processus, les développeurs techniques responsables de l'implémentation des processus, et les directeurs commerciaux qui suivent et gèrent les processus. Par conséquent, BPMN est prévu pour servir comme langage visant à combler un déficit de communication qui survient souvent entre le design des processus métier et l'implémentation. [5]

4.3. Les principaux éléments graphiques de BPMN :

4.3.1. Objets de flux :

4.3.1.1. Les évènements :

Un événement est représenté par un cercle. Les évènements servent à identifier un état particulier dans le processus. Il représente quelque chose qui survient, il déclenche alors une

action ou le résultat d'une action. Il n'effectue aucune tâche. Les évènements peuvent être de trois types : début, intermédiaire, fin. [5]

Les types d'évènements de sortie les plus importants sont :

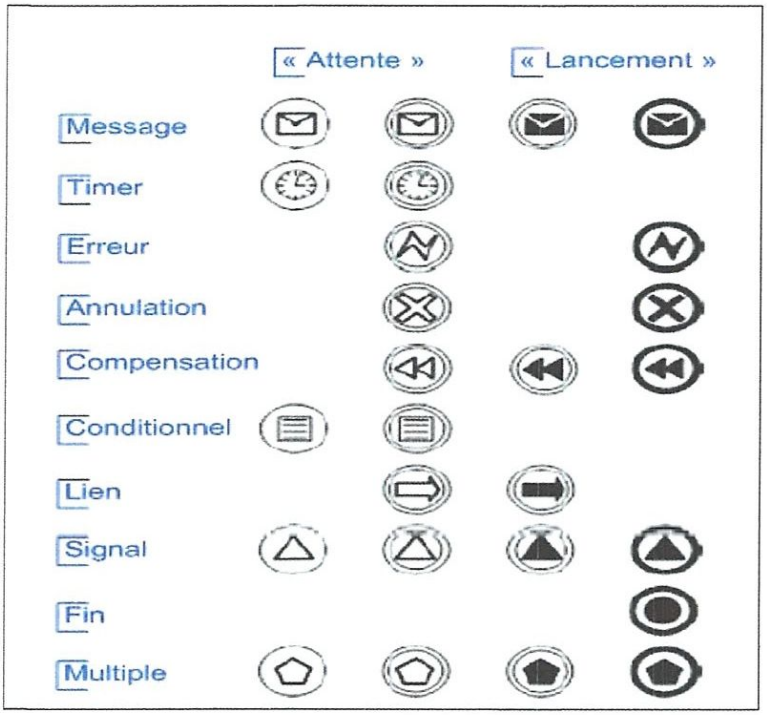


Figure 2.6 : *Les évènements BPMN. [5]*

4.3.1.2. Les activités:

Une activité peut être un processus, un sous-processus ou une tâche. Elle est représentée par un rectangle aux coins arrondis.

Une tâche est un élément indivisible. Elle représente une action. Chaque tâche a un début et une fin, par conséquent une tâche ne peut débuter que si la tâche précédente est terminée. [5]

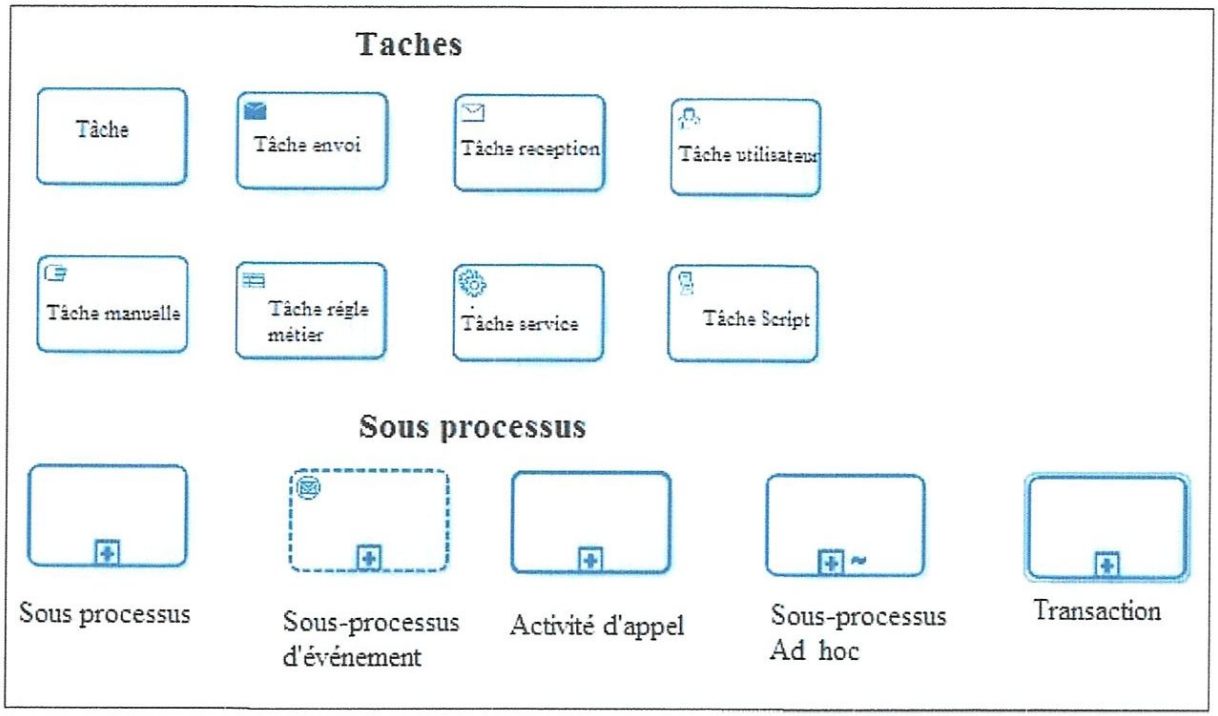


Figure 2.7 : *Type de Tâche.*

4.3.1.3. Les branchements :

Le branchement est un objet essentiel dans la norme BPMN. Il sert à représenter la condition de routage entre les flux d'entrée et les flux de sortie. Il est représenté comme une forme de losange. Le symbole à l'intérieur du losange sert à identifier le comportement du branchement.

Le branchement n'est pas une tâche et n'effectue aucune action. Il est donc possible de définir un comportement de traitement précis. [5]

Le tableau suivant montre les différents types de branchements de BPMN :

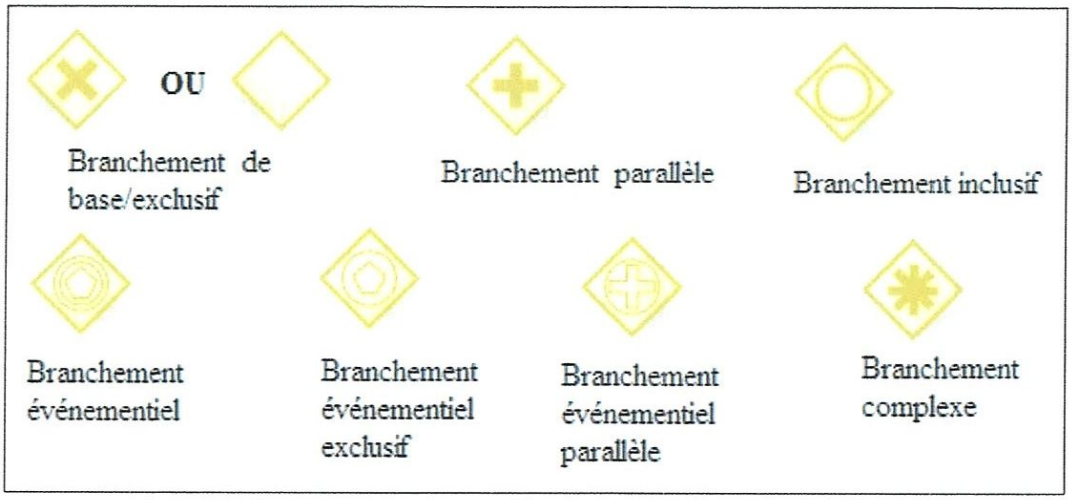


Tableau 2.8 : *Types de branchement. [5]*

4.3.2. Les objets de connexion :

Les objets de connexion sont utilisés pour connecter des objets de flux au sein d'un diagramme. [5]

Il existe différents types de connexions possibles :

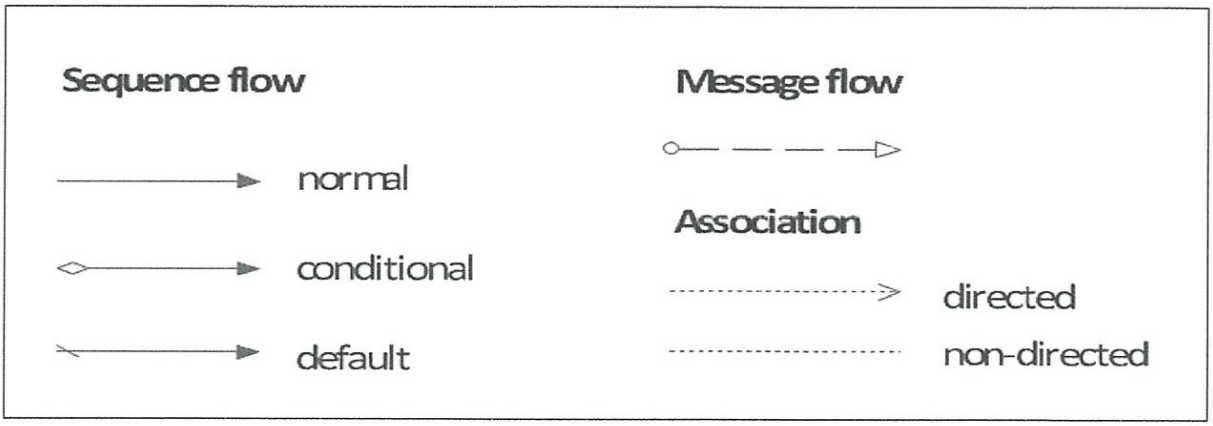


Figure 2.9 : *Les objets de connexion BPMN. [5]*

4.3.3. Les couloirs et les bassins (swimlanes) :

Un bassin appelé « Pool» est un conteneur graphique permettant de partitionner un ensemble de processus et de taches d'autres pools, généralement dans le contexte de situations business to business.

Un couloir est une sous-partition d'un pool et étend la longueur totale de ce dernier, verticalement ou horizontalement. Les couloirs permettent d'organiser et de catégoriser les activités d'un pool. [11]

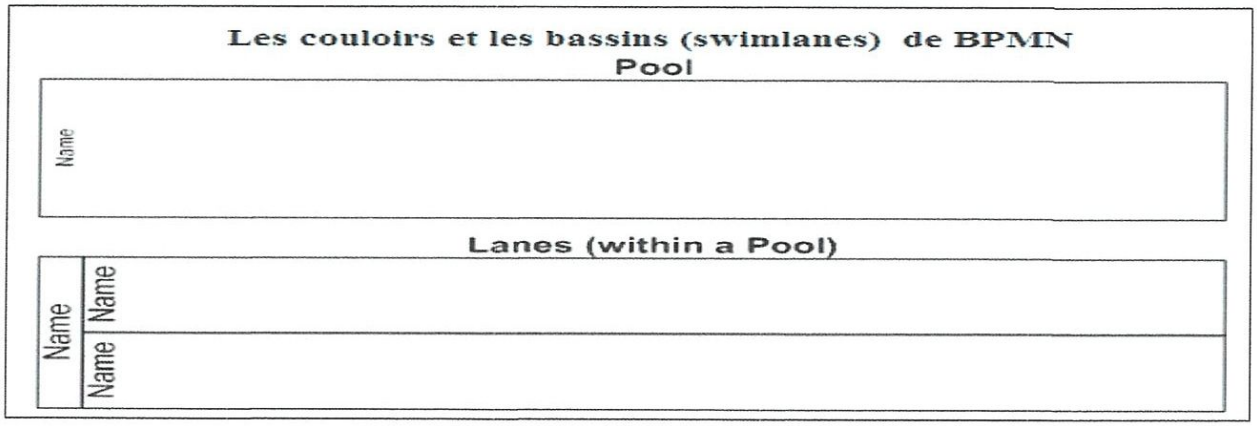


Figure 2.10 : *Les swimlanes BPMN. [5]*

4.3.4. Les artefacts :

Ce sont des objets additionnels utilisés pour mieux comprendre le schéma : comme les activités de même catégorie, les données traitées ou bien des commentaires ou annotations. Les artefacts sont utilisés pour ajouter plus d'information au diagramme ou pour effacer l'ambiguïté dans le diagramme. [5]

- ***Les objets de données:*** utilisés pour expliquer quelles données sont nécessaires dans le diagramme.
- ***Les groupes :*** utilisés pour des activités de groupes différents, sans affecter le flux dans le diagramme.
- ***Les annotations :*** utilisées pour donner plus d'informations et des commentaires sur le diagramme.

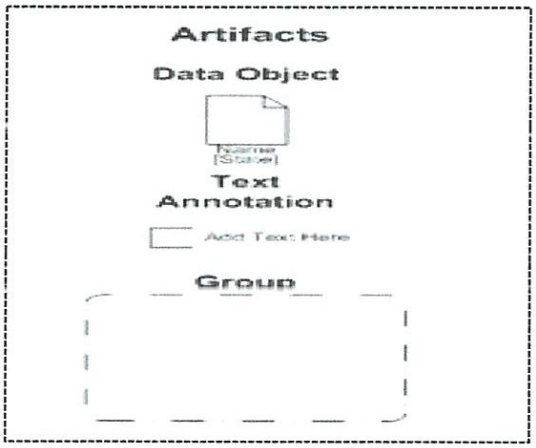


Figure 2.11 : *Les éléments annexes.* [5]

4.4. Les outils de modélisation des processus :

Les outils de modélisation des processus sont de plus en plus nombreux sur le marché en raison de l'augmentation des besoins d'automatisation des processus métiers. A l'aide de ces outils, les entreprises d'aujourd'hui peuvent plus aisément créer des modèles de leurs processus et suivre leur exécution. [11]

La figure suivante montre les outils les plus utilisé dans le domaine de modélisation des processus :

Figure 2.12 : *Les outils de modélisation des processus.*

5. BPM et SOA: concurrence ou synergie ?

Plusieurs travaux ont montré des synergies entre BPM et SOA. L'utilisation de SOA permet de découvrir des services réutilisables pouvant être orchestrés pour mettre en place des processus métiers dynamiques. Cette combinaison assure la conception itérative des processus basés sur des services qui pourront être changés rapidement [2].

La combinaison de BPM et SOA crée des opportunités en rendant visibles les processus de l'entreprise et en l'aidant à développer une infrastructure IT flexible. Cependant, l'alignement de ces deux concepts exige une profonde transformation des organisations [2].

D'une part, la démarche BPM ne fournit pas d'unité élémentaire de granularité suffisamment fine pour construire un système, alors que la notion de service offre un cadre d'architecture pour la conception de processus métiers en assurant la répartition des capacités de l'entreprise.

D'autre part, SOA peut produire des services réutilisables, mais sans garantie d'agilité métier [2].

Le but de SOA est de découvrir des services stables et réutilisables alors que les processus découverts avec BPM sont centrés sur les besoins du client (amenés à changer pour garantir l'agilité de l'entreprise). SOA promeut la consolidation des opérations redondantes et améliore la capacité d'adaptation aux changements métiers. [2]

Pour comprendre la synergie SOA-BPM, il faut répondre à une question essentielle à savoir : ***que représente un processus métier dans SOA ?*** Deux visions s'opposent. La première, souvent utilisée dans des approches techniques de SOA, positionne les processus métiers au-dessus des services métiers. La deuxième explique qu'un processus métier qui invoque un service fait lui-même partie d'un autre service métier [2].

6. Conclusion :

D'après ce chapitre, on peut conclure que l'approche processus a simplement pour but d'aider les différentes activités à se dérouler le mieux possible. Elle permet de décrire le fonctionnement de l'organisme d'une façon claire et stratégique à l'aide de certains formalismes comme la cartographie des processus et la carte d'identité des processus, en plus d'expliciter l'impact des processus sur l'atteinte des objectifs de l'entreprise et la satisfaction des clients. Plusieurs méthodes et outils permettent de modéliser les processus de l'organisation. [11]

BPMN est la notation standard la plus adaptée pour la gestion des processus, permettant de modéliser puis d'implémenter les processus d'une organisation et de développer des applications métier plus efficacement.

Dans notre cas nous allons utiliser cette notation de l'approche processus comme point de départ pour identifier les services potentiels qui peuvent être programmés et faciliter le passage vers une architecture orientée service en se basant sur le respect des caractéristiques de cette dernière (la SOA).

Dans le chapitre suivant nous allons détailler la problématique et les objectifs de notre projet ainsi que les solutions proposées.

Chapitre 03 :

Conception

1. Introduction :

Aujourd'hui, plusieurs entreprises se basent sur des systèmes d'information complexes, monolithiques, et inflexibles, parfois non conformes aux changements rapides du marché. L'Architecture Orientée Service (SOA) apparaît comme la meilleure approche permettant l'intégration flexible des applications autonomes, distribuées et hétérogènes au sein et au-delà de l'entreprise. [8]

Au cours des dernières décennies, plusieurs méthodes d'identification des services ont été proposées, mais il n'y a pas de consensus sur la «meilleure méthode» ni sur une approche prédominante pour identifier les services candidats. Les méthodes d'identification des services (SIM) sont classées suivant trois stratégies d'identification possibles : Top down, Bottom-up et Meet in the middle [18] [19].

Dans le contexte de notre travail nous nous somme intéressé à l'approche Top-Down où nous essayons d'automatiser l'identification des services à partir du point de vue de l'entreprise qui est représenté par les diagrammes BPMN. Donc nous prenons les processus métier décrits par l'entreprise comme un point de départ pour enfin arriver à identifier des services candidats qui peuvent être mis en œuvre suivant une architecture SOA.

2. L'objectif global :

L'objective de notre travail est d'automatiser la phase d'identification de service, qui est une étape incontournable dans n'importe quelle approche de mise en œuvre d'une application orientée service.

Nous avons choisi de suivre une approche top down, qui consiste à démarrer l'identification des services à partir du point de vue et des besoins de l'entreprise. Ces besoins sont modélisés suivant un BPM. Dans notre cas nous avons utilisé la norme BPMN2 qui est la représentation la plus standardisée.

Afin d'atteindre ce but nous avons dû nous focaliser sur plusieurs axes :

1- Proposer un ensemble de règles qui nous permettent d'extraire les services candidats à partir des représentations BPMN. Ces règles sont déduites à partir des propriétés et caractéristiques des services définies suivant une architecture orientée service, à savoir l'autonomie, la réutilisabilité, le couplage faible entre les services et la cohésion forte intra service. Nous adoptons comme définition au service candidat celle d'Erl [20] qui définit un service candidat en tant que service abstrait (non implémenté) qui, pendant

la phase de conception d'un cycle de vie du service, pourrait être choisi pour être implémenté comme un service ou comme une fonction d'application.

2- Développer un Framework (ensemble de composants logiciels cohérents) qui permet d'automatiser l'identification des services en appliquant les règles prédéfinis par défaut, mais aussi d'appliquer de nouvelles règles qui peuvent être enrichies au fur et à mesure par les experts.

3- Nous avons tenu à paramétrer notre Framework de façon à ce qu'il reste toujours flexible. Ainsi un gestionnaire de règle a été implémenté et qui permet d'introduire à travers une interface spécialisée de nouvelles règles formées à partir des artefacts d'un BPMN.

4- Une évaluation des services candidats identifiés a été proposée suivant deux façons

a) A travers une évaluation des règles d'identification proposée sous forme d'un taux de confiance introduit par les experts.
b) A travers le calcul des métriques qui nous permettent d'avoir une idée sur la qualité du service identifié (dans notre cas, calcul du couplage).

3. Processus d'identification des services :

Notre objectif est d'automatiser l'identification des services à partir du point de vue de l'entreprise représenté par les différents processus métiers qui seront formalisés dans une représentation en BPMN 2.0. Nous voulons décomposer l'ensemble des fonctionnalités fournies par cette représentation en un ensemble de fonctionnalités cohérentes basics appelées services. Dans notre cas une tâche ou bien un ensemble de tâches peuvent former un service à condition de respecter les propriétés d'un service.

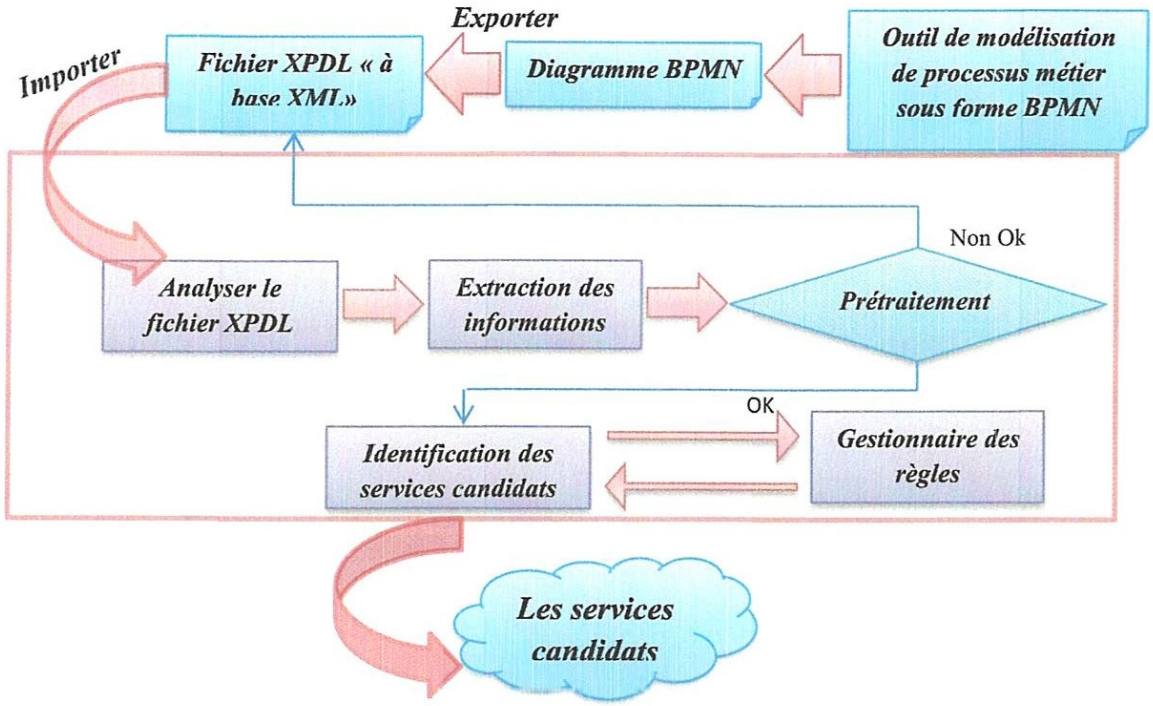


Figure 3.1 : *Processus général d'identification des services.*

3.1. La phase prétraitement des diagrammes BPMN:

Cette étape consiste à vérifier que le diagramme BPMN soit bien formulé et que toutes les informations nécessaires pour l'identification des services soient présentes, en particulier toutes les taches doivent être renseignées par l'information type de tache. Si les informations manquent, le traitement sera arrêté et l'utilisateur sera invité à compléter le diagramme.

3.2. La phase d'analyse des diagrammes BPMN :

L'étape d'analyse permet d'extraire les informations d'une représentation textuelle du diagramme BPMN qui est le format XPDL. Dans notre cas nous nous basons sur deux types de diagrammes processus et collaboration. Ces informations concernent les activités qui composent un processus. Cela nécessite un parcours de la totalité du processus pour atteindre ces derniers.

A partir de la représentation du processus, nous parcourons tous les éléments et les informations utiles dans les instructions simples (le type de tâche) et composées (les branchements, les sous-processus) et d'autres éléments significatifs comme l'échange des messages à travers les flux de messages et aussi l'envoie et la réception des messages, les services identifiés peuvent être des services composites qui orchestrent des services atomiques pour offrir des fonctionnalités plus avancées ou des services atomiques qui peuvent fournir

des fonctionnalités par eux-mêmes. En effet les propriétés des services identifiés sont liées à la réutilisabilité, l'autonomie, la composition. Ces mêmes propriétés existent pour la SOA.

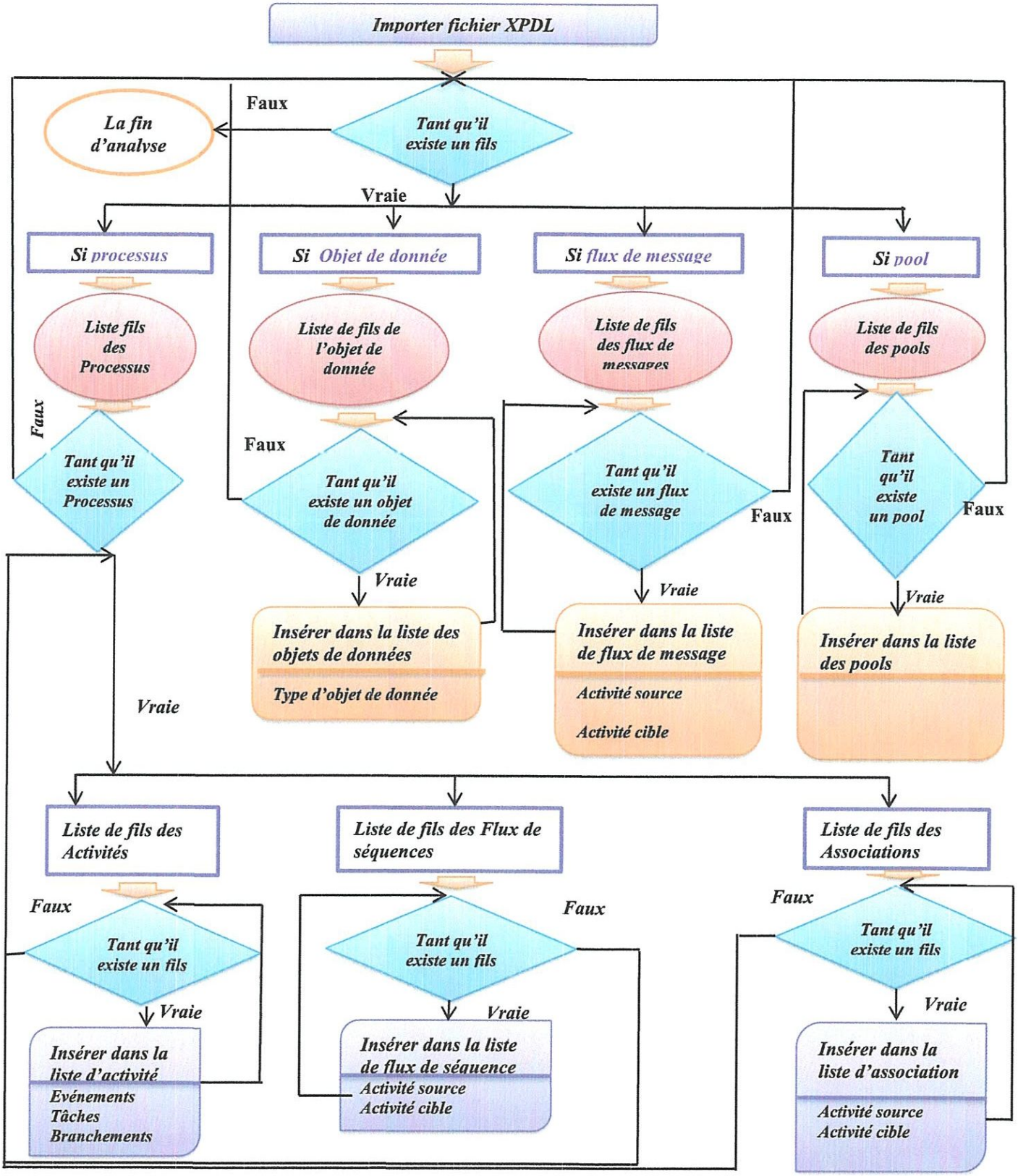


Figure 3.2 : *Processus général de la phase analyse.*

3.3. Règles d'identification des services

Afin de détecter et d'identifier les services candidats, nous avons proposé un certain nombre de règles qui nous permettent de regrouper une ou plusieurs activités de la représentation BPMN sous forme de services. Ces règles sont déduites à partir des caractéristiques que doit assurer un service dans une architecture SOA, à savoir garantir une autonomie, réutilisabilité, composabilité, couplage faible avec son environnement extérieur et une cohésion forte avec les composants internes du service. La tâche est l'élément de base qui peut constituer un service, en se basant sur son type qui permet de préciser son fonctionnement.

3.3.1. Règle 1 : Une Tâche de type service c'est une tâche automatisé c.à.d. sans intervention humain. L'application informatique déclenchée est vue comme un service demandé nous ne connaissons pas le contenue de l'application, mais nous connaissons le résultat produit. Une tâche de type service pourrait être un service Web ou une application automatisée.

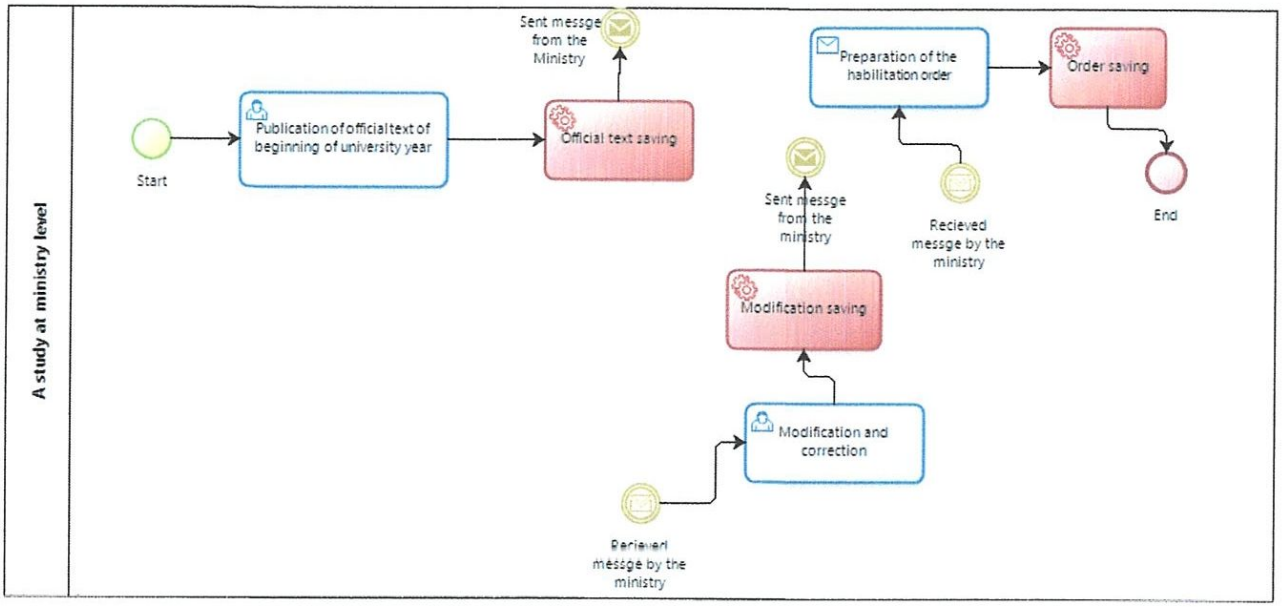


Figure 3.3 : Exemple illustratif de la règle 01.

3.3.2. Règle 2 : Tout un processus peut être considéré comme service si l'ensemble des tâches le constituant sont toutes des tâches automatiques (non manuels et non utilisateur).

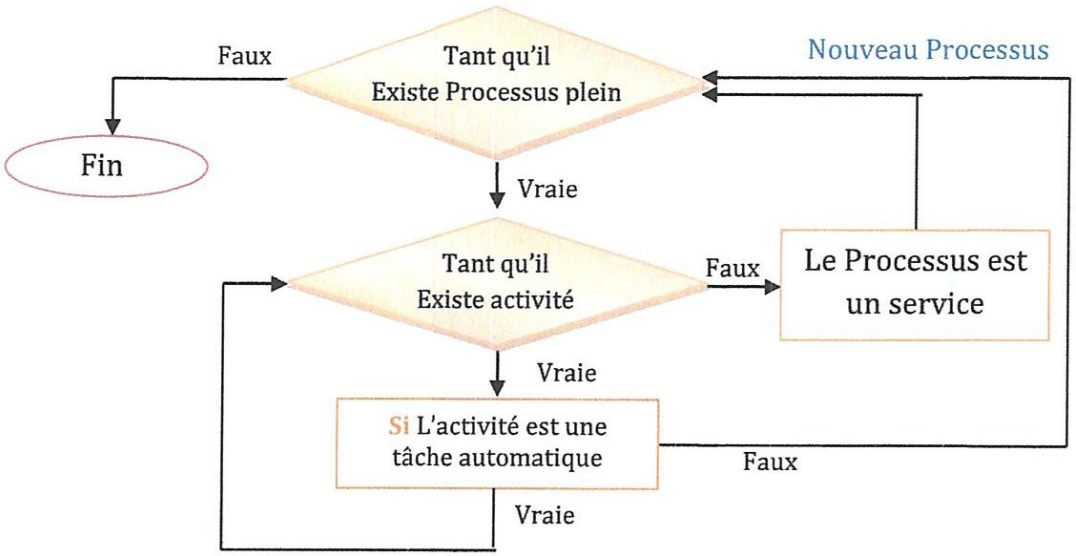


Figure 3.4: *Diagramme illustratif de la règle 2 (Processus).*

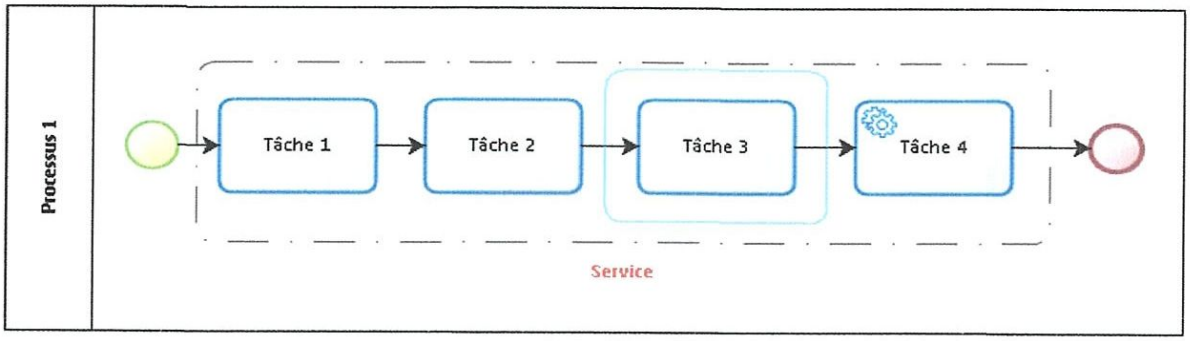


Figure 3.5: *Exemple illustratif de la règle 02.*

3.3.3. Règle 3 : Un processus qui contient au moins une tâche de type non automatique ne peut être considéré globalement comme un service.

3.3.4. Règle 4 :

- Un sous processus détaillé explicitement est traité de la même façon qu'un processus donc il peut être considéré globalement comme service si ces taches sont automatique, comme il peut être une partie d'un service plus grand au niveau du processus père.
- Un sous processus non détaillé ne peut être considéré comme service puisque nous ne pouvons faire un jugement sur le type de ces taches internes.

3.3.5. Règle 5 : Chaque chemin après un branchement peut être considéré comme un service à condition qu'il ne contienne que des tâches de type automatique.

- ✓ Si le branchement est de type parallèle : la passerelle parallèle est utilisée pour synchroniser plusieurs branches simultanées (fusion du comportement).
- ✓ ou inclusif : la passerelle inclusive sélectionne un sous-ensemble de flux alternatifs. Étant donné qu'un chemin d'écoulement peut ne pas être sélectionné dans certaines conditions.
- ✓ Ou exclusif : la passerelle exclusive sélectionne un seul flux entre les flux alternatifs et ne considère aucune priorité parmi eux.

Ce choix est justifié par la propriété de l'autonomie des services. Si chaque branche et automatisée et peut être exécutée en parallèle cela veut dire quelle est autonome des restes.

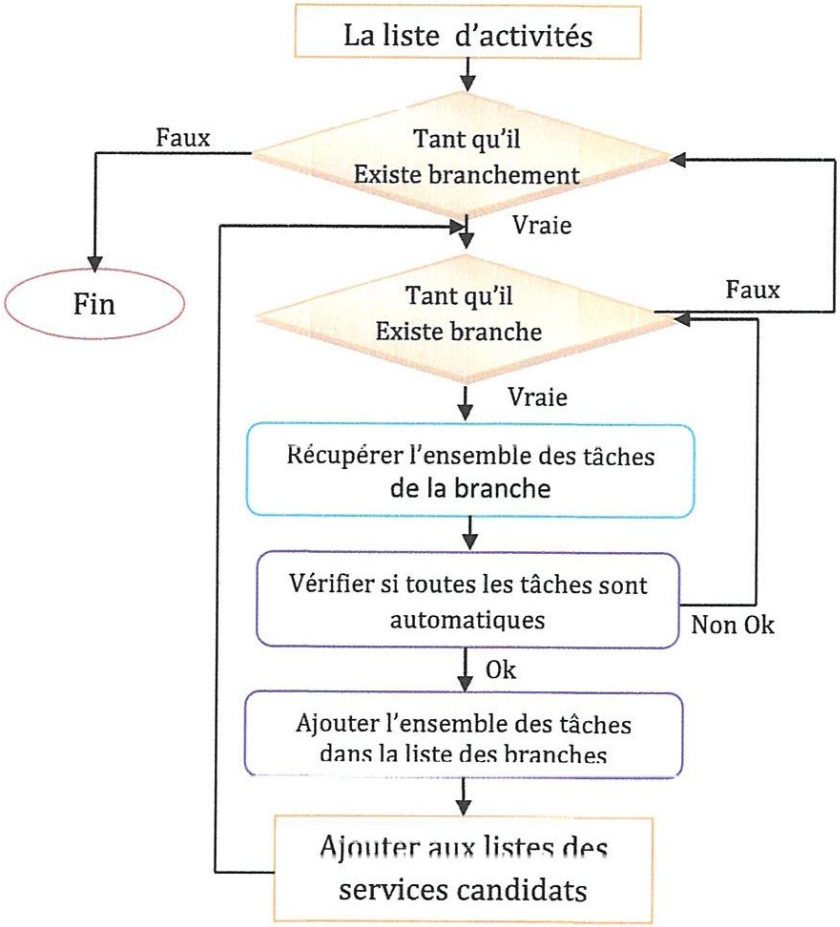


Figure 3.6: *Diagramme illustratif de la règle 5 (règle de branchement).*

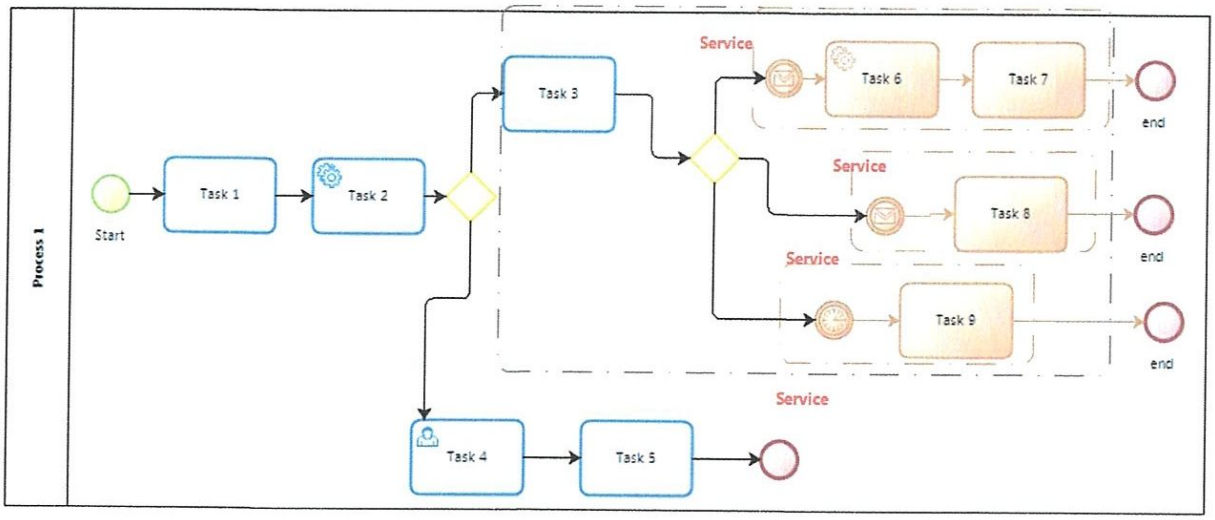


Figure 3.7 : *Exemple illustratif de la règle 05.*

3.3.6. Règle 6 : Un service est une tâche où un ensemble de tâches successives entre la réception et l'envoie de message vers le même processus demandeur de traitement.

Si A et B deux processus qui échangent des messages tout en respectant les conditions suivantes :

- ✓ A demande un traitement à B
- ✓ B envoie un résultat à A
- ✓ La ou les tâches qui sont automatiques de B et qui se trouvent entre la réception et la réponse peuvent être considérées comme service.

En cas de branchement, chaque branche qui retourne une réponse peut être considérée comme service potentiel.

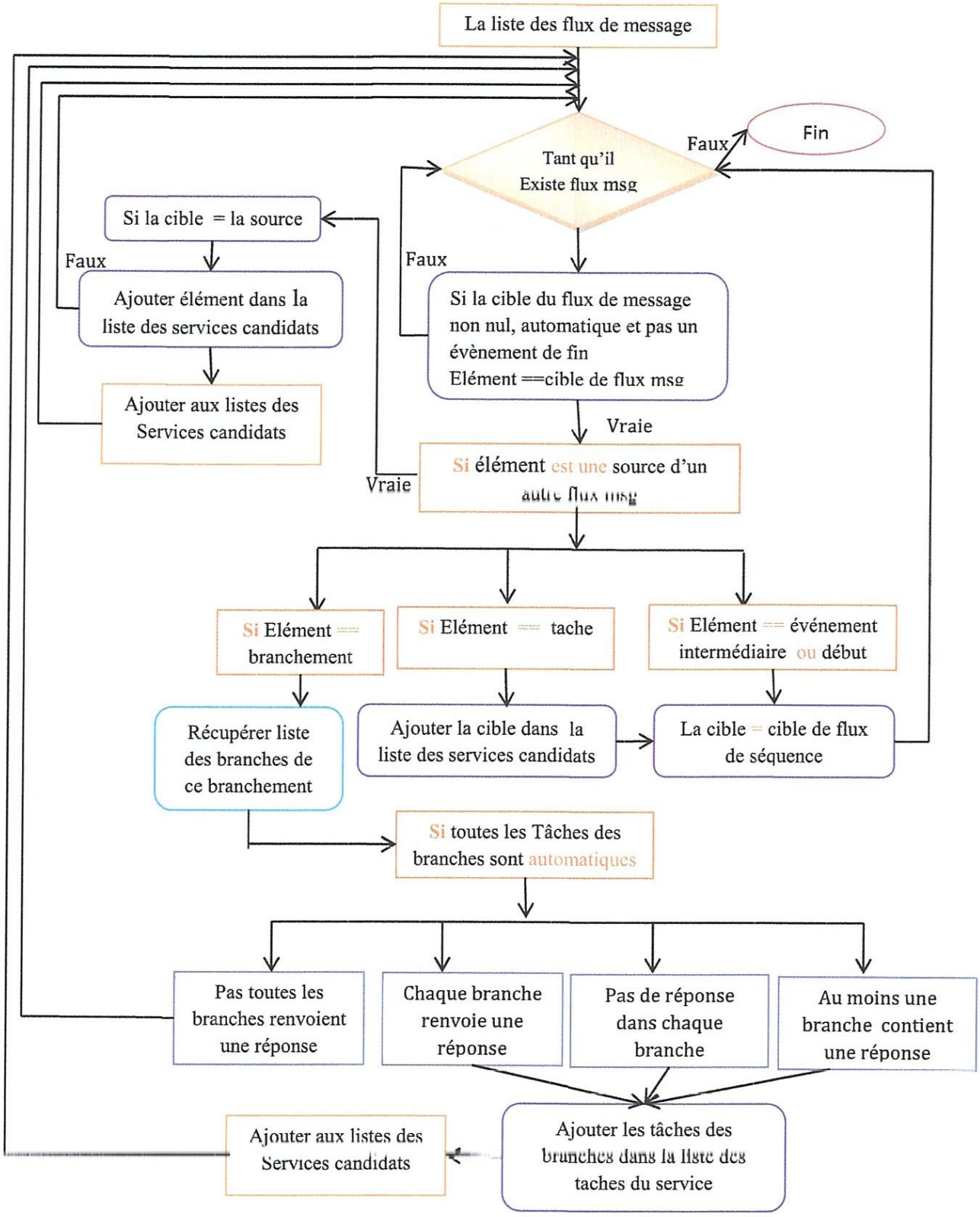


Figure 3.8 : *Diagramme illustratif de la règle 6.*

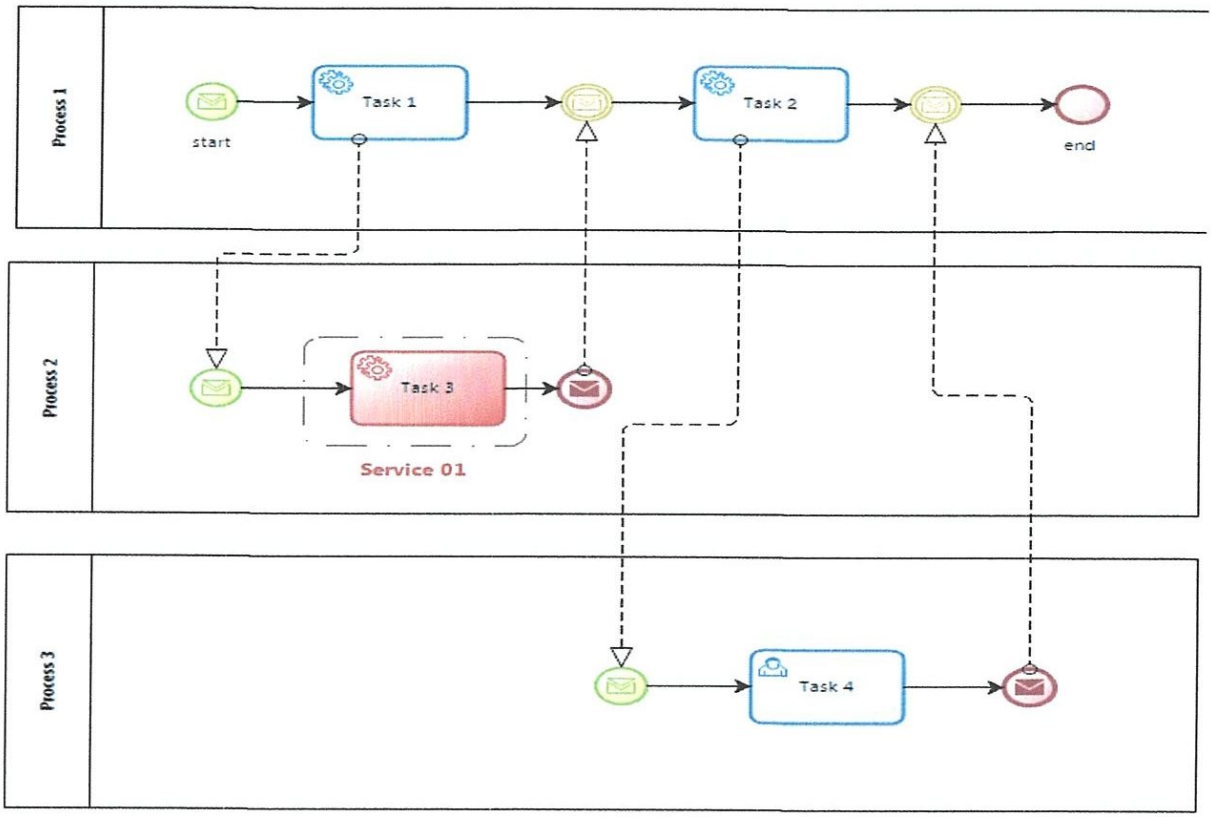


Figure 3.9: *Exemple 1 illustratif de la règle 06.*

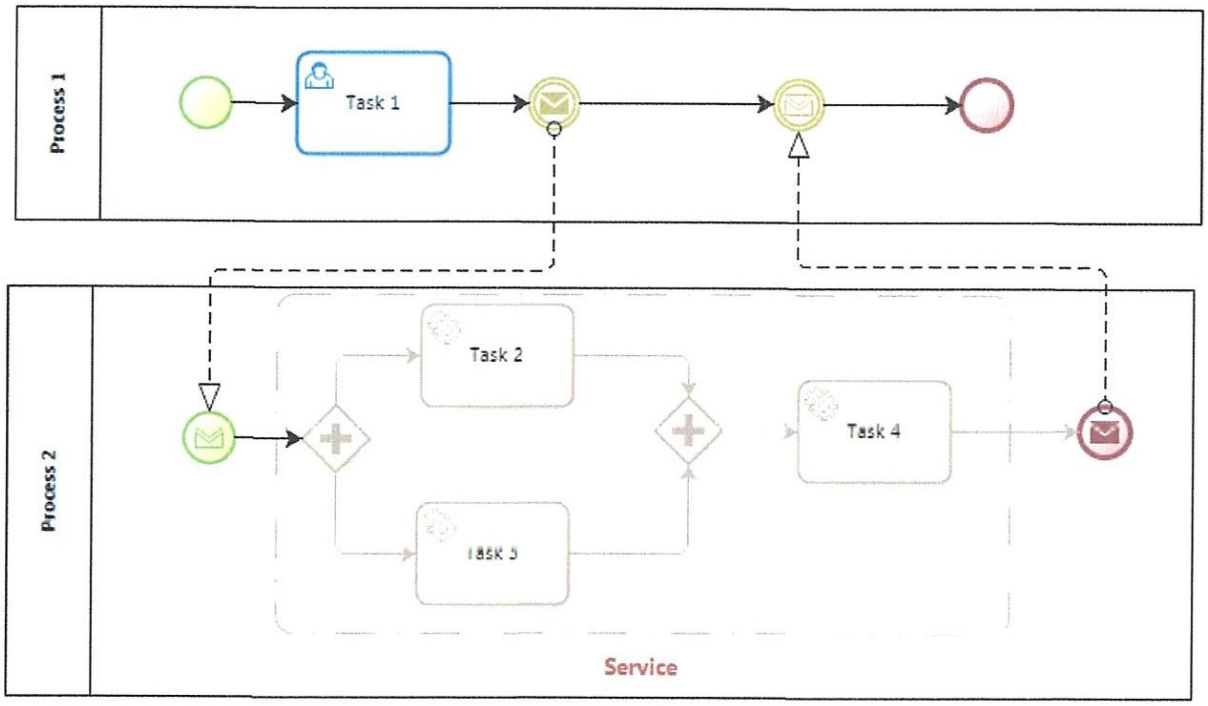


Figure 3.10: *Exemple 2 illustratif de la règle 06.*

3.3.7. Règle 07 : Une séquence de tâches automatiques qui a comme bornes les évènements début, fin, branchement ou une tâche non automatique peut être considéré comme un service.

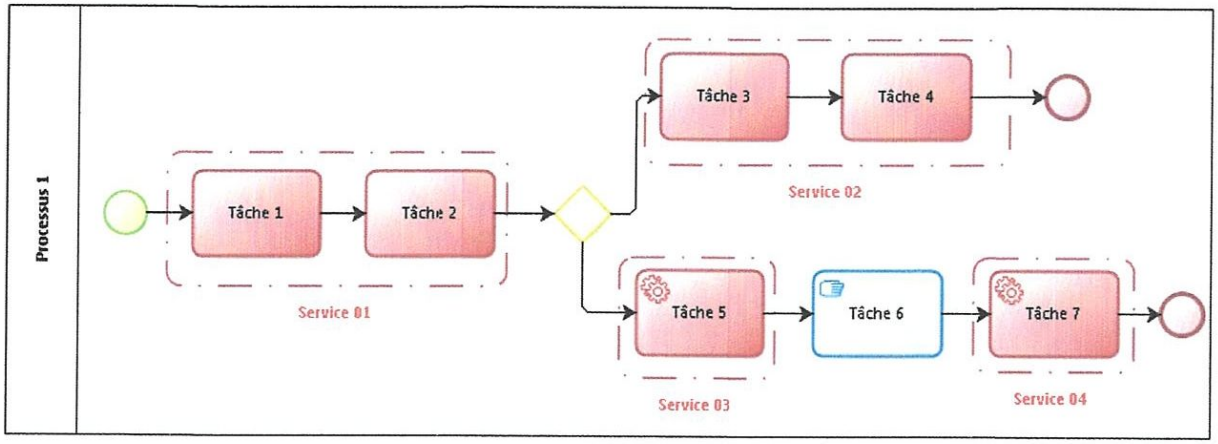


Figure 3.11: *Exemple illustratif de la règle 07.*

3.3.8. Règle 08 : un service candidat peut être identifié à partir d'une structure en boucle (tâche ou ensemble de tâche, branche,...etc). la structure boucle est une structure dans laquelle une ou plusieurs activités peuvent être exécutées à plusieurs reprises.

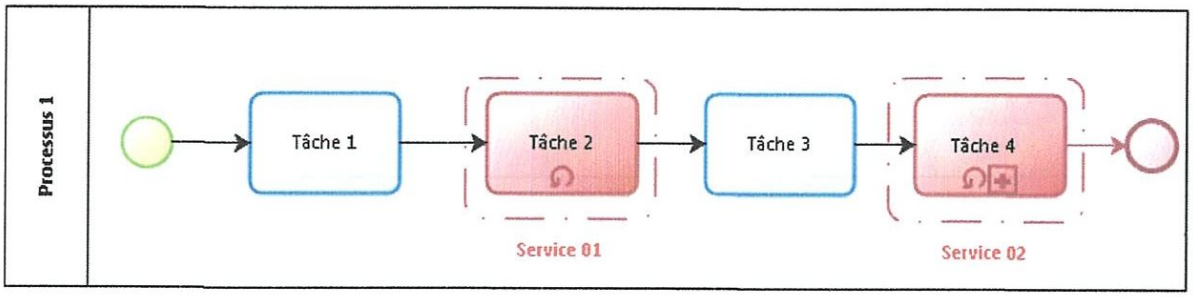


Figure 3.12: *Exemple illustratif de la règle 08.*

3.3.9. Règle 09 : les services candidats peuvent être identifiés à partir d'une Activité multi-instance. Une tâche à instance multiple est une tâche qui peut avoir plusieurs instances d'exécution distinctes s'exécutant simultanément dans le même cas de flux de travail. Chacun de ces cas s'exécute indépendamment.

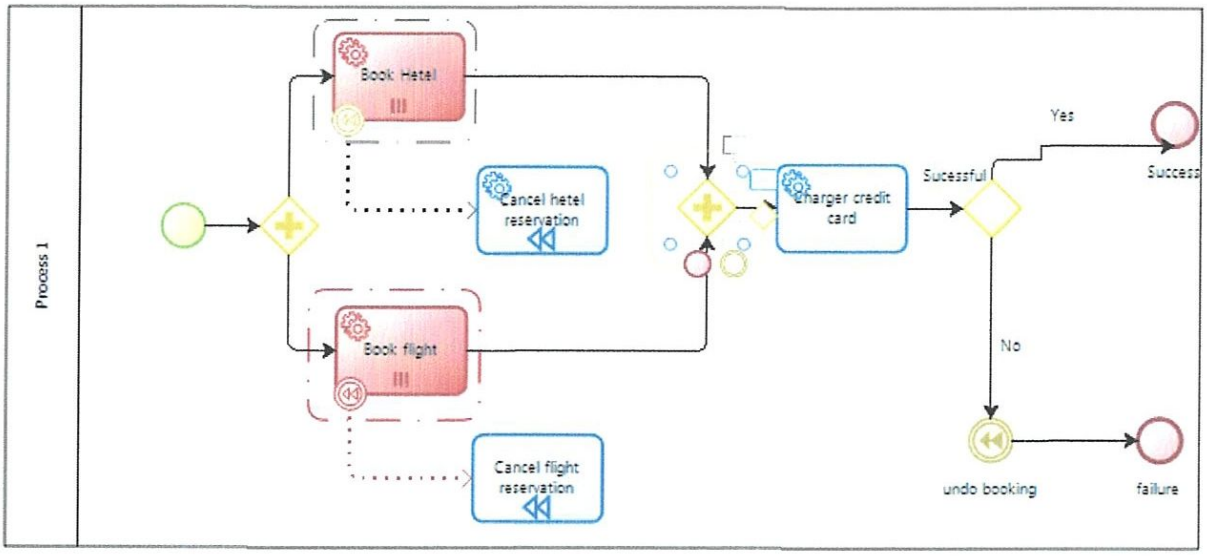


Figure 3.13 : *exemple illustratif de la règle 09.*

4. Gestion des règles :

Un gestionnaire de règle a été développé comme module indépendant et complémentaire à l'analyseur. Il a comme but de rendre notre Framework le plus flexible et paramétrable possible.

Deux types des règles sont gérés les règles prédéfinies et les règles additives.

- Les règles prédéfinies sont les règles les plus évidentes qui permettent une identification de service avec un taux de confiance très élevé détecté de façon empirique par des experts du domaine.

- Les règles additives : sont des règles que nous pouvons ajouter à partir d'un générateur de règles qui a été conçu de façon à ce qu'il puisse prendre en charge tous les éléments qui forment la règle (les évènements, les tâches, les passerelles, les flux de messages, les flux de séquences). Ces règles permettent une flexibilité maximale pour l'identification.

4.1. Gestionnaires des règles :

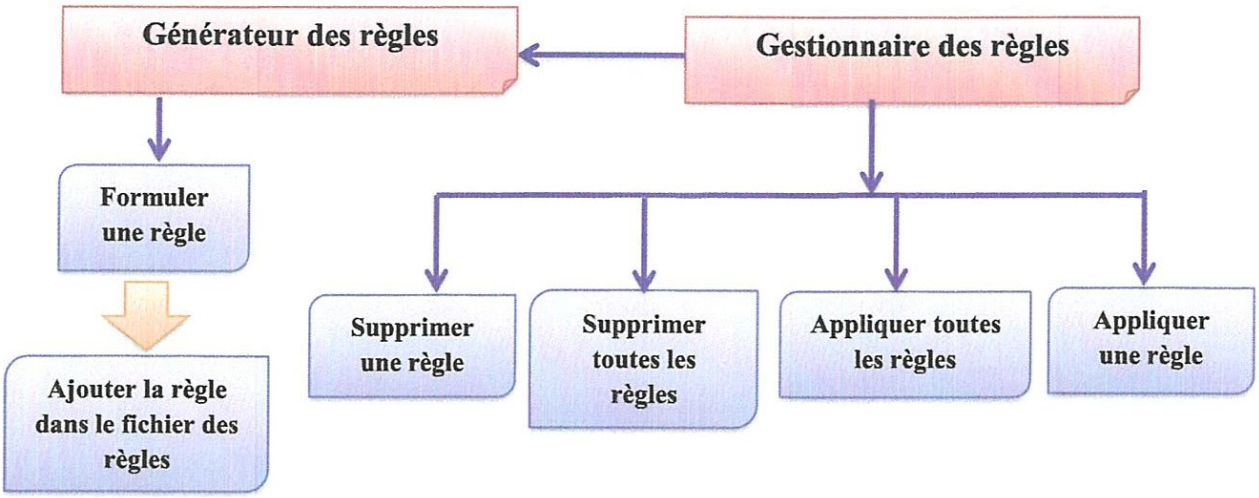


Figure 3.14 : *Gestion des règles.*

4.1.1. Ajouter une règle :

Permet d'ajouter n'importe quelle règle de façon paramétrable.

4.1.2. Application des règles :

- Appliquer plusieurs règles : permet de parcourir les règles prédéfinies implémentées et de les appliquer l'une après l'autre, ainsi que les règles additives ou toutes les règles prédéfinis et additives en même temps.
- Appliquer une règle : permet d'appliquer une règle parmi les règles prédéfinies selon le choix ou une règle parmi les règles additives.

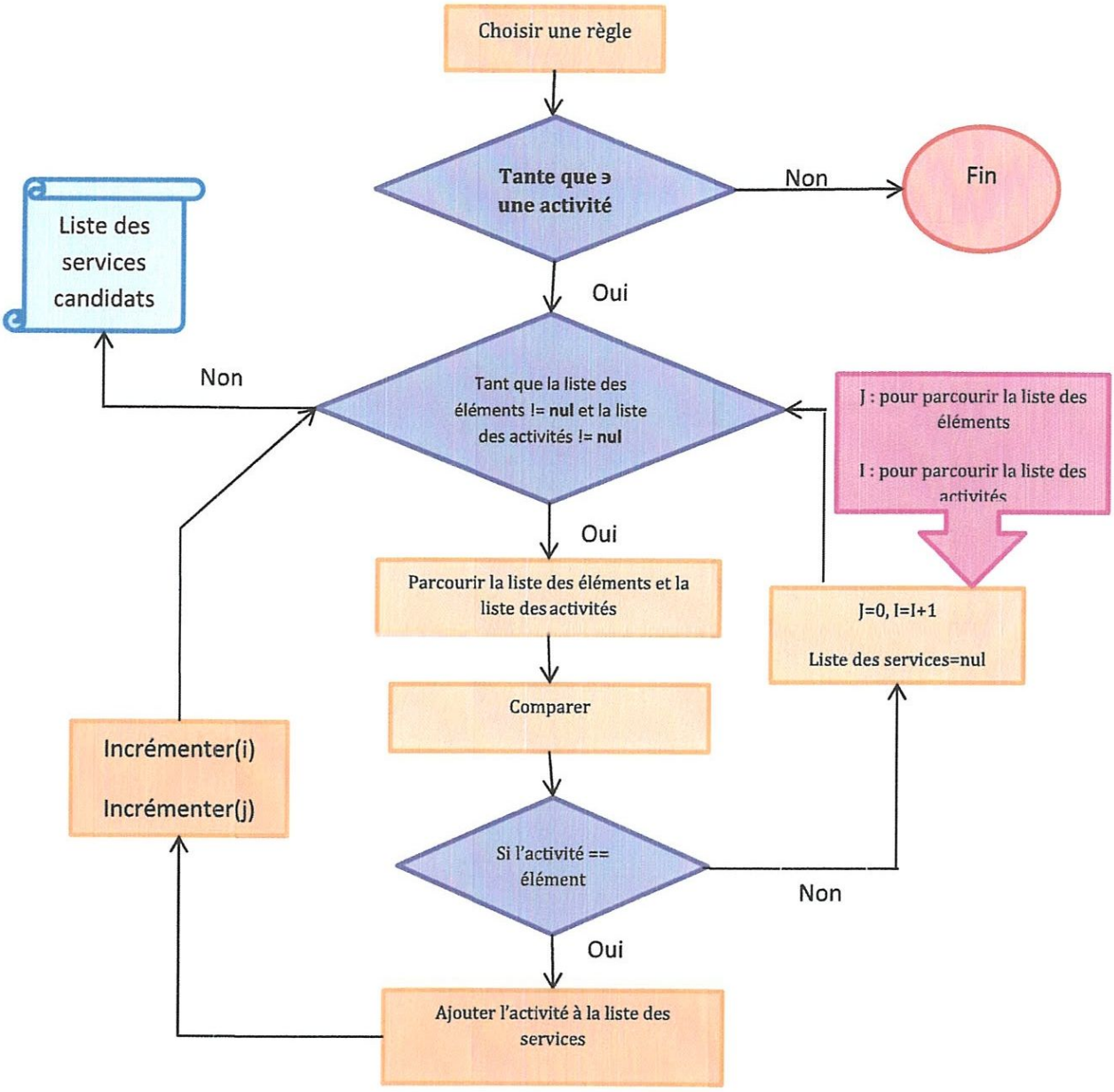


Figure 3.15 : *Diagramme illustratif d'application d'une règle additif.*

5. Validation des services candidats :

Afin de valider les services candidats identifiés, nous avons procédé par deux méthodes.

5.1. Evaluation par les experts : Cette évaluation est complètement empirique, nous donnons la main aux experts du domaine d'affecter à chaque règle, un taux de confiance, qui exprime le niveau de confiance donné à une règle pour identifier les services. ce taux de confiance est affecté automatiquement aux services identifié à partir de la règle, et sa valeur est comprise entre 0 et 1.

5.2. Evaluation par calcul des métriques : Il est nécessaire d'évaluer la qualité des services candidats par un ensemble de mesures de qualité. Malheureusement il est très rare de trouver des métriques qui font le consensus pour mesurer la qualité des services identifiés à partir des BPM. Dans des cas, nous trouvons des métriques sans qu'ils soient épaulés par des formules mesurables. Une des métriques que nous avons pu calculer est le couplage qui mesure les dépendances fonctionnelles et informatives entre les tâches / processus dans un modèle de processus métier.

5.2.1. calcul du Couplage : parmi les formules de calcul de couplage qui ont fait un consensus acceptable est celle de Vanderfeesten [21].

$$CP = \frac{\sum_{t_1,t_2 \in T} connected(t_1, t_2)}{|T| * (|T| - 1)}$$

where $connected(t_1, t_2) =$

$$\begin{cases} 1 & , \text{ if } (t_1 \rightarrow t_2) \wedge (t_1 \neq t_2) \\ 1 & , \text{ if } (t_1 \rightarrow \text{AND} \rightarrow t_2) \wedge (t_1 \neq t_2) \\ \frac{1}{(2^m-1)\cdot(2^n-1)} + \frac{(2^m-1)\cdot(2^n-1)-1}{(2^m-1)\cdot(2^n-1)} \cdot \frac{1}{m \cdot n} & , \text{ if } (t_1 \rightarrow \text{OR} \rightarrow t_2) \wedge (t_1 \neq t_2) \\ \frac{1}{m \cdot n} & , \text{ if } (t_1 \rightarrow \text{XOR} \rightarrow t_2) \wedge (t_1 \neq t_2) \\ 0 & , \text{ if } (t_1 = t_2) \end{cases}$$

Dans lequel

- t1and t2 sont des activités.
- m et le nombre d'arcs entrants dans le connecteur.
- n le nombre d'arcs sortants du connecteur.

Dans notre projet, nous avons calculé deux types de couplage :

- ***Couplage Externe de service :*** qui doit être faible avec l'extérieur.
  Le but du principe de couplage faible entre les services est de garder une faible liaison entre les services, pour atteindre un haut degré de flexibilité dans l'architecture SOA.
- ***Couplage Interne de service :*** qui doit être fort entre les activités du service.
  Un couplage interne fort d'un service implique que les activités de ce dernier sont fortement reliées entre elles.

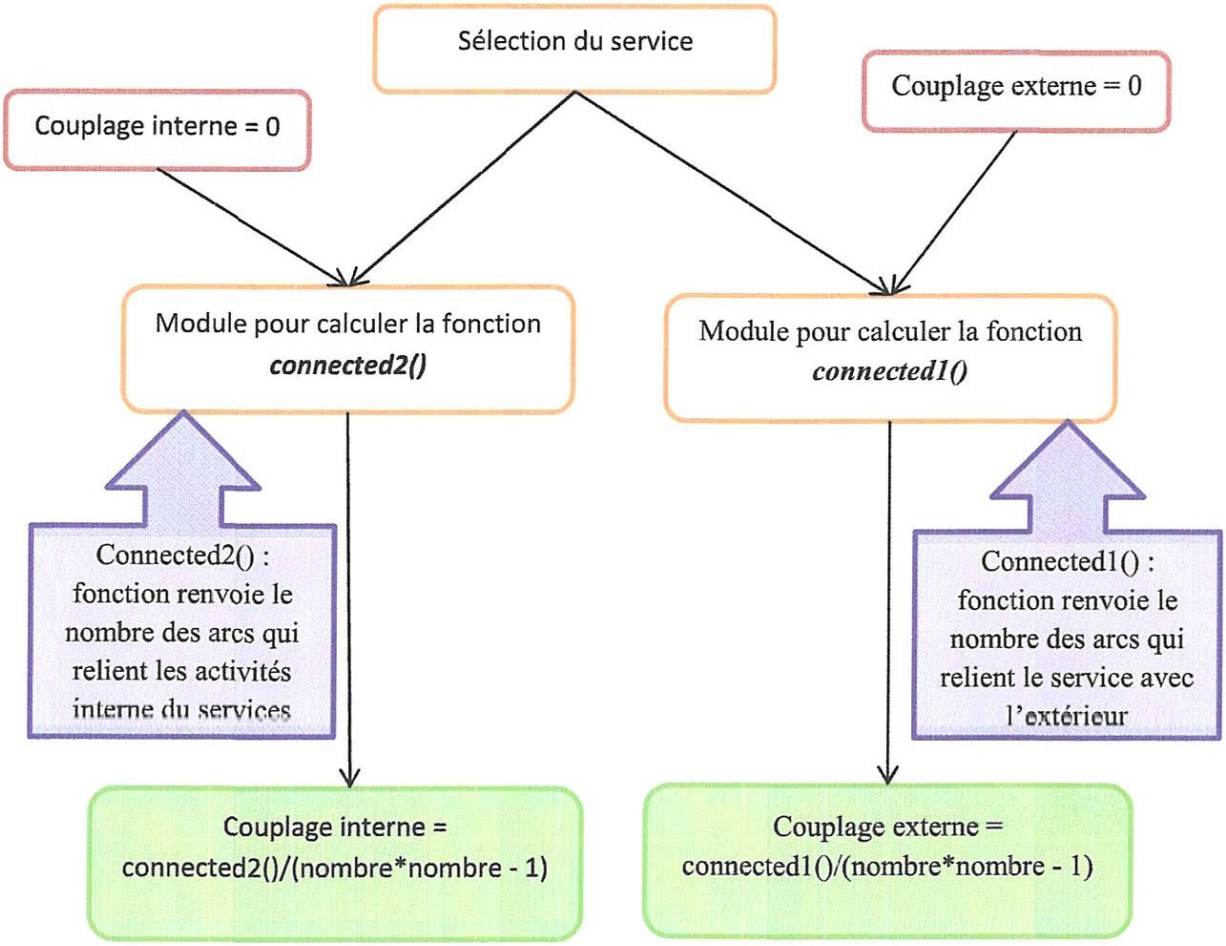


Figure 3.16: *calcul du couplage.*

6. Conclusion :

La phase de conception consiste à présenter la démarche à suivre qui dépend du problème à traiter. Nous avons présenté dans ce chapitre la problématique à traiter, à savoir l'automatisation de l'identification des services à partir des diagrammes BPMN, Puis nous avons fixé clairement les objectifs à atteindre et enfin nous avons détaillé les solutions apportées à travers un Framework dont nous avons explicité chaque composant.

Le chapitre prochain sert à présenter les détails d'implémentation du projet ainsi que les outils utilisés.

1. Introduction :

Ce chapitre a pour objectif de présenter les détails d'implémentation de notre projet. C'est la phase de réalisation de notre Framework qui consiste à automatiser l'identification des services à partir d'une représentation BPMN2 d'un processus métiers.

Ce chapitre est composé de deux parties : la première partie présente l'environnement de développement alors que la seconde partie concerne les principales interfaces graphiques.

2. Les outils de développement :

2.1. Java :

C'est un langage de programmation orienté objet, développé par Sun Microsystems. C'est un langage portable qui ne dépend pas d'une plateforme donnée. Il peut être utilisé sous n'importe quel système d'exploitation à savoir Windows, macintosh, linux, etc. Java est donc un langage multiplateforme qui permet aux développeurs d'écrire un code qu'ils peuvent exécuter sur divers plateformes.

2.2. La plateforme Eclipse :

Eclipse est un environnement de développement intégré (Integrated Development Environment) dont le but est de fournir une plate-forme modulaire pour permettre de réaliser des développements informatiques.

2.3. Bizagi Modeler :

C'est l'outil de modélisation avec le meilleur environnement permettant de représenter graphiquement les activités et les processus, l'outil a également un fonctionnement intuitif et une façon très pratique de gérer les processus, il utilise une interface de « style RIBBON ». Il permet de créer des diagrammes BPMN [14].

2.4. XPDL (XML Prosessing Description Language) :

L'entrée de notre application est un fichier XPDL, XPDL est une Norme du WfMC (Workflow Management Coalition) destinée à décrire les processus métier à l'aide du langage de balisage XML (Extensible Markup Language). Avec XPDL, un produit peut écrire une définition de processus avec une pleine fidélité, et un autre produit peut lire et reproduire le même schéma qui a été envoyé.

Chapitre 04 :

Implémentation

2.5. JDOM :

Nous avons utilisé JDOM comme un package dans notre application pour analyser le fichier XML. JDOM est disponible sous une licence open source.

JDOM est un "modèle d'objet de document" basé sur Java pour les fichiers XML. JDOM n'est pas un analyseur XML, C'est un modèle objet de document qui utilise des analyseurs XML pour créer des documents, de naviguer dans leur structure, d'ajouter, de modifier, ou de supprimer leur contenu.

3. L'architecture globale de notre application :

Notre framework est composé essentiellement de trois modules principals :

- **Module d'analyse :** responsable de la verification de bonne structure du fichier à analyser puis de la récuperetaion de toutes des informations necessiares pour l'identification.
- **Module de gestion des régles** : qui a comme objectif de gerer toutes les régles permetant l'identification, en plus il integre un générateur de règles qui permet d'introduire les nouvelles règles.
- **Module d'identification :** qui travaille en étroite collaboration avec les modules cités ci-dessus afin d'identifier les services candidats en applicants les différentes regles sur les différentes liste issues de l'étape d'analyse

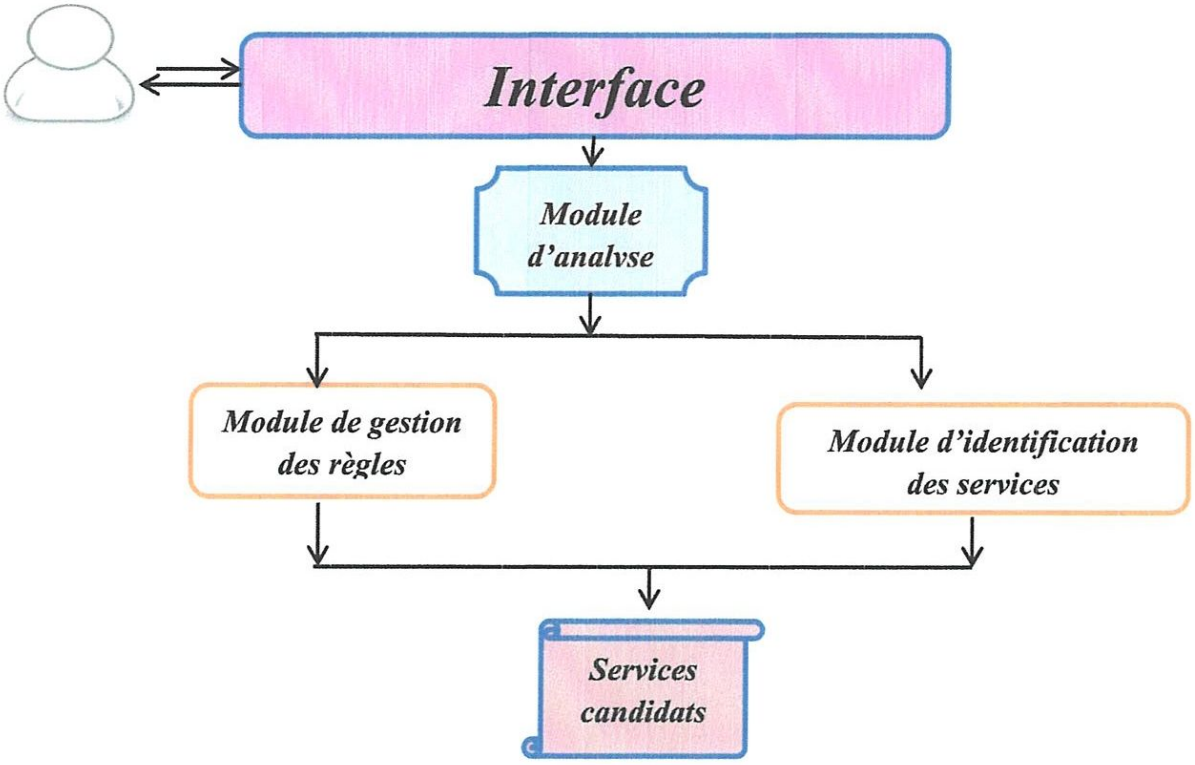


Figure 4.1 : *l'architecture globale de notre application.*

4. Présentation de notre application :

Figure 4.2: *L'interface de notre application.*

4.1. Les options d'accueil :

- Charger un fichier XPDL : 
- Afficher le diagramme de BPMN 2.0 graphiquement : 
- Appliquer toutes les règles et identifier les services : 
- Appliquer une règle et identifier les services : 
- Ajouter une règle : 
- Gestionnaire des règles : 
- Exporter la liste des services candidats 
- Le guide d'utilisation : 
- Reinitialiser les fenêtres précédentes : 
- Quitter l'application : 

La partie suivante détaille chaque fonctionnalité.

4.1.1. Charger un fichier XPDL :

Après avoir modélisé le processus à l'aide de l'outil Bizagi et de l'avoir exporté dans un fichier XPDL nous pouvons l'importer dans notre application.

4.1.1.1. La phase prétraitement :

Si le type des taches n'est pas défini, un message d'information signale les taches qui ne sont pas définies.

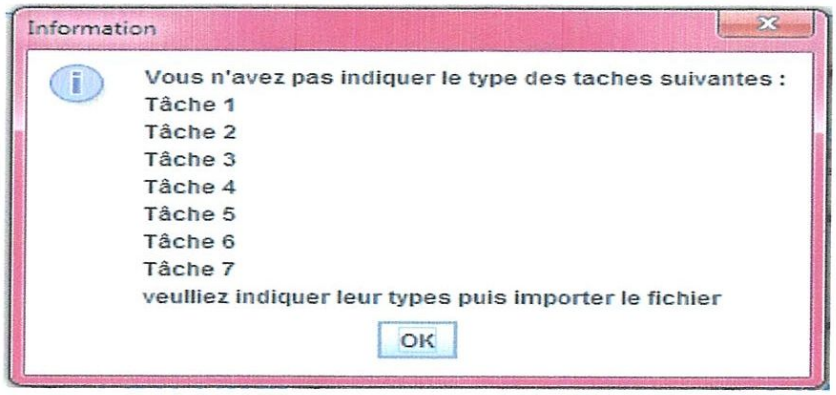


Figure 4.3: *La phase prétraitement.*

4.1.2. Afficher le diagramme de BPMN 2.0 graphiquement :

Après avoir importé le fichier XPDL on peut l'afficher graphiquement :

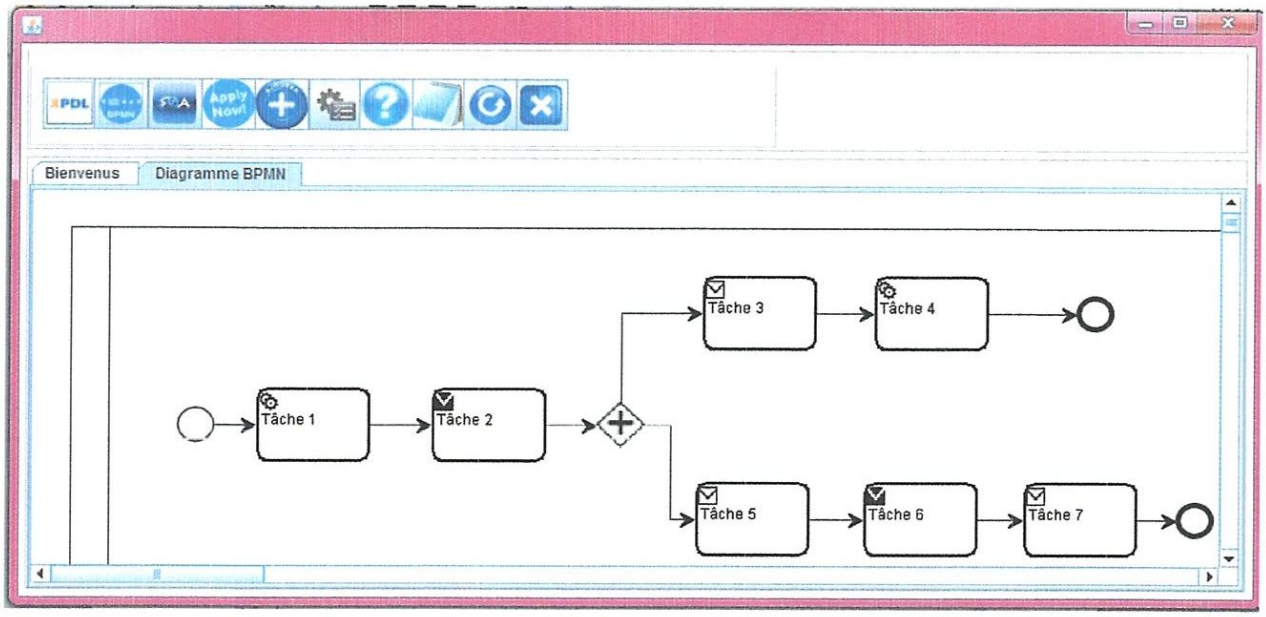


Figure 4.4 : *Afficher graphiquement le diagramme de BPMN 2.0.*

4.1.3. Appliquer toutes les règles et identifier les services :

Cette partie permet d'appliquer sur le fichier importé les règles prédéfinies suivantes :

1. ***Règle 01 :*** Regrouper les taches de type service (Une Tâche de type service c'est une täche automatisé c.à.d. sans intervention humain.).
2. ***Règle 02 :*** Un processus (pool) est un service constitué de l'ensemble de ses tâches si toutes ces tâches ne sont pas de type manuel.

3. ***Règle 03 :*** Chaque chemin après un branchement peut être considéré comme un service à condition de :
   - ✓ Si le branchement est de type parallèle ou inclusif ou exclusif.
   - ✓ Si le chemin ne contient aucune tâche de type manuel.
4. ***Règle 04 :*** Un sous-processus peut être considérer comme un service.
5. ***Règle 05 :*** Un service est une tâche où un ensemble de tâches successives entre la réception et l'envoie de message vers le même processus demandeur de traitement.

Et d'afficher les services candidats :

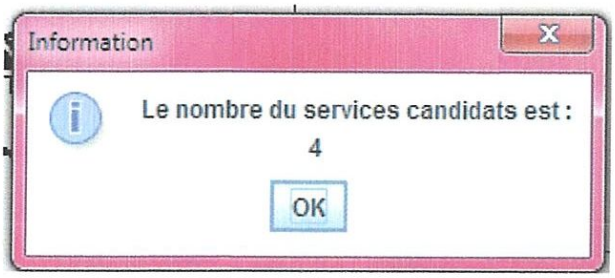


Figure 4.5: *Le nombre des services candidats.*

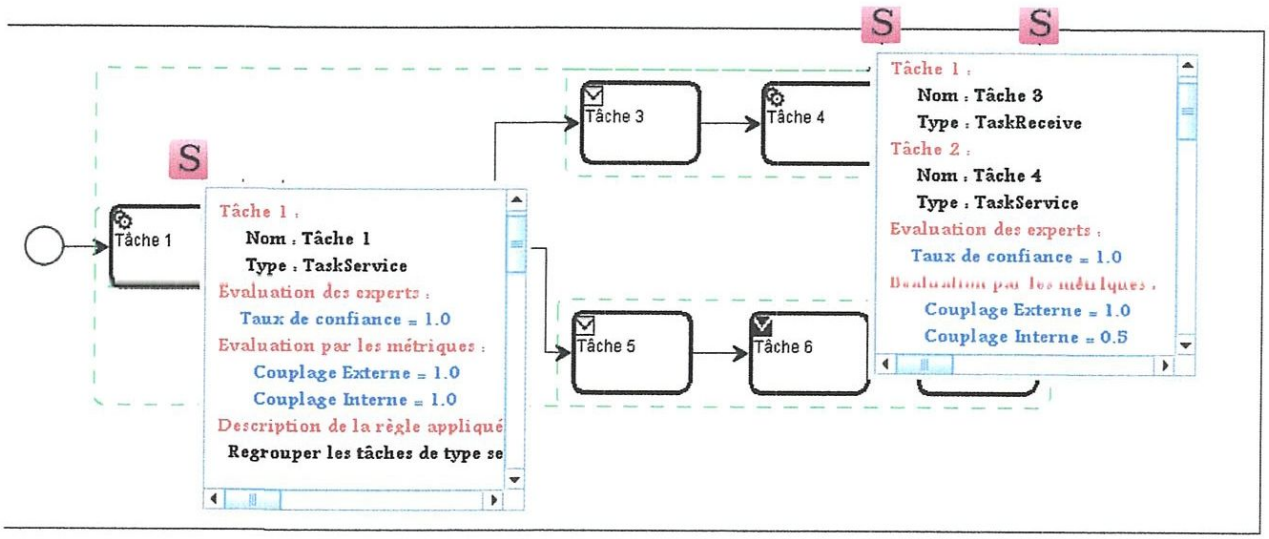


Figure 4.6: *résultat d'identification des services.*

Si l'utilisateur n'a pas sélectionné un fichier XPDL, une boite de dialogue signale l'absence du fichier :

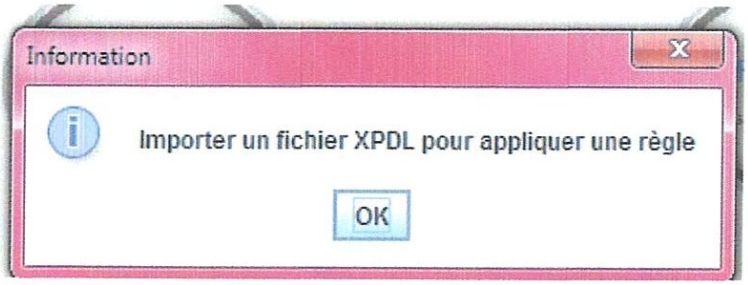


Figure 4.7: *manque d'un fichier XPDL.*

4.1.4. Appliquer une règle et identifier les services :

Cette partie permet d'appliquer une règle parmi les règles que nous avons citées ci-dessus :

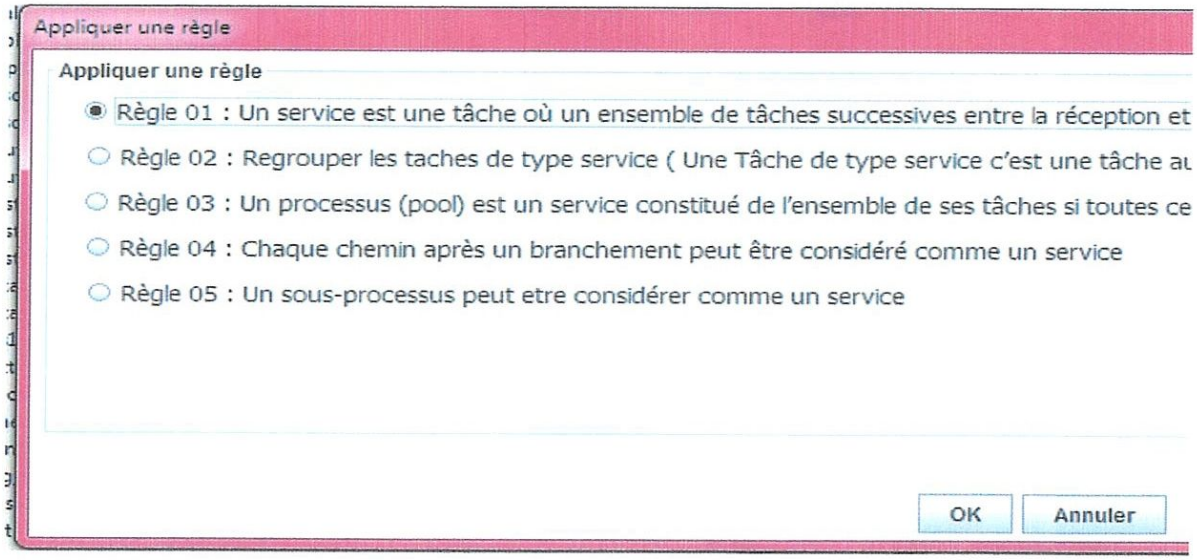


Figure 4 .8: *Appliquer une règle.*

Après avoir choisi une règle, cette règle sera affichée pour la confirmer :

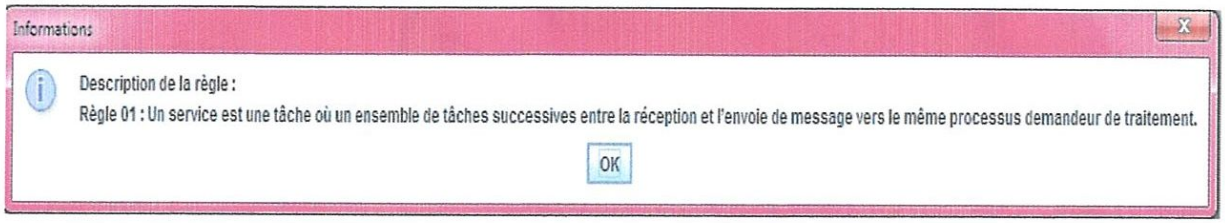


Figure 4.9: *description de la règle.*

Enfin, on va afficher la liste des services candidats.

4.1.5. Générateur de règle :

Cette partie permet de formuler et d'ajouter de nouvelles règles, à travers une interface qui comporte les éléments de modélisation BPMN.

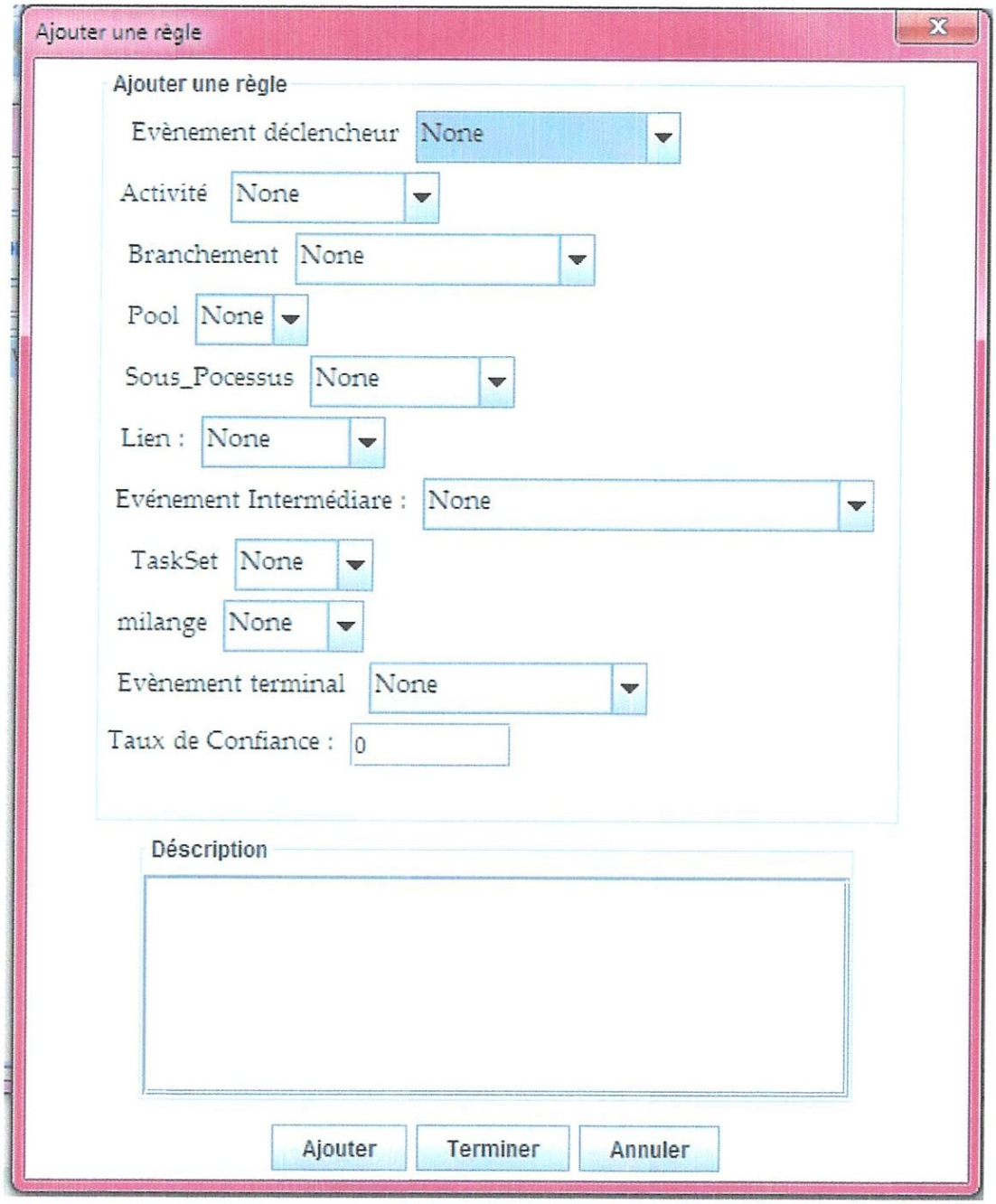
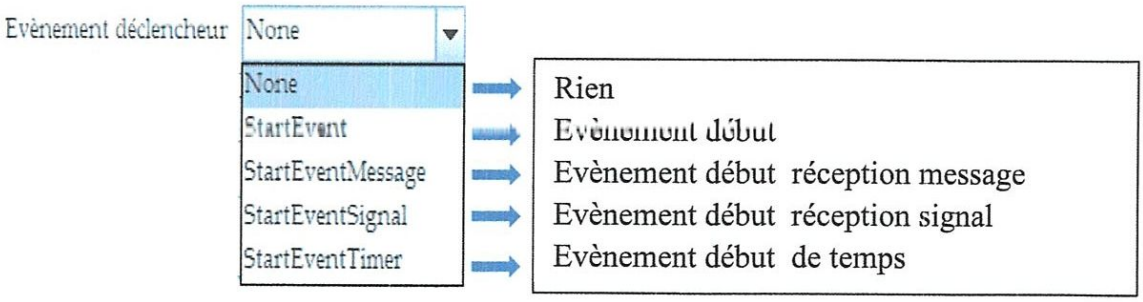
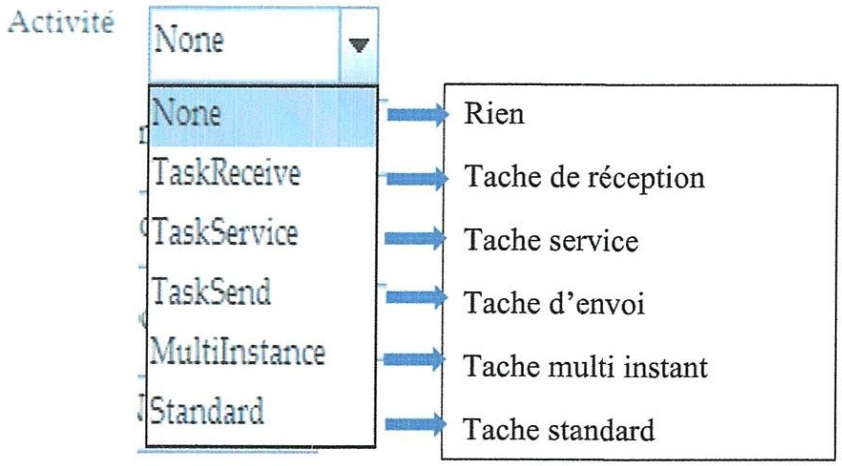
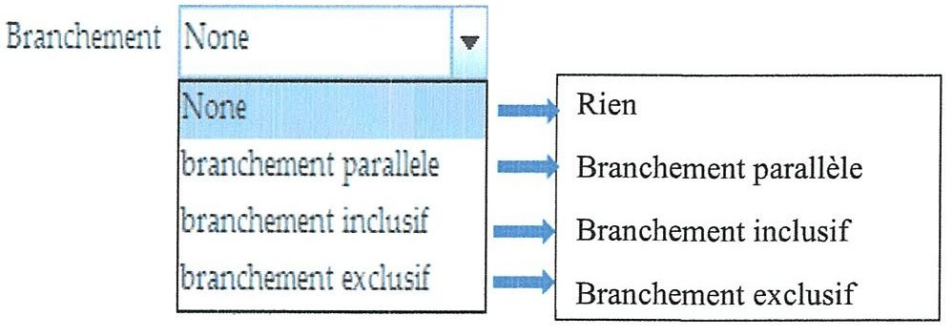
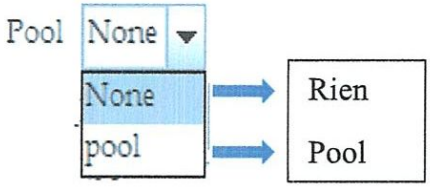


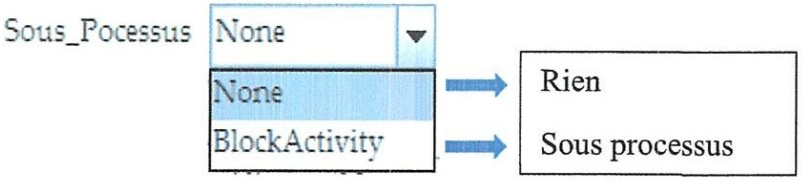
Figure 4.10: *ajouter une règle.*




Activité
None
None
TaskReceive
TaskService
TaskSend
MultiInstance
Standard
Rien
Tache de réception
Tache service
Tache d'envoi
Tache multi instant
Tache standard


Branchement
None
None
branchement parallele
branchement inclusif
branchement exclusif
Rien
Branchement parallèle
Branchement inclusif
Branchement exclusif


Pool
None
None
pool
Rien
Pool


Sous_Pocessus
None
None
BlockActivity
Rien
Sous processus

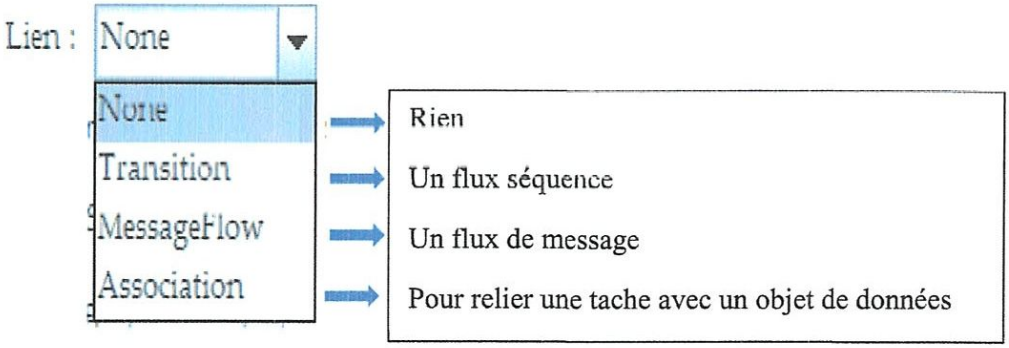

Lien :
None
None
Transition
MessageFlow
Association
Rien
Un flux séquence
Un flux de message
Pour relier une tache avec un objet de données

Figure 4.11 : *Signification des éléments de l'interface du générateur de règle.*

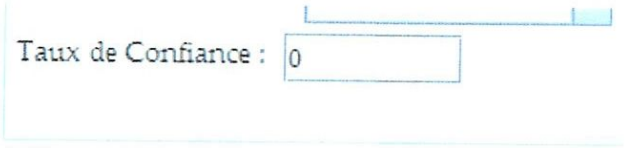


Figure 4.12 : *évaluation des experts.*

- **Bouton Ajouter :** permet de valider un élément.
- **Bouton Terminer :** permet de regrouper les éléments validés et de les enregistrer sous forme de règle dans un fichier, puis quitter la page.

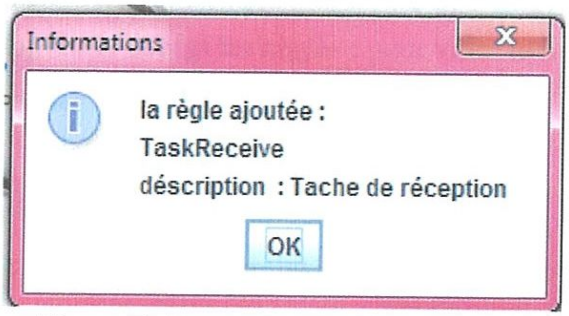


Figure 4.13: *exemple de règle ajoutée.*

- **Bouton Annuler :** permet de quitter la page sans enregistrer la règle.

4.1.6. Gestionnaire des règles :

Cette partie permet d'appliquer plusieurs opérations sur les règles ajoutées :

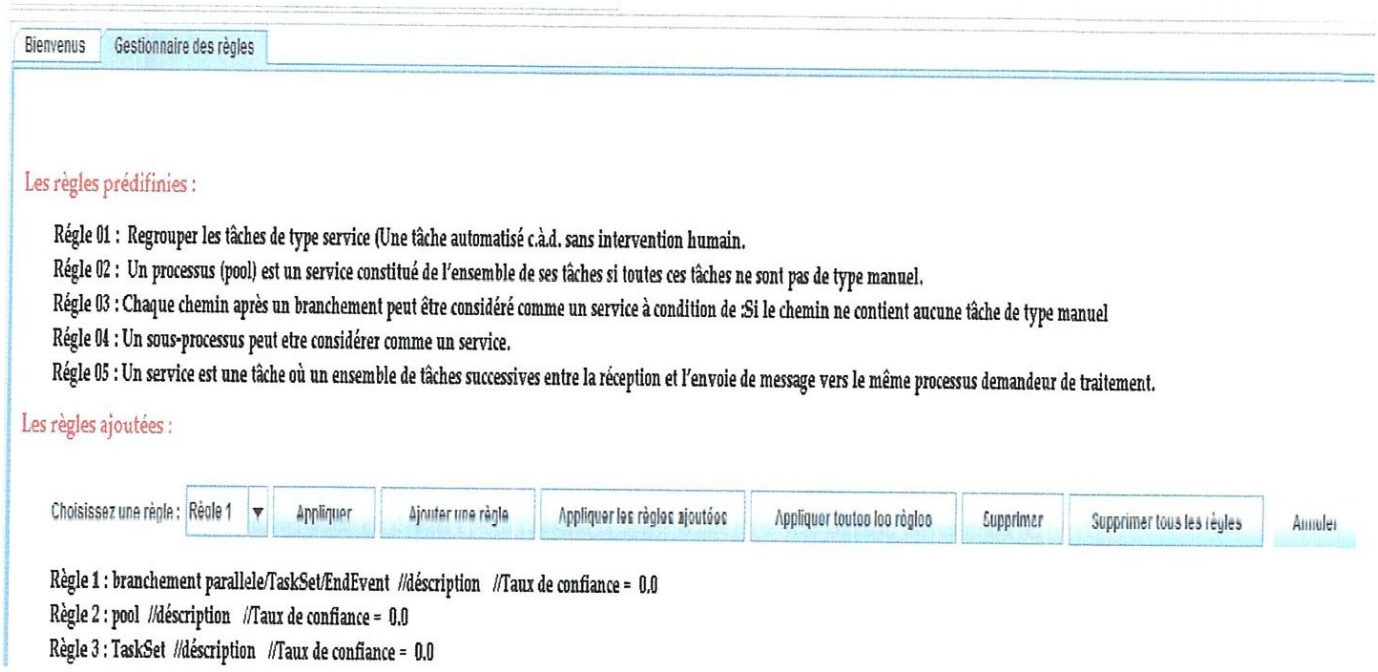


Figure 4.14: *gestionnaire des règles.*

4.1.8. Exemple de code source d'identification :

execut.java | analyseur.java | regle.java | Regle1.java

```
527             //System.out.println("i = "+i);
528             if((i==activitiesList.size())) j=d.size();
529             else{
530                 System.out.println("     while j "+j +" i = "+i+" k = "+k+"  "+activitiesList.get(i).name);
531                  if((activitiesList.get(i).type).equals((d.get(j)))
532                         &&((d.get(0)).equals("branchement parallele")||(d.get(0)).equals("branchement inclusif")
533                                 ||
534                                 (d.get(0)).equals("branchement exclusif")
535                                 ||(d.get(0)).equals("StartEventMessage")
536                                 ||(d.get(0)).equals("StartEventSignal")
537                                 ||(d.get(0)).equals("IntermediateEventMessage")
538                                 ||(d.get(0)).equals("IntermediateEventSignal"))
539                         &&(d.get(1)).equals("TaskSet")){
540                      System.out.println("     while j "+j +" i = "+i+" k = "+k);
541
542                      List<String> s=nbre_existe(d,activitiesList);
543                      for(int l=0;l<s.size();l++){
544                     iddd=s.get(l);
545
546                      List<String> b=branchem1(iddd,transitions);
547
548                 branchem(iddd,Service,activitiesList,services,transitions,b,d,messages);
549
550                     Service = new ArrayList<Activity>();
551                     i = activitiesList.size();
553                 }
554
555             else if((activitiesList.get(i).type).equals((d.get(j)))
556                     &&((d.get(0)).equals("branchement parallele")||(d.get(0)).equals("branchement inclusif")
557                     ||(d.get(0)).equals("branchement exclusif")
558                     ||(d.get(0)).equals("StartEventMessage")
559                     ||(d.get(0)).equals("StartEventSignal")
560                     ||(d.get(0)).equals("IntermediateEventMessage")
561                     ||(d.get(0)).equals("IntermediateEventSignal")
562                     )
563                     &&(d.get(1)).equals("milange")){
564                  List<String> s=nbre_existe(d,activitiesList);
565                  System.out.println("size nbre_existe =  "+s.size());
566                  for(int l=0;l<s.size();l++){
567                 iddd=s.get(l);
568                  System.out.println("size nbre_existe =  "+s.size()+" l = "+l);
569                 System.out.println("branchemmmm");
570                 List<String> b=branchem1(iddd,transitions);
571                 System.out.println("branchem1 = "+b.size());
572                 for(int m=0;m<b.size();m++){
573                 System.out.println("                    bbb size  = "+getActivityById(b.get(m),activitiesList).name);}
574                 //String idd=tache3(activitiesList.get(i).id,transitions);
575                     System.out.println("idddddd111     "+iddd);
576                 System.out.println("idddddd     "+iddd);
577                 //if(idd!=iddd){
578             branchem(iddd,Service,activitiesList,services,transitions,b,d,messages);
579
580             Service = new ArrayList<Activity>();
581                 i = activitiesList.size();}}
582             else if((activitiesList.get(i).type).equals((d.get(j)))
583                     ||(activitiesList.get(i).typeloop).equals((d.get(j)))){
584                 iddd=activitiesList.get(i).id;
585                 System.out.println("      "+activitiesList.get(i).name+"    "+d.get(j));
586                 j++;
587                  List<Integer> l1 = new ArrayList();
588                  Servic.add(activitiesList.get(i));
589                     if (existe(Servic, services)) {
590
591
592                         } else {
593
594                     l1.add(activitiesList.get(i).XCoordinate-150);
595                     l1.add(activitiesList.get(i).YCoordinate-5);
596                     l1.add(activitiesList.get(i).XCoordinate+Servic.get(0).Width+10);
597                     l1.add(activitiesList.get(i).YCoordinate+Servic.get(0).Height-10);
598
599                     if((Servic.size()>0)&(Servic.size()==(d.size()))){
600
601                         lll = lll + 1;
602                         String llll2 = new String();
603                         llll2 = llll2.valueOf(lll);
604                         double CouplageExterne=0;
605                         double CouplageInter=0;
606                         Servicetemp sztem = new Servicetemp("",l1, llll2, Servic, 6, Servic.get(0).id sp,Servic.size());
```

Figure 4.16 : *Exemple de code source d'identification à partir de règles additives.*

4.1.8. Exporter la Liste des services candidats :

Afin de mémoriser les résultats de l'identification, nous avons proposé une option qui permet d'exporter ces derniers dans un fichier texte qui contient toutes les informations associées aux services candidats identifiés.

5. Evaluation des services :

a) Le schéma suivant donne une idée sur le résultat attendu par une opération d'identification où nous pouvons remarquer les tâches composant le service ainsi que l'évaluation de ce dernier suivant les deux méthodes : (évaluation des experts et évaluation par les métriques).

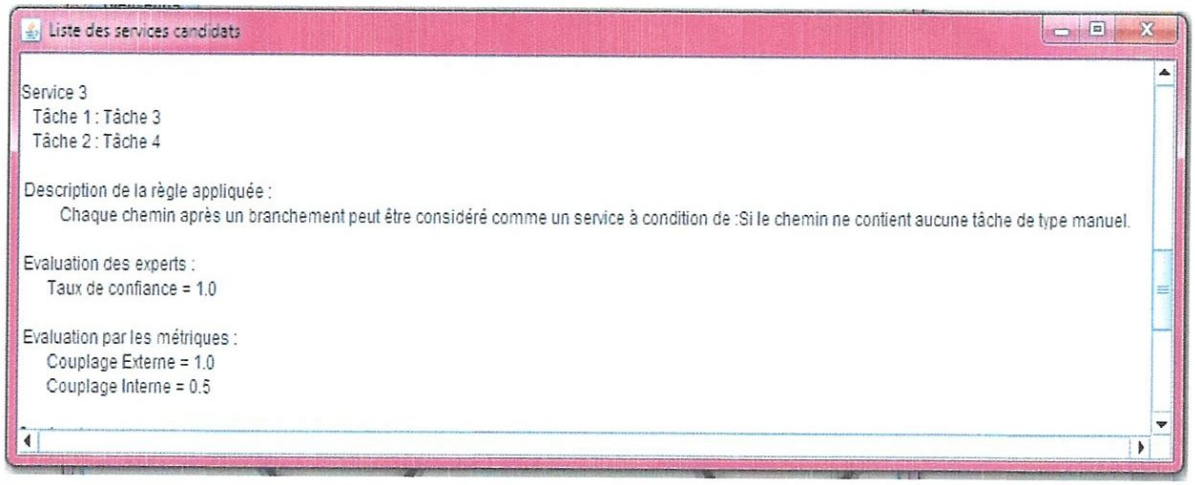


Figure 4.17: *Exporter un fichier texte.*

6. Conclusion :

Dans ce chapitre nous avons présenté les différents modules utilisés pour développer notre Framework d'identification des services, nous avons aussi présenté le fonctionnement général de notre application.

Des imprimes écrans ont aussi été insérés afin de montrer les différentes options de notre projet.

Conclusion générale

Conclusion générale et perspectives :

L'architecture orientée service n'a cessé depuis la dernière décennie de gagner de plus en plus de terrain, vu les avantages qu'elle apporte aux entreprises dans le domaine de l'interopérabilité entre systèmes hétérogènes et surtout de l'agilité des systèmes d'informations face aux changements continus. Malheureusement il n'y a pas encore une méthode qui fait le consensus et qui permet de conduire un projet complet qui démarre de la spécification des besoins jusqu'à la mise en œuvre d'une application à base de service.

La mise en place de solutions BPM nécessite, par définition, elle aussi une mise à plat des processus. Son objectif est de rationaliser l'utilisation des applications pour la réalisation d'un processus métier. Une architecture SOA permettra une implémentation des applications BPM beaucoup plus simple et rapide, pour de meilleurs résultats.

SOA représente les processus métier comme des services. Le BPM devra intégrer cette notion en orientant ses applications pour assurer une communication optimale et une meilleure flexibilité.

BPM et SOA font donc partie de la même stratégie, offrant une infrastructure ouverte, permettant un changement rapide des applications ou services, dans le but d'obtenir un retour sur investissement tout en assurant des objectifs business.

L'identification des services est l'une des étapes qui fait au moins l'unanimité dans toutes les approches SOA. C'est dans ce domaine que nous avons voulu axer notre contribution à travers l'automatisation de la phase d'identification. En choisissant une approche top-down nous nous sommes mis l'objective d'identifier les services à partir du point de vue de l'entreprise et de ses besoins. Ce point de vue est exprimé à travers des processus métiers qui sont à leur tour modélisés à travers des diagrammes BPMN.

Afin d'atteindre ce but, nous avons conçu et développé un Framework qui permet d'importer les diagrammes BPMN, de les analyser puis d'extraire et d'identifier des services candidats en appliquant des règles que nous avons déduits à partir des caractéristiques et des propriétés des services à savoir l'autonomie , la réutilisabilité, le couplage faible avec l'extérieur du service et la cohésion forte à l'intérieur du service. Ces règles sont déduites de façon empirique et peuvent être mises à jour par les experts à tout moment à travers un gestionnaire de règle et un générateur de règle mis à leur disposition.

Nous avons aussi proposé une évaluation des services identifiés suivant deux méthodes, l'une par les experts qui fixent un taux de confiance pour chaque règle et l'autre par le calcul des métriques de qualité du service et plus particulièrement le couplage.

Comme perspective à ce travail, nous pouvons proposer une fonction plus sophistiquée qui calcul la qualité des services identifiés à partir de plusieurs métriques.

Références bibliographiques

Liste des bibliographies :

[1] Zerabi Somaya, « passage d'une architecture classique vers une architecture orientée», UNIVERSITE DE M'SILA, 13 /09/2012

[2] Youness LEMRABET, « Proposition d'une méthode de spécification d'une architecture orientée services dirigée par le métier dans le cadre d'une collaboration inter-organisationnelle», thèse de doctorat, 7 Juin 2012.

[3] LANANI Sadok, « Une approche BPM (Business Process Managment) par composition d'applications dans le cloud computing», dirigé par *Kazar Okba,* UNIVERSITE MOHAMED KHIDER DE BISKRA.

[4] OUAAR. Hanane, « Un Framework pour la gestion et l'interaction des processus métiers intra et inter-entreprises», Université Mohamed Khider Biskra, 15/07/2010

[5] Stéphanie BAUCHE, Mémoire Master« *la modélisation des processus métiers* », Université Paris Ouest Nanterre La défense

[6] Soumia BENDEKKOUM, « Un processus d'Intégration d'Applications Intra & Inter-Entreprises », 17 / 05 /2009, Dirigé par : *Pr. Mahmoud BOUFAÏDA.*

[7] F.Gonzales SOA: comprendre l'approche orientée service 2005

[8] Fayçel BACHTARZI, « Une approche de composition des Services Web Basée transformation de Graphes»,Université Abdelhamid Mehri Constantine 2,07/12/2014

[9] SINI Ghanima, thèse doctorat "Méthode et outils pour la gestion des workflow - Modélisation ontologique des processus pour l'analyse", 07/03/2013 ;

[10] Marie-Hélène Gentil (Maître de Conférences, Université de Bordeaux),"Management des processus, Bonnes pratiques et retours d'expérience".

[11] SERIDI Meryem,BENRDJEM Ahlam,"Implémentation de l'approche processus dans une organisation étude de cas 'Les moulins Amor BENAMOR'", juin 2016,dirigé par : Mme DJAKHDJAKHA Lynda.

[12] Khouloud Boukadi, « Coopération interentreprises à la demande :Une approche flexible à base de services adaptables», École Nationale Supérieure des Mines –SAINT ETIENNE, 05/11/ 2009

[13] Livre blanc BEA : SOA et virtualisation : quelle complémentarité 2008.

[14] Cyrielle Lablanche, Florens Seine, Sébastien Gastaud, « Les Web Services ». Mémoire de licence, Université de Nice-Sophia, Antipolis, 2004–2005.

[15] Chelaghmia Rima-Nebili Wafa, « Identification des services web à partir d'application orientée objet en utilisant les algorithmes de classification (étude comparative)», Université de 8 Mai 1945 – Guelma -, dirigé par 'Mr. Séridi ali', Juin 2015.

[16] A. Arsanjani. Service-oriented modeling and architecture: How to identify, specify and realize services for your soa. Technical report, IBM, Software Group, 2004.

[17] H. Demirkan, R. J. Kauffman, J. A. Vayghan, H.-G. Fill, D. Karagiannis, and P. P. Maglio. Service-oriented technology and management: Perspectives on research and practice for the coming decade. Electronic Commerce Research and Applications, 7(4):356–376, 2008.

[18] T. Erl. SOA: principles of service design. Prentice Hall, 2007.

[19] Rosane S. Huergo, Paulo F. Pires, and Fl´avia C. Delicato, MDCSIM: A method and a tool to identify services, Université de Federal do Rio de Janeiro, Brazil.

[20] Erl, T.: Service-Oriented Architecture: Concepts, Technology, and Design. Prentice Hall, Englewood Cliffs (2005).

[21] Irene Vanderfeesten, Jorge Cardoso, Hajo A. Reijers," A weighted coupling metric for business process models", Technische Universiteit Eindhoven, Department of Technology Management –Netherlands-; University of Madeira, Department of Mathematics and Engineering -Portugal-.

Liste des webographies :

[W1] JF Couturier, « Systèmes d'information dans les entreprises», Disponible sur : https://cours.etsmtl.ca/mti515/Notes/Cours08/Cours%2008%20BPMN.pdf (consulté le 15 juin 2017).

[W2] approche processus disponible sur: http://www.pqb.fr/platform.php?i&if=33&ch=551# (consulté le 15 juin 2017).

[W3] Modélisation des processus Disponible sur: http://www.piloter.org/process-management/modelisation-processus.htm (consulté le 15 juin 2017).

[W4] Hans BRANDENBURG,"L'APPROCHE PROCESSUS -mode d'emploi-" Disponible sur : http://www.allquality.org (consulté le 15 juin 2017).

[W5] Cédric MORA, SOA : Définition, Utilisation dans le monde de la banque et méthodologie de test, IFIPS - Maison de l'ingénieur - Spécialité informatique 2008,disponible sur http://www.memoireonline.com/12/08/1644/m_SOA--Definition-Utilisation-dans-le-monde-de-la-banque-et-methodologie-de-test2.html (consulté le 15 juin 2017).

[W6] http://deptinfo.unice.fr/~baude/WS/cours_SOA_AO+FB.ppt (consulté le 15 juin 2017).

[W7] https://openclassrooms.com/courses/les-services-web (consulté le 15 juin 2017).

[W8] http://www-inf.it-sudparis.eu/cours/WebServices/Docs/Bob_WS-1.pdf(consulté le 15 juin 2017).

[w9] http://blog.xebia.fr/2009/03/04/soa-du-composant-au-service-le-contrat-standardise/ (Consulté le 15 juin 2017).

Annexe

Algorithme 1: Calcule de couplage interne et externe des services candidats

1. Double Couplage Interne = 0 ;
2. Double Couplage Externe = 0 ;
3. A : la liste des activités de service ;
4. T : la liste des transitions ;
5. M : la liste des messages ;
6. Créer un service Temporaire: sztem;
7. If (sztem.size()==1) { Couplage Externe= 1 ; Couplage Interne =1}
8. Else {
9. For (int i=0; i< A.size() ; i++){
10. String idTache = sztem.A.get(i).id;
11. Couplage Externe= Couplage Externe+connected1 (idTache,T,sztem,M) ;}
12. For (int i=0; i< A.size() ; i++){
13. String idTache = sztem.A.get(i).id;
14. Couplage Interne = Couplage Interne +connected2 (idTache,T,sztem,M) ;}
15. Couplage Externe = Couplage Externe / (nombre des taches du service * (nombre des taches du service -1));
16. Couplage Interne = Couplage Interne / (nombre des taches du service * (nombre des taches du service - 1)) ;}

Algorithme 2: la fonction connected1 (String id, List<transition>T,ServiceTemp S,List<message>M)

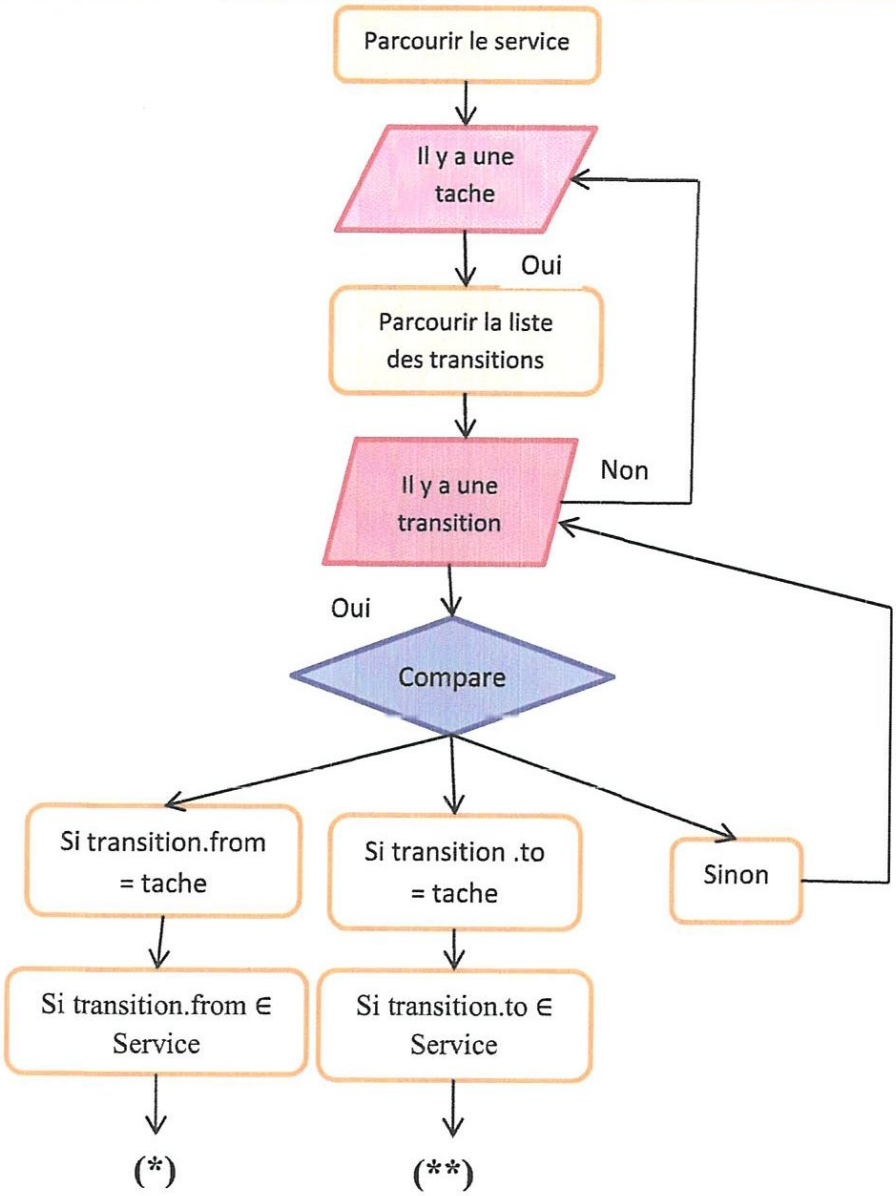
1. int i=0 ; double nbre=0;
2. int m: nombre arcs entrants de service S;
3. int n : nombre arcs sortants de service S;
4. while (i< T.size()){
5. if (id.equals(T.get(i).from.id)) {
6. Activity A= T.get(i).to.id;
7. if(A!ϵ S)){
8. if(T.get(i).to.type.equals("branchement parallele")){
9. nbre++;}
10. else if(T.get(i).to.typc.cquals("branchement inclusif")){
11. nbre+= (1/(m*n));}
12. else if(T.get(i).to.type.equals("branchement exclusif")){
13. nbre+= $\frac{1}{(2^m-1)\cdot(2^n-1)} + \frac{(2^m-1)(2^n-1)}{(2^m-1)\cdot(2^n-1)} \cdot \frac{1}{m \cdot n}$ }
14. else{
15. nbre++;}
16. else if (id.equals(T.get(i).to.id)) {
17. Activity A= T.get(i).from.id;
18. if(A!ϵ S)){

```
19. if(T.get(i).from.type.equals("branchement parallele")){
20. nbre++;}
21. else if(T.get(i).from.type.equals("branchement inclusif")){
22. nbre+= (1/(m*n));}
23. else if(T.get(i).from.type.equals("branchement exclusif")){
24. nbre+= 1/((2^m-1)·(2^n-1)) + ((2^m-1)·(2^n-1)-1)/((2^m-1)·(2^n-1)) · 1/(m·n) }
25. else{
26. nbre++;}
27. i++;}
28. i=0;
29. while(i<M.size()){
30. if (id.equals(M.get(i).from.id)) {
31. Activity A= M.get(i).to;
32. If(A!ϵ S){
33. nbre++;
34. }
35. }
36. else if (id.equals(M.get(i).to.id){
37. Activity A= M.get(i).from;
38. If(A!ϵ S){
39. nbre++;
40. }
41. }
42. i++;}
43. Return  nbre;
```

Algorithme 3: Fonction connected2 (String id,List<Transition> T, ServiceTemp S, List<message> M)

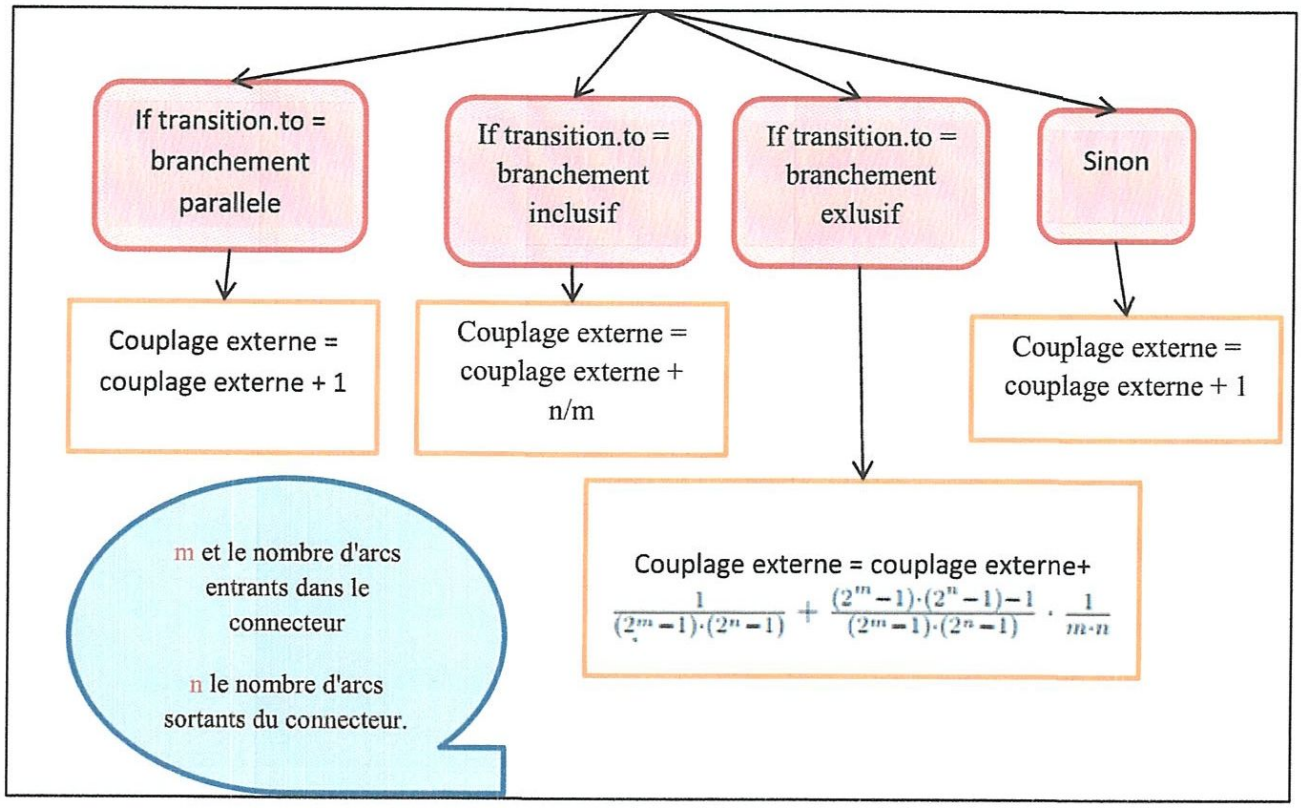
```
1.  int i=0 ; double nbre=0;
2.  int m: nombre arcs entrants de service S;
3.  int n : nombre arcs sortants de service S;
4.  while (i< T.size()){
5.  if (id.equals(T.get(i).from.id)) {
6.  Activity A= T.get(i).to.id;
7.  if( A ϵ S)){
8.  if(T.get(i).to.type.equals("branchement parallele")){
9.  nbre++;}
10. else if(T.get(i).to.type.equals("branchement inclusif")){
11. nbre+= (1/(m*n));}
12. else if(T.get(i).to.type.equals("branchement exclusif")){
13. nbre+= 1/((2^m-1)·(2^n-1)) + ((2^m-1)·(2^n-1)-1)/((2^m-1)·(2^n-1)) · 1/(m·n) }
```

14. else{
15. nbre++;}
16. else if (id.equals(T.get(i).to.id)) {
17. Activity A= T.get(i).from.id;
18. if(A ϵ S)){
19. if(T.get(i).from.type.equals("branchement parallele")){
20. nbre++;}
21. else if(T.get(i).from.type.equals("branchement inclusif")){
22. nbre+= (1/(m*n));}
23. else if(T.get(i).from.type.equals("branchement exclusif")){
24. nbre+= $\frac{1}{(2^m-1)\cdot(2^n-1)} + \frac{(2^m-1)\cdot(2^n-1)-1}{(2^m-1)\cdot(2^n-1)} \cdot \frac{1}{m\cdot n}$ }
25. else{
26. nbre++;}
27. i++;}
28. i=0;
29. while(i<M.size()){
30. if (id.equals(M.get(i).from.id)) {
31. Activity A= M.get(i).to;
32. If(A ϵ S){
33. nbre++;
34. }
35. }
36. else if (id.equals(M.get(i).to.id){
37. Activity A= M.get(i).from;
38. If(A ϵ S){
39. nbre++;
40. }
41. }
42. i++;}
43. Return nbre;

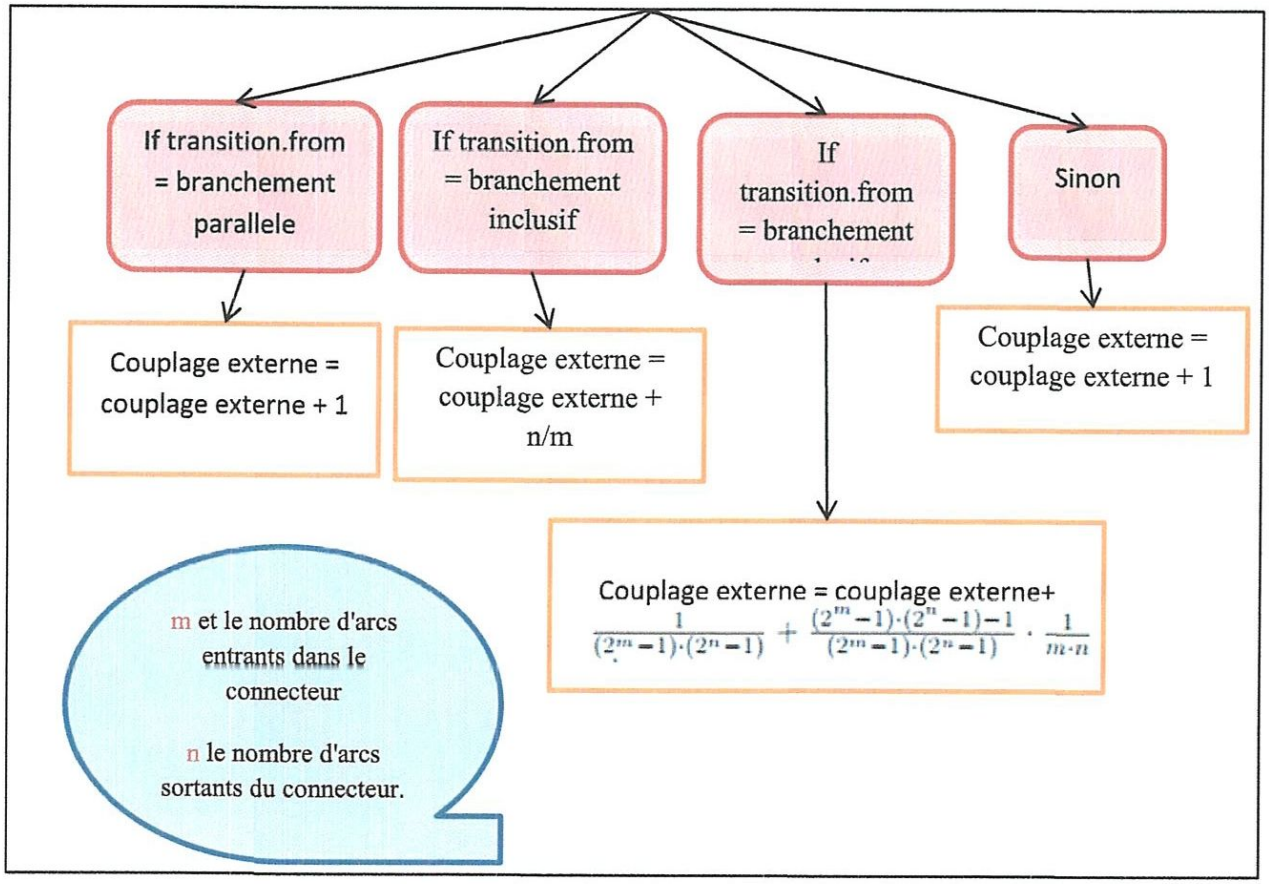


Module pour calculer la fonction ***connected2()***

(*)


If transition.to = branchement parallele
If transition.to = branchement inclusif
If transition.to = branchement exlusif
Sinon
Couplage externe = couplage externe + 1
Couplage externe = couplage externe + n/m
Couplage externe = couplage externe + 1
m et le nombre d'arcs entrants dans le connecteur
n le nombre d'arcs sortants du connecteur.
Couplage externe = couplage externe+ $\frac{1}{(2^m-1)\cdot(2^n-1)} + \frac{(2^m-1)\cdot(2^n-1)-1}{(2^m-1)\cdot(2^n-1)} \cdot \frac{1}{m\cdot n}$

(**)


If transition.from = branchement parallele
If transition.from = branchement inclusif
If transition.from = branchement
Sinon
Couplage externe = couplage externe + 1
Couplage externe = couplage externe + n/m
Couplage externe = couplage externe + 1
m et le nombre d'arcs entrants dans le connecteur
n le nombre d'arcs sortants du connecteur.
Couplage externe = couplage externe+ $\frac{1}{(2^m-1)\cdot(2^n-1)} + \frac{(2^m-1)\cdot(2^n-1)-1}{(2^m-1)\cdot(2^n-1)} \cdot \frac{1}{m\cdot n}$