الجمهورية الجزائرية الديمقراطية الشعبية

République **A**lgérienne **D**émocratique et **P**opulaire

Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

Université **A**bbes **L**aghrour de **K**henchela

Faculté des Sciences et Technologies

Département Maths et Informatique

Filière : Informatique

Spécialité : GLSD (Génie Logiciel et Systèmes Distribués)

MEMOIRE

Présenté pour obtenir le Diplôme de Master

Estimation de RUL par Data mining (Services Web composites)

Réalisé Par :

Slimi Abderahmane

Aoun Mohamed Elamine

Devant le jury :

PRESIDENT :	MCB	Ledmi Makhlouf
RAPPORTEUR :	MCA	Mahdaoui Rafik
EXAMINATEUR :	MAA	Mebarki Djmouai

Année universitaire : 2021 / 2022

Remerciement

Au terme de ce mémoire, nous voudrions d'abord exprimer notre profonde gratitude à notre Dieu, qui nous a donné la force, la volonté, le courage, et surtout la patience pour faire ce travail.

Nous remercions chaleureusement le ***Dr. Rafik Mahdaoui*** qui n'a pas hésité à nous faire confiance et accepté de diriger ce travail. Il nous a fait partager son expérience et ses connaissances. Nous tenons également à le remercier énormément pour ses conseils et son soutien ciblés.

Nous tenons également à exprimer notre profonde gratitude à tous ceux qui ont collaboré de près ou de loin à ce travail.

Dédicace

A nos chères familles :

Père, mère, frères et sœurs.

Á nos amis :

Athman, Oualid, Taki. Oussama…

.

Nous t’aimons !

Résumé :

Dans nos jours, les concepts traditionnels de maintenance préventive et corrective sont peu à peu complétés par de nouvelles formes de maintenance. Un tel type de maintenance pour objectif d'assurer la sûreté de fonctionnement des systèmes industriels et d'augmenter leur disponibilité à moindre coût, l'estimation du RUL (temps résiduel avant défaillance) d'un équipement doit permettre d'éviter des dépenses de maintenance inutiles.

Donc, dans ce travail nous allons modéliser et simuler une application web composite basée sur les services web pour le problème de pronostic industriel. L'objectif est de calculer le RULGlobale, ce qui représente le temps de défaillance de notre système, mais pour l'obtenir, il faut d'abord trouver le temps de défaillance de chaque composant de notre system étudie (RULi locale). La méthode proposée consiste à construire un processus métier BPEL, ce dernier appel des services web externes qui feront le calcul et renvoient le résultat en sortie.

Mots-clés : Services Web, RUL, Maintenance, Pronostic industriel, BPEL, application web composite.

Abstract:

Nowadays, the traditional concepts of preventive and corrective maintenance are gradually being supplemented by new forms of maintenance. This type of maintenance aims to ensure the reliability of industrial systems and to increase their availability at a lower cost, the estimation of RUL (remaining useful life) of an equipment allow to avoid unnecessary maintenance expenses.

So, in this work we will model and simulate a composite web application based on web services for the industrial prognosis problem. The objective is to calculate the RULGlobal, which represents the failure time of our system, but to obtain it, we must first find the failure time of each component of our studied system (RULi local). The proposed method consists in creating a BPEL process, which calls external web services that will do the calculation and return the result as output.

Keywords: web service, RUL, maintenance, industrial prognosis, BPEL, composite web application.

الملخص:

في الوقت الحاضر، يتم استكمال المفاهيم التقليدية للصيانة الوقائية والتصحيحية تدريجياً بأشكال جديدة من الصيانة. يهدف هذا النوع من الصيانة إلى ضمان موثوقية الأنظمة الصناعية وزيادة توافرها بتكلفة أقل، يسمح تقدير الوقت المتبقي قبل الفشل (RUL) للمعدات بتجنب نفقات الصيانة الغير ضرورية.

لذلك، في هذا العمل، سنقوم بنمذجة ومحاكاة تطبيق ويب مركب بناءً على خدمات الويب لمشكلة التنبؤ الصناعي. الهدف هو حساب RULGlobal والذي يمثل وقت فشل نظامنا، ولكن للحصول عليه، يجب علينا أولاً العثور على وقت فشل كل عنصر من عناصر نظامنا المدروس (RULi locale). الطريقة المقترحة هي باستعمال BPEL، هذا الأخير يقوم باستدعاء خدمات الويب الخارجية التي ستقوم بالحساب وإرجاع النتيجة كمخرجات.

الكلمات المفتاحية: خدمات الويب، الوقت المتبقي قبل الفشل، الصيانة، التنبؤ الصناعي، BPEL, خدمات الويب المركبة.

Table des matières

Liste des figures

Liste des tableaux

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

AFNOR Association Française de NORmalisation

RUL Remaining Useful life

IA Intelligence Artificielle

RN Réseaux de Neurones

RMI Remote Method Invocation

CORBA Common Object Request Broker Architecture

QOS Quality Of Service

XML Extensible Mark-up Language

SOAP Simple Object Access Protocol

RPC Remote Procedure Call

WSDL Web Services Description Language

W3C World Wide Web Consortium

UDDI Universal Description Discovery and Integration

BPEL Business Process Execution Language

BPEL4WS Business Process Execution Language for Web Services

WSFL Web Services Flow Language

UML Unified Modeling Language

XSD XML Schema Definition

JBI Java Business Integration

URI Uniform Resource Identifier

Introduction Générale

Avec la diffusion d'Internet, les services Web sont devenus indispensables dans presque tous les domaines. Il s'agit d'une classe de solutions Web utilisées dans des secteurs tels que les usines, les banques et les télécommunications. Un service Web peut être simplement décrit comme tout service fourni sur le Web. C'est ce qui nous a fait le choisir comme solution au problème du pronostic industriel, qui est aujourd'hui l'un des enjeux les plus importants de la sûreté de fonctionnement des systèmes industriels et de la recherche pour améliorer leur disponibilité à moindre coût.

Le mémoire est structuré de la manière suivante :

Le premier chapitre : des définitions de notation de base et nous allons parler sur les approches de pronostic industriel.

Le deuxième chapitre : nous allons parler sur les services Web globalement les caractéristiques et la composition des service web.

Le troisième chapitre : Ce chapitre basé sur la composition des service web**.** Nous allons utiliser un service Web qui calcule le RULi local pour trouver le temps de défaillance de chaque composant de notre système, puis nous pourrons calculer le RULGlobal en appelant un autre service Web. Tout cela est fait par le processus BPEL.

CHAPITRE : 1

Pronostic Industriel et RUL

1.1 Introduction

L'objectif principal du pronostic est d'estimer le temps résiduel avant défaillance d'un équipement (RUL, Remaining Useful life), indicateur permettant d'optimiser les stratégies de maintenance. Dans ce chapitre on va voir les trois approches de pronostic proposées dans la littérature : les approches basées sur un modèle physique, les approches guidées par les données et les approches basées sur l'expérience.

1.2 Définitions

- **Défaillance**

« Cessation de l'aptitude d'un bien à accomplir une fonction requise ».

Une défaillance peut être partielle ou complète. La norme AFNOR propose une corrélation entre les types de défaillances et les types d'interventions (Figure 1). Un système peut remplir sa fonction tout en présentant une anomalie de comportement. Par exemple, une machine électrotechnique peut produire un bruit anormal tout en entraînant correctement une charge, en supposant que telle soit sa fonction. Le bruit anormal est un défaut qui peut permettre de présager d'une défaillance à venir. [1]

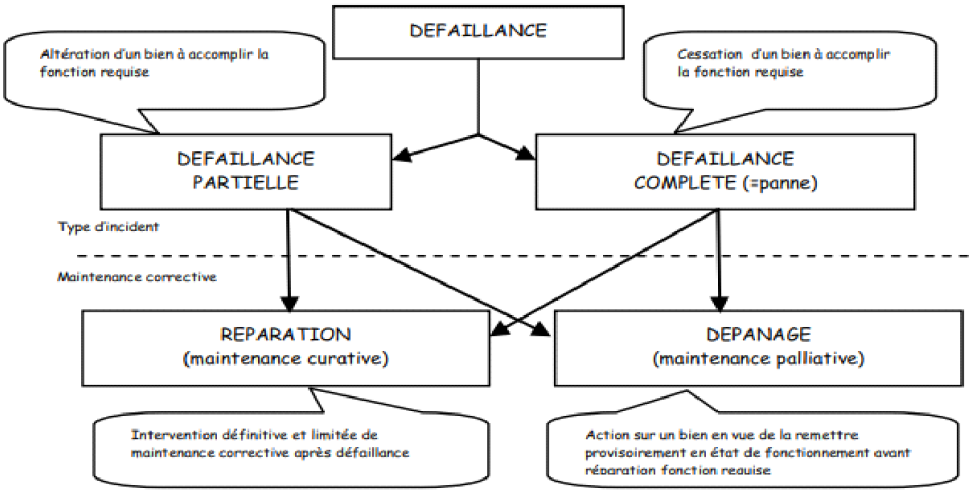


Figure 1.1 - Les défaillances d'après (AFNOR,2001)

- **Dégradation**

« Modification irréversible d'une ou plusieurs caractéristiques de la marchandise liée au temps, à la durée d'utilisation ou à des causes externes ». Si les performances passent en dessous du seuil d'arrêt défini dans la spécification fonctionnelle de l'appareil, il ne s'agit plus d'une dégradation mais d'un échec.

- **Panne**

« État d'un bien inapte à accomplir une fonction requise, excluant l'inaptitude due à la maintenance préventive ou à d'autres actions programmées ou à un manque de ressources extérieures ».

Une défaillance est un événement à distinguer d'une panne qui est un état [1]. En termes de temps, la panne correspond à une date, et la panne correspond à la durée entre la date de la panne et la fin de la réparation . Sur la base de ces concepts de défaillance, de dégradation et de défaillance, la maintenance et la fiabilité peuvent être localisées.

- **Un défaut**

Est tout écart entre la caractéristique observée sur le dispositif et la caractéristique de référence lorsque celui-ci est en dehors des spécifications.

- **RUL**

Remaining Useful Life time, en anglais représente la durée de vie utile restante avant la défaillance ou dans certains systèmes on définit comme la probabilité pour que le système fonctionne durant un certain temps.

1.3 PRONOSTIC

Le terme pronostic provient du grec progignôskein qui signifie "connaître à l'avance"
La norme internationale ISO 13381 de 2004 définit le pronostic comme l'estimation de la durée de vie utile et du risque d'existence ou d'apparition ultérieure d'un ou de plusieurs modes de défaillance. Cependant, dans la littérature, les définitions associées au concept du pronostic s'adaptent au contexte, aux objectifs et au domaine d'application considéré, parmi ces interprétations on trouve:

Selon [2] : In the industrial and manufacturing areas, prognosis is interpreted to answer the question : what is the remaining useful lifetime of a machine or a component once an impending

failure condition is detected and identified : Dans les domaines de l'industrie et de la fabrication, le pronostic est interprète pour répondre à la question : quelle est le temps de vie résiduel d'une machine ou d'un composant lorsqu'un début de défaillance est détectée et identifiée

Selon [3] : Prognosis is an assessment of the future health : Le pronostic est une évaluation de l'état de santé futur.

Selon [4] : Prognostics is the ability to perform the future of an item from its present, its past, its degradation laws and the maintenance actions to be investigated : Le pronostic est la capacité d'estimer l'état futur d'un élément à partir de son état présent, son état passé, ses lois de dégradation et les actions de maintenance à faire.

1.4 Classification des approches du pronostic

D'après la Classification de Byington en 2002 [5], il existe trois approches de pronostic : basées sur des modèles physiques, basées sur des connaissances d'experts et guidées par des données, comme illustré dans la **Fig.1.2**. Cette classification met en évidence l'étendue du champ d'application, le coût de mise en œuvre, l'exactitude et la complexité.

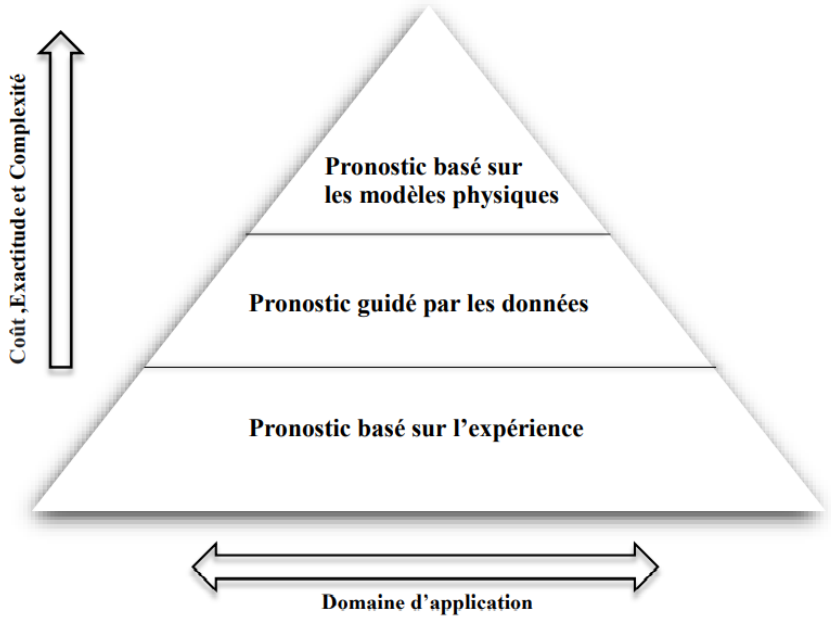


Figure 1.2 - Les trois approches de pronostic

Des méthodes hybrides qui combinent les propriétés des trois approches commencent à émerger dans la littérature, dans le but d'exploiter toute l'information disponible et de répondre aux exigences des industriels.

1.4.1 Approche basée sur le modèle physique

Le pronostic basé sur les modèles [6] fait appel à une fonction mathématique connue de la dynamique de la dégradation, surveillée par un indicateur, conduisant à la défaillance d'un composant. Le résultat de la fonction pronostic implémentant ce type d'approche est le temps de vie résiduelle ou RUL du composant.

La mise en œuvre est assez aisée si l'évolution dans le temps de l'indicateur associé à la dégradation est une fonction monotone. La dégradation est aussi souvent fonction du mode de fonctionnement des composants [7]. Ainsi, lorsqu'un changement de mode de fonctionnement intervient dans l'utilisation du système, la fonction de dégradation doit être mise à jour pour une dégradation donnée. Le problème de ce type d'approche vient du fait qu'il faut disposer des indicateurs de surveillance de dégradation et de leur fonction d'évolution pour chaque mode de fonctionnement. Ceci n'est pas toujours possible.

En effet, il apparaît que les mesures émanant d'un dispositif à surveiller ne sont pas forcément pertinentes pour la surveillance de l'état de dégradation du système. Il apparaît donc que l'implémentation d'une fonction de pronostic sur un système doit être établie dès la conception an de pouvoir spécifier et lister les données nécessaires à la mise en œuvre de ce type de résultats.

Des alternatives existent et nécessitent la construction d'un indicateur générique issu de différentes mesures qui mettent en œuvre des techniques de traitement du signal. Ces alternatives peuvent être utiles dans le cas où une surveillance des dégradations doit être établie sur un système déjà existant ne permettant pas d'évolution vers ce type de services ou ne pouvant pas accueillir de capteurs dont les mesures seraient utilisées à des fins de pronostic [8].

1.4.2 Approche guidée par des données

Elles estiment le RUL par l'analyse d'une grande masse de données observées. Grâce au perfectionnement des systèmes de capteurs modernes ainsi que les technologies de traitement du signal et de stockage de données, ces méthodes sont largement utilisées aujourd'hui pour le pronostic. Elles peuvent être divisées en deux catégories : les méthodes de l'Intelligence Artificielle (IA) et les méthodes d'analyse de tendance.

1.4.2.1 Méthodes de l'Intelligence Artificielle

Lorsque les seules informations disponibles sur le système sont les variables mesurables et que l'on ne peut pas utiliser la redondance physique, la technique habituelle est d'apprendre le comportement du système à l'aide de l'historique des données : ce sont les données d'apprentissage. On part du principe qu'une même cause aura toujours les mêmes effets. Ce sont des systèmes du type "boîte noire" qui ont pour principal avantage d'utiliser "une aveugle "les données, sans aucune considération physique. Leur force réside dans l'aptitude à apprendre (par les exemples) et à capturer les relations subtiles entre données, même si ces relations sont inconnues ou difficiles à décrire. Les réseaux de neurones (RN) et les réseaux d'ondelettes (le perceptron multi couches, les réseaux de neurones probabilistes, les réseaux d'ondelettes avec auto organisation, etc.) sont les principales classes d'outils de ce type. Le principal inconvénient des réseaux de neurones réside dans l'acquisition et la couverture des données d'apprentissage.

1.4.2.2 Méthodes d'analyse de la tendance

La gamme des outils utilisés dans cette catégorie inclut les outils de prévision de série temporelles et les modèles de classification multivariable plus perfectionnés ont développé pour l'armée américaine, un système embarqué de pronostic par analyse de tendance. Son fonctionnement repose sur une méthode de régression mise en œuvre sur l'évolution de l'indicateur FOM. Le nombre de points utilisés par la fonction de régression varie pour que la prévision reste cohérente par rapport aux dernières mesures. [9]

1.4.2.3 Principe de fonctionnement

Le pronostic guidé par les données exploite les indicateurs de dégradation ou d'interventions de maintenance délivrés respectivement par les processus de surveillance et d'aide à la décision (par exemple, données calorimétriques de calibrage, données spectrométriques, puissance, vibration et signal, température, pression, débris d'huile, tensions acoustiques de courants).

Le diagnostic situé en amont conditionne le succès du pronostic par sa capacité à fournir une estimation fiable et précise de l'état de santé courant du système et une mise à jour des paramètres des processus de détérioration. Ce type de pronostic se fonde sur l'hypothèse que les caractéristiques statistiques des données sont relativement inchangées à moins qu'un défaut de fonctionnement ne se produise dans le système. La capacité d'adaptation à tout type d'application disposant de données suffisantes en quantité et en qualité représente un point fort pour cette approche de pronostic. En même temps, la mise en œuvre d'une approche "guidée par les données" est relativement simple car elle ne requiert pas la connaissance formelle des mécanismes de dégradation. La capacité à transformer des données bruitées en information pertinentes pour des décisions de diagnostic / pronostic est un autre avantage pouvant être souligné. L'inconvénient principal de ces approches est que leur efficacité est grandement dépendante de la quantité et de la qualité des données opérationnelles de système.[8]

1.4.3 Approches basées sur l'expérience

Le pronostic basé sur l'expérience est basé sur la formalisation des mécanismes physiques détérioration des composants par modèles stochastiques (loi de fiabilité, processus markoviens ou non-markoviens) initiés par connaissances a priori et jugement d'expert.

1.5 Méthodologie pour choisir une approche de pronostic

1.5.1 Avantages et les inconvénients des approches de pronostic

Le tableau 1.1 résume les avantages et les inconvénients des différentes approches.

Les approches de pronostic	Les avantages	Les inconvénients
Approche basée sur le modèle physique	- Précis. - Meilleure performances de pronostic obtenues. - Flexibilité de l'approche. - Interprétable.	- Difficile. - eux d'obtenir le modèle mathématique. - Les phénomènes de dégradation n'est pas exhaustive. - Chaque composant a une propre modèle ou les lois de fiabilité. - Nécessite une connaissance liée au mécanisme de dégradation.
Approche guidée par des données	- Adaptée à tout type d'application instrumenté. - Connaissance des mécanismes de dégradation directement inclue dans les données. - Ne pas nécessite la connaissance de modèles analytique des dégradations.	- Nécessité des scénarios de dégradation pour différentes conditions opérationnelles. - Acquisition et couverture données d'apprentissage
Approche basé sur expérience	- Applicabilité très large. - Relativement simple à mettre en œuvre et pas contenu.	- Nécessite un historique d'expression passée. - Moins précis. - Les données sont généralement incomplètes. - Le temps nécessite à leur collection peut être très important.

Table 1.1 - Les approches de pronostic

1.5.2 Méthodologie de choix [10]

Le choix d'une approche dépend de la disponibilité des données approprié. L'approche physique s'appuie sur des modèles issus des lois physiques ainsi que des modèles de dégradation construits par expérimentation permettant d'obtenir une représentation mathématique du mécanisme de dégradation. Ensuite l'approche basée sur les données utilise directement les données mesurées issues de capteurs qui représentent l'état actuel du système. Enfin l'approche de pronostic basée sur l'expérience ne nécessite que l'historique des défaillances afin de déterminer la probabilité de défaillance à un moment donné dans le futur.

Les trois approches ont pour but de donner une estimation de RUL. On propose la méthodologie de choix de l'approche de pronostic illustré en **figure 1.3**.

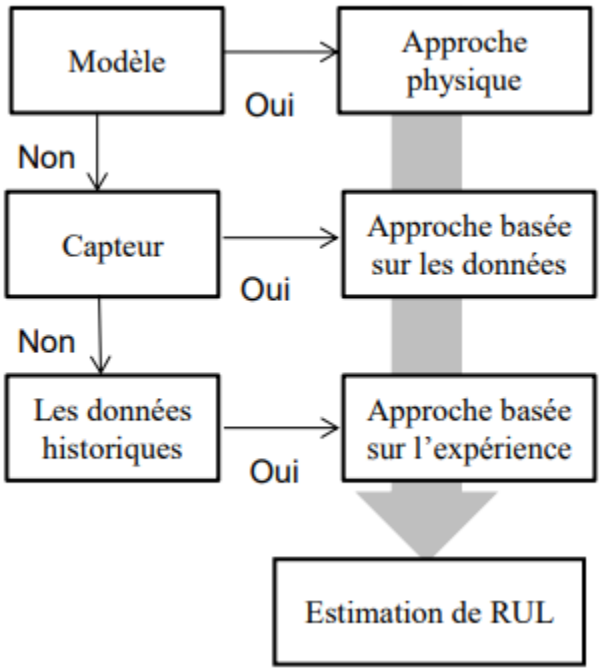


Figure 1.3 - La méthodologie du choix d'une approche

1.6 Conclusion

Dans ce chapitre nous vue les concepts de base lie au temps moyen avant défaillance ou Remaining Useful Life (RUL), en plus les trois méthodes du modèle de système de pronostic industriel. On savait aussi que les méthodes hybrides avaient combiné les propriétés des trois approches, dans le but d'exploiter toutes les informations disponibles et de répondre aux exigences industrielles.

CHAPITRE : 2

Services Web et La composition

2.1 Introduction

L'utilisation de systèmes informatiques a été largement utilisée dans ces applications, scientifiques, manufacturières et commerciales. L'idée principale des systèmes informatiques distribués est d'utiliser une architecture client-serveur en combinaison avec l'appel de méthode à distance. L'invocation de ces méthodes consiste à envoyer des données d'un client à un serveur pour traitement, puis le serveur renvoie des données au client pour présentation. En général, les plates-formes distribuées Java-RMI (Remote Method Invocation), CORBA [16] (Common Object Request Broker Architecture). Ces modèles sont complexes, peu compatible et difficilement interopérables entre eux. Ils restent donc souvent utilisés en Intranet [17].

Les services Web fournissent une gamme de méthodes pour communiquer avec des objets distribués sur des ordinateurs distants et d'en récupérer les résultats à travers le web.

2.2 Définition d'un service Web

Plusieurs définitions des services web ont été proposées. Nous citons celles de W3C, d'IBM et de Wikipédia. [11]

Le groupe de W3C qui travaille sur les services Web a défini le service Web comme « un système logiciel destine à supporter l'interaction ordinateur-ordinateur sur le réseau. Il a une interface décrite en un format traitable par l'ordinateur (e.g. WSDL). Autres systèmes réagissent réciproquement avec le service web d'une façon prescrite par sa description en utilisant des messages SOAP, typiquement transmis avec le protocole http et une sérialisation XML, en conjonction avec d'autres standards relatifs au web ».[13]

Selon [14] : « Un service web est une agrégation de fonctionnalités publiées pour être utilisées. Il utilise internet comme conduit pour réaliser une tâche. Il est semblable à un processus métier virtuel qui définit des interactions au niveau application ».

Selon IBM [15] : « Les services web sont la nouvelle vague des applications Web. Ce sont des applications modulaires, auto-contenues et auto descriptive qui peuvent être publiées, localisées et invoquées depuis le web. Les services web effectuent des actions allant de simples requêtes à des processus complexes. Une fois qu'un service Web est déployé, d'autres applications peuvent le découvrir et l'invoquer ».

2.3 Caractéristiques des services Web

Chaque service web a deux types de caractéristiques [12] :

2.3.1 Les caractéristiques fonctionnelles

Les caractéristiques fonctionnelles d'un service web désignent les opérations qu'il peut fournir. Elles sont décrites dans la description de service en termes d'opérations et reflètent le fonctionnement du service.

2.3.2 Les caractéristiques non fonctionnelles

Les caractéristiques non fonctionnelles d'un service web appelées aussi : qualités de service QoS (Quality Of Service) définissent les capacités de service à fonctionner dans de bonnes conditions en termes de disponibilité, performance, etc. QoS est un ensemble de propriétés opérationnelles du service que l'on doit constater dans la réalisation de service. Formalisées dans le contrat, ces propriétés représentent un ensemble d'exigences concernant la mise en œuvre du service et constituent donc un engagement de niveau de service.

Nous donnons ici une liste des propriétés classées par en suivant les paramètres définis par le W3C :

- **Le coût (ou le prix) :** désigne le prix que le client doit payer pour chaque invocation du service.
- **La latence (aussi connue comme le temps de réponse) :** désigne le temps pris par un service pour répondre à une requête.
- **Le débit :** le débit d'un service représente le nombre de demandes que le service est en mesure de traiter dans un intervalle de temps donné.
- **La fiabilité (aussi connue comme le taux d'exécution avec succès) :** désigne la capacité du service à accomplir sa fonction correctement pendant une période de temps spécifiée. Il peut être mesuré par le temps moyen entre pannes.
- **La disponibilité :** ce paramètre désigne la probabilité que le service soit actif. Il peut être exprimé par le ratio : période active / période totale. La période active représente le temps où le service est opérationnel en répondant aux demandes des clients. La période totale (période active + période inactive), elle désigne la durée durant laquelle le client souhaite que le service soit actif.

- **La sécurité :** la sécurité d'un service web comporte différents aspects assurant que les échanges de messages entre le client et le service soient sécurisés. Plus précisément c'est un regroupement d'un ensemble de qualités à savoir : la confidentialité, le cryptage des messages et le contrôle d'accès.
- **La réputation :** c'est une mesure de la crédibilité du service. Elle dépend principalement des expériences d'utilisateurs finaux.

Ces paramètres peuvent être classés, selon leur mesure, en trois catégories [12] :

- Métriques annoncées par le fournisseur : métriques dont les valeurs ont été annoncées par le fournisseur. Exemple : le coût d'un service.
- Métriques évaluées par le consommateur : métriques dont les valeurs ont été données par l'utilisateur. Exemple : la réputation.
- Métriques observables : métriques dont les valeurs ont été obtenues par la surveillance ou le test. Comme exemple : le temps de réponse ou la disponibilité.

2.4 Les principales technologies des services Web

Le terme technologies de services web désigne un ensemble de technologies basées sur des standards ouverts (non propriétaires), essentielles pour le transport et la transformation des données d'un client vers un service d'une façon standard. Les plus courants sont HTPP, XML, SOAP, WSDL, UDDI [18].

2.4.1 XML (Extensible Mark-up Language)

Le langage de balisage extensible (XML) est un format texte simple pour représenter des informations structurées : documents, données, configuration, livres, transactions, factures, et bien plus encore. Il a été dérivé d'un ancien format standard appelé SGML (ISO 8879), afin d'être plus adapté à une utilisation Web.

Aujourd'hui XML est l'un des formats les plus utilisés pour partager des informations structurées : entre programmes, entre personnes, entre ordinateurs et personnes, à la fois localement et entre réseaux [18].

2.4.2 SOAP (Simple Object Access Protocol)

SOAP est un protocole d'invocation de méthodes sur des services distants. Basé sur XML, SOAP a pour principal objectif d'assurer la communication entre machines. Le protocole permet d'appeler une méthode RPC (Remote Procedure Call) et d'envoyer des messages aux machines distantes via HTTP. Il utilise d'autres protocoles tels que SMTP, ainsi que différents formats 10 comme MIME. Ces derniers sont très répandus sur plusieurs plates-formes, ce qui donne à SOAP une grande portabilité et interopérabilité. Ce protocole est très bien adapté à l'utilisation des services Web, car il permet de fournir au client une grande quantité d'informations récupérées sur un réseau de serveurs tiers. SOAP est bien plus populaire et utilisé que XML-RPC. C'est une recommandation du W3C. D'après cette recommandation, SOAP est destiné à être un protocole léger dont le but est d'échanger des informations structurées dans un environnement décentralisé et distribué. Une des volontés du W3C vis-à-vis de SOAP est de ne pas réinventer une nouvelle technologie. SOAP a été construit pour pouvoir être aisément porté sur toutes les plates-formes et les technologies existantes [18].

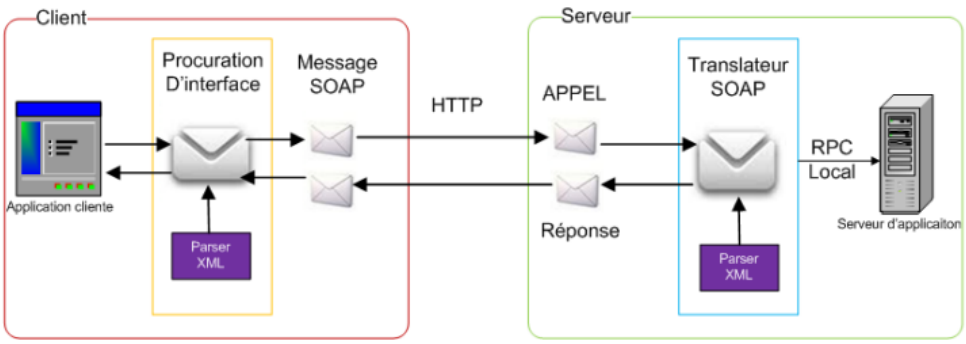


Figure 2.1 - Communication client-serveur à travers des messages SOAP

2.4.2.1 Structure d'un message SOAP

Un message SOAP est un document XML constitué d'une enveloppe, contenant un entête (facultatif) et le corps du message (voir figure ci-dessous)

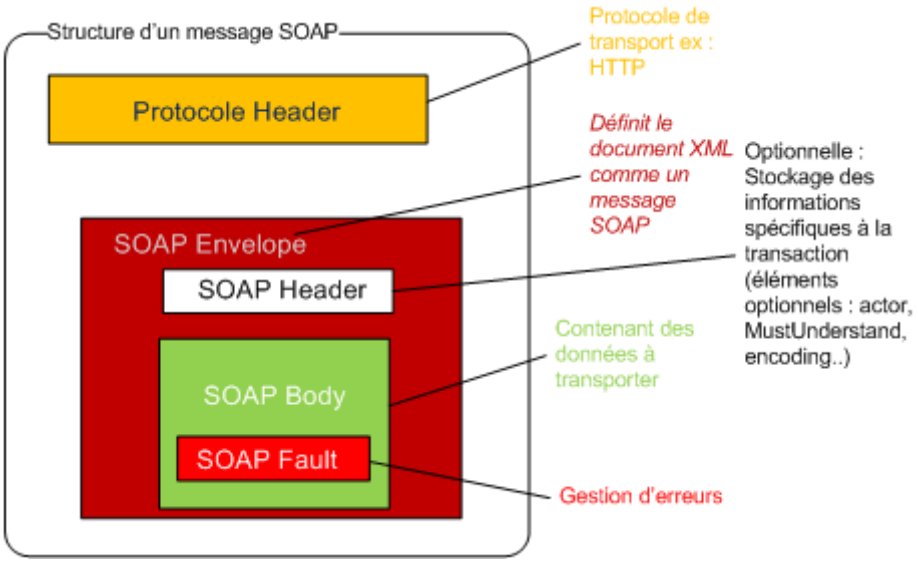


Figure 2.2 - Structure d'un message SOAP

2.4.3 WSDL (Web Services Description Language)

WSDL (Web Services Description Language) est un langage de description fondé sur XML (Extensible Markup Language). Il a été soumis au W3C (World Wide Web Consortium) comme standard industriel pour la description des services Web. WSDL tire sa puissance de deux principes architecturaux principaux : la possibilité de décrire un ensemble d'opérations métier et la possibilité de séparer la description entre deux unités de base. Ces unités fournissent une description des opérations et les détails du mode d'intégration de l'opération et des informations qui lui sont associées. [19]

Un document WSDL définit des services en tant que collections de points de contact de réseau ou de ports. Dans WSDL, les définitions abstraites des points de contact (end points) et des messages sont séparées de leur déploiement réseau concret ou des liaisons de format de données. Ainsi, la réutilisation des définitions abstraites est possible : messages, qui sont des descriptions abstraites des données échangées et types de ports, qui sont des collections abstraites d'opérations. Les spécifications de format de données et de protocole concret pour un type de port particulier constituent une liaison réutilisable. Un port est défini en associant une adresse réseau à une liaison réutilisable et une collection de ports définit un service. C'est pourquoi, un document WSDL est composé de plusieurs éléments. [19]

La structure d'informations dans un fichier WSDL est la suivante :

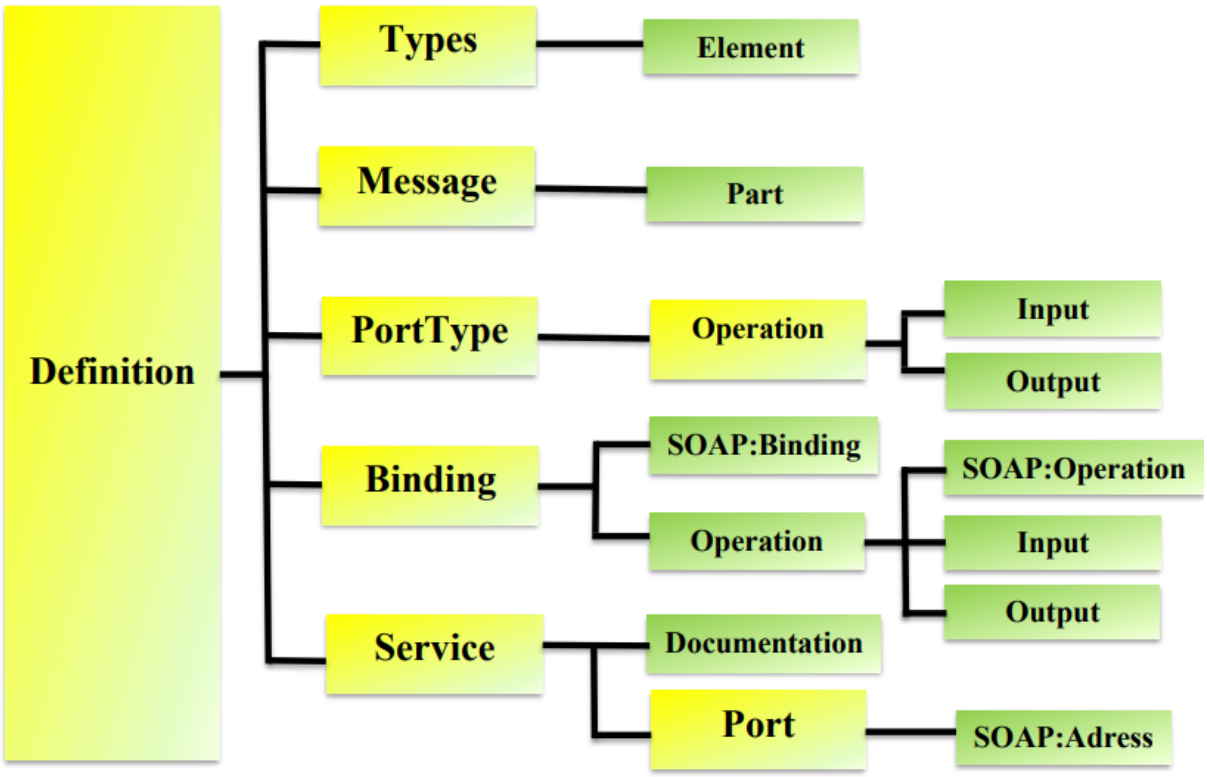


Figure 2.3 - Structure d'un document WSDL

Une description WSDL d'un service Web est faite sur deux niveaux, niveau abstrait et niveau concret [20] :

- La description abstraite : Elle définit d'une façon abstraite l'interface du service, c'est à dire les informations qui sont indépendantes d'un contexte de mise en œuvre.

  ***Définitions** Cet élément contient la définition du service. C'est la racine de tout document WSDL. Il définit le nom du service Web et contient tous les éléments qui décrivent le service web.

  ***Types** Définition des types de données utilisés lors de l'échange sous la forme d'un schéma XML.

*** Message** Permet la définition des structures de données et leur contenu qui vont être échangées dans l'utilisation du service (données à transmettre, valeur de retour).

*** PortType** Il définit l'interface du service qui permet la définition abstraite d'un ensemble d'opérations offertes par un service Web. Chaque opération fait référence à un message d'entrée et à des messages en sortie. Il y a quatre types d'opération : one-way, request-response, solicit-response et notification.

*** Opérations** Est une description abstraite d'une fonction supportée par le service Web. C'est à dire la description d'une action exposée dans le port.

❖ La description concrète : Elle désigne la définition de l'implémentation d'un service donné, c'est-à-dire les informations liées à un usage contextuel du service.

*** Binding :** Spécifie une liaison d'un à un protocole concret (SOAP, HTTP, MIME, . . .). Un peut avoir plusieurs liaisons.

*** Service :** Cet élément précise les informations complémentaires nécessaires pour invoquer le service. Il contient plusieurs éléments 'ports', contenant chacun un nom, une URL de point d'accès et une référence à une liaison donnée.

*** Port :** Adresse réseau d'une liaison définissant le point d'accès associé à un service.

Inconvénients de WSDL

— Il est difficile d'apprentissage, car c'est un langage de balises.
— Il n'est pas extensible et il est peu expressif.
— Il ne définit pas la logique des composants de service, il ne prend donc pas en charge la comparaison de services.

2.4.4 UDDI (Universal Description Discovery and Integration)

UDDI, a été conçu en 2000 à l'initiative d'un ensemble d'industriels (Ariba, IBM, Microsoft), en vue de devenir le registre standard de la technologie des services Web. Pour convenir à la technologie des services Web, les services référencés dans UDDI sont accessibles par l'intermédiaire du protocole de communication SOAP, et la publication des informations concernant les fournisseurs et les services doit être spécifiée en XML afin que

la recherche et l'utilisation soient faites de manière dynamique et automatique. UDDI est une spécification définissant la manière de publier et de découvrir les services Web sur un réseau [20].

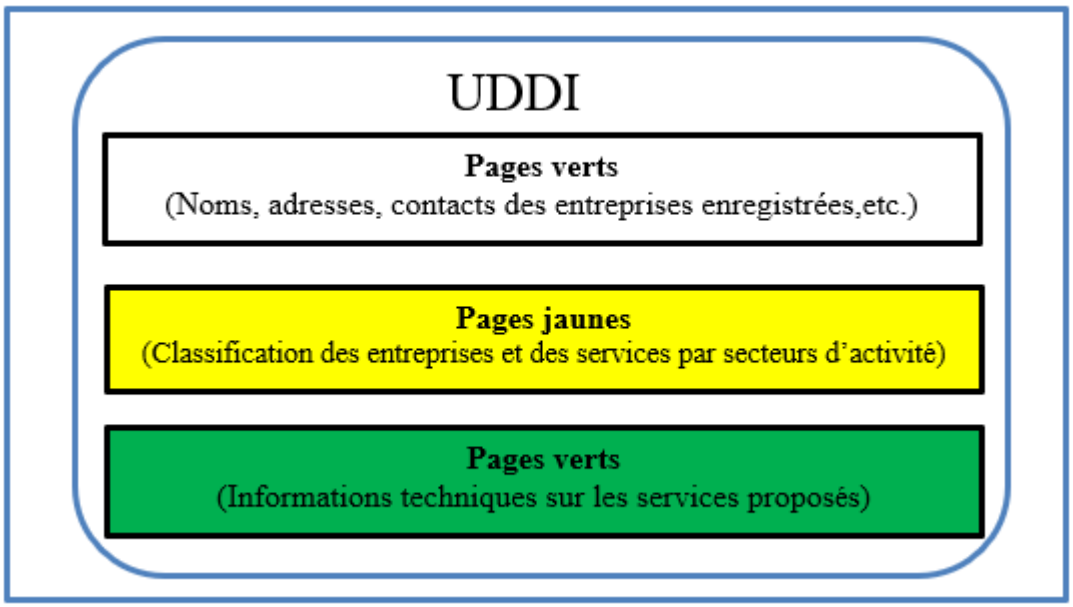


Figure 2.4 - les trois types de l'annuaire UDDI

Les informations qu'il contient peuvent être divisées en trois catégories :

Les Pages blanches comprennent des adresses, des informations de contact et des identifiants liés aux services Web.

Les Pages jaunes qui identifient les domaines d'activité liés aux services Web.

Les pages vertes fournissent des informations techniques précises sur les services fournis.

2.5 Les Avantages et les inconvénients des services Web

2.5.1 Avantages des services Web [18]

Aujourd'hui, les services Web joue un rôle majeur dans les secteurs d'administration de serveur et de développement car ils offrent beaucoup d'avantages tels que :

- Les services Web permettent des programmes écrits dans différentes langues et sur des plateformes différentes de communiquer entre elles en utilisant certaines normes.

En d'autres termes, les services Web permettent une meilleure interopérabilité entre les logiciels.

- Les services Web utilisent des normes et des protocoles ouverts tel que SOAP et HTTP.

- Basés sur le protocole HTTP, les services Web peuvent fonctionner via de nombreux pare-feu sans nécessiter de modifications des règles de filtrage.

- Les outils de développement, basés sur ces standards, permettent la création automatique de programmes utilisant les services web existants.

- Les services sont conçus de façon à ce qu'ils puissent être réutilisés ultérieurement pour minimiser la redondance et rendre le service réutilisable par les différentes applications du système d'information.

- Les couts réduits grâce aux services Web (automatisation interne et externe des processus).

2.5.2 Inconvénients des services Web

- Manque de sémantique.

- Le langage de description (WSDL) est classique.

2.6 Composition des Web services

Dans cette section, nous présenterons la composition des services Web et leurs concepts.

2.6.1 Définition

Une composition de services est un ensemble de techniques d'assemblage Services Web pour l'exécution de processus métier. Le résultat de la composition services peuvent être un nouveau service. Ce nouveau service est appelé service composite et les services invoqués sont des composants de service.

2.6.2 Objectifs de la composition

L'objectif principal est d'offrir des services intégrés à valeur ajoutée en combinant des services existants pour résoudre les problèmes provient généralement des sources suivantes :

- Les processus métiers se complexifient :

  - Plusieurs applications à intégrer.

- Fort besoin de paralléliser les processus.

- De nombreux partenaires pour l'intégration.

- Hétérogénéité des langages due aux plateformes de développement : Java, .NET, PHP…etc.
- Fort besoin d'évolution des processus : Intégration de nouveaux processus.

2.6.3 Orchestration versus Chorégraphie

Les services Web exposent généralement les opérations de certaines applications ou systèmes d'information. Par conséquent, combiner plusieurs services Web implique en fait l'intégration des applications sous-jacentes et de leurs fonctionnalités.

Les services Web peuvent être combinés de deux manières :

- Orchestration
- Chorégraphie

2.6.3.1 L'orchestration des services web [12]

Il permet de décrire l'enchaînement des services selon une séquence prédéfini (un processus avec conditions et exceptions). Ensuite, un coordinateur central (l'orchestrateur) est ensuite créé pour implémenter cette séquence. Il prend le contrôle de tous les services web impliqués et coordonne l'exécution des différentes opérations des services composants qui participent dans le processus (voir figure 2.5) :

- Pour des compositions de services simples, l'orchestration est faite dans le code (Java, C#...) résidant dans le composite.
- Pour des orchestrations complexes, des outils sont utilisés pour créer un modèle visuel d'une séquence et générer le code qui exécute cette séquence dans un environnement d'exécution dédié.
- Des éditeurs graphiques dédiés à la conception de la séquence sont généralement intégrables dans les environnements de développement (Eclipse, NetBeans, Visual Studio, …) et le moteur d'orchestration BPEL est intégré dans la majorité des serveurs d'application.

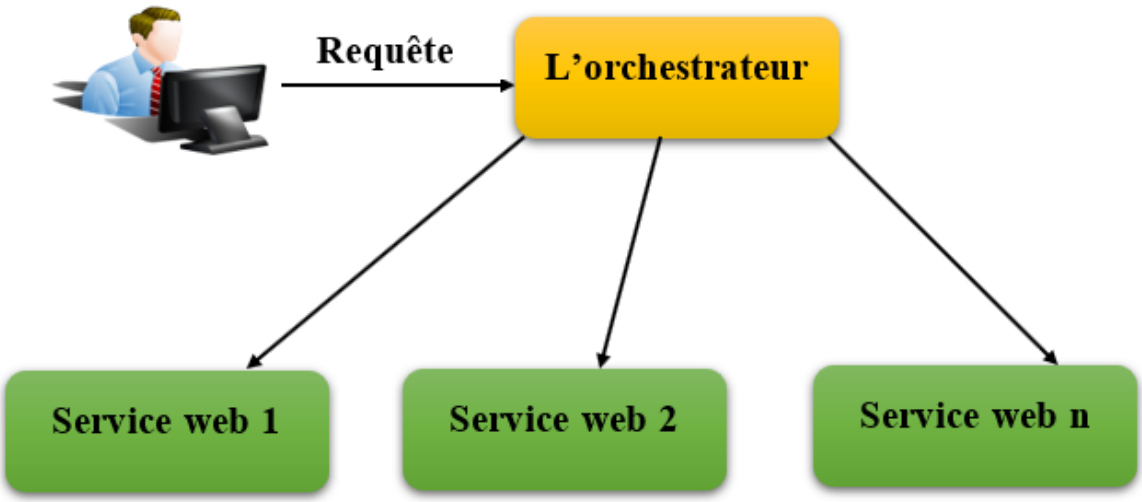


Figure 2.5 - Orchestration des services web

2.6.3.2 Chorégraphie des services web

Contrairement à l'orchestration, La Chorégraphie est un effort de collaboration dans lequel chaque participant du processus décrit l'itération qui l'appartient. (Voir figure 2.6).

Le comportement global du processus est basé sur l'interaction des différentes parties, chacune suivant son propre rôle de manière autonome.

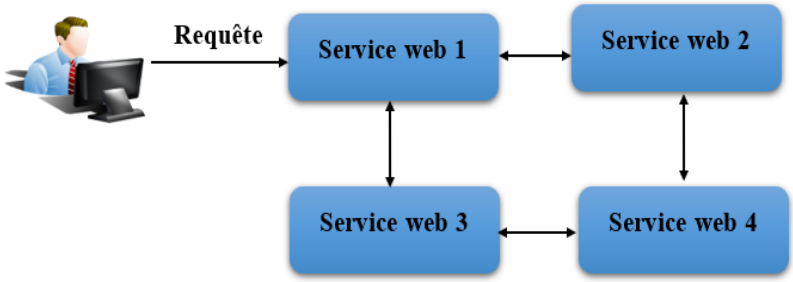


Figure 2.6 - Collaboration entre les services web.

2.6.4 Cycle de vie de la composition

La composition est un processus relativement complexe. Ça passe par plusieurs étapes. Nous montrons un cycle de vie dans la figure 2.7, qui montre la séquence des étapes de composition.

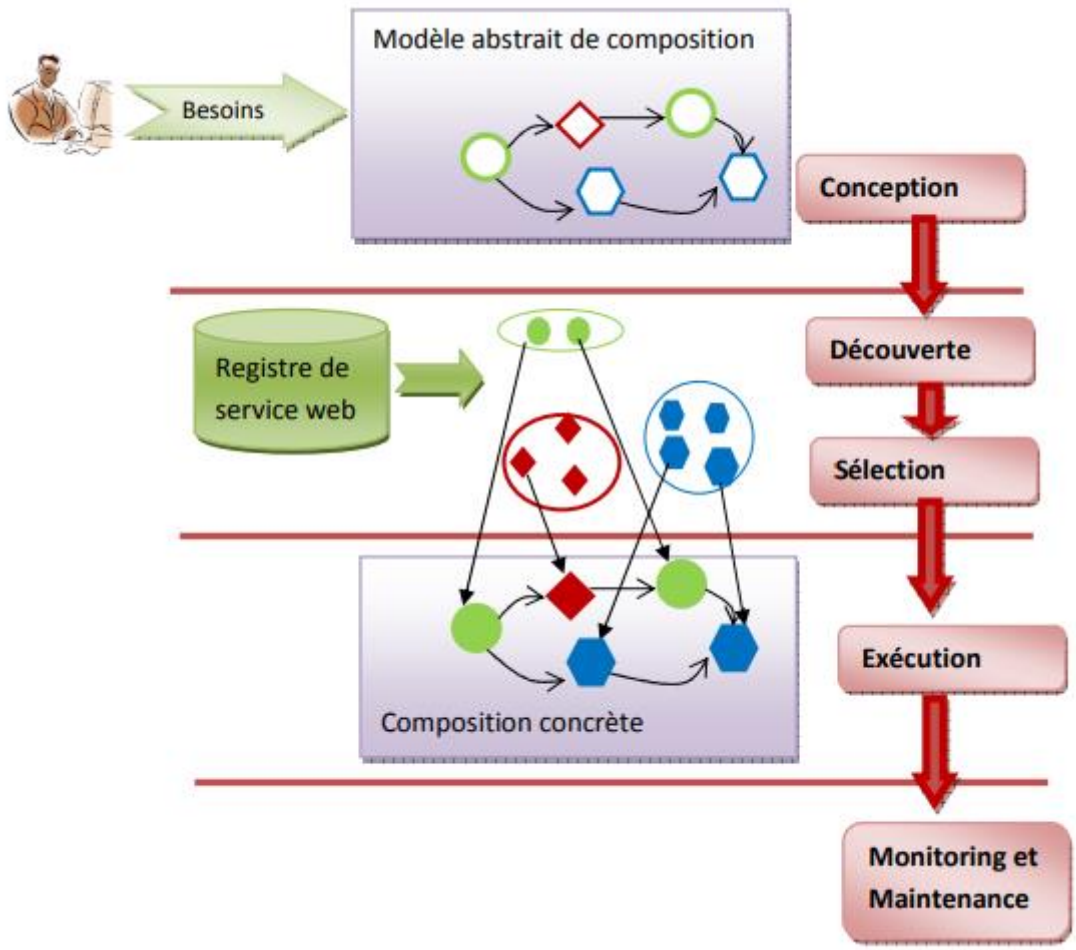


Figure 2.7 - La séquence des étapes de composition.

2.7 Conclusion

Nous avons présenté dans ce chapitre une vision générale sur les services Web. Il représente actuellement l'ensemble de standards les plus connus pour la réalisation des applications Internet. Cette technologie repose sur des standards très populaires comme XML.

Nous avons également vu la composition des service web et les deux manières de combinaison (Orchestration, Chorégraphie).

CHAPITRE : 3

Modélisation et Implémentation

3.1 Introduction

L'objectif principal de ce chapitre est de concevoir une application SOA utilisant l'environnement Netbeans et le serveur Glassfish. Pour cela nous devons Connaître quelques notions. Afin d'atteindre cet objectif nous a allons d'abord en premier lieu expliquer le langage d'exécution de processus métier pour les services web à savoir BPEL. Ensuite, nous allons créer des services partenaires et enfin détailler comment nous allons implémenter notre application dans un service web composite.

3.2 Le langage d'exécution de processus métier pour les services Web

BPEL ou BPEL4WS (Business Process Execution Language for Web Services) est un langage exécutable basé sur XML standardisé par OASIS. BPEL est une fusion de Microsoft XLANG11 et WSFL12 5.3. Règles de transformation IBM BPEL. Il combine les fonctionnalités de Block Structured Process Language (XLANG) et de Business Process-Based Process Language (WSFL). Nous avons choisi d'utiliser BPEL car c'est le langage d'orchestration le plus couramment utilisé dans l'industrie.

3.3 Contexte BPEL

Tout d'abord, un peu de contexte. BPEL est construit sur des services XML et Web, en utilisant des langages basés sur XML qui prennent en charge la pile technologique des services Web, notamment SOAP, WSDL, UDDI, WS-Reliable Messaging, WS-Addressing, WS-Coordination et WS-Transaction.

BPEL représente une fusion de deux premiers langages de workflow ; Web Services Flow Language (WSFL) et XLANG. WSFL a été conçu par IBM et est basé sur le concept de graphes orientés. XLANG est un langage structuré en blocs conçu par Microsoft. BPEL combine ces deux approches et fournit un vocabulaire riche pour décrire les processus métier.

La première version de BPEL a été développée en août 2002. Depuis lors, de nombreux fournisseurs majeurs (y compris Oracle) se sont joints, ce qui a entraîné des modifications et des améliorations, et en mars 2003, la version 1.1 a été adoptée. En avril 2003, BPEL a été soumis à l'Organisation pour l'avancement des normes d'information structurée à des fins de normalisation (OASIS) et le comité technique du langage d'exécution des processus métier des

services Web (WSBPEL TC) a été créé. Cet effort a acquis une reconnaissance plus large dans l'industrie.[21]

3.4 Construire un processus métier

Le processus BPEL spécifie l'ordre exact dans lequel les services Web participants doivent être appelés séquentiellement ou en parallèle. À l'aide de BPEL, vous pouvez exprimer un comportement conditionnel. Par exemple, un appel à un service Web peut dépendre de la valeur d'un appel précédent. Vous pouvez également construire des boucles, déclarer des variables, copier et affecter des valeurs, définir des gestionnaires d'erreurs, etc. En combinant toutes ces structures, vous pouvez définir des processus métier complexes de manière algorithmique. En effet, les processus métiers étant essentiellement des diagrammes d'activités, il est utile de les exprimer à l'aide de diagrammes d'activités UML (Unified Modeling Language).[12]

Dans un scénario typique, un processus métier BPEL reçoit une demande. Pour ce faire, le processus appelle le service Web concerné, puis répond à l'appelant d'origine. Étant donné que le processus BPEL communique avec d'autres services Web, il s'appuie fortement sur la description WSDL du service Web qui compose les appels de service Web. Un processus BPEL se compose d'étapes ; chaque étape est appelée une "activité". BPEL prend en charge les activités primitives et structurées. Les activités primitives représentent des constructions de base et sont utilisées pour des tâches courantes. Comprendre un processus BPEL nécessite de définir un ensemble commun de concepts :

- **Les services partenaires (Partner Services) :** Représente tout service ou client externe qui interagit avec le processus BPEL. Un processus BPEL utilise un ou plusieurs partenaires externes tout au long de son exécution, qui peuvent être des services Web, des bases de données ou d'autres processus BPEL.
- **Les activités :** ce sont les tâches métier individuelles dans le processus, permettant de réaliser un objectif plus large. Ils représentent chaque étape du processus. Un processus BPEL commence toujours avec une activité de réception (receive), puis invoque des services externes (invoke) et enfin renvoient le résultat au client (reply).

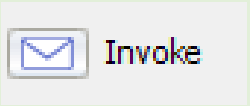
NOM	SYMBOLE	Description
ASSIGN		Exécute différents traitements en parallèle. Permet de manipuler les variables d'un processus ➔Initialisation de variable ➔Copie de variable ➔Manipulation XML (XPath)
FLOW		Exécute différents traitements en parallèle.
Invoke		Invoquer d'autres services Web.
Receive		Attente que le client invoque le processus métier en envoyant un message.
Reply		Génération d'une réponse pour les opérations synchrones.

Table 3.1 - Activites BPEL de base

❖ **Les variables :** De nombreuses variables et messages circulent entre les activités d'un processus et entre les activités et les partenaires. Dans le tableau **3.1** il y a l'activité Assign qui permet d'affecter les éléments de l'itinéraire (v1, v2...) à une nouvelle variable. Cela peut être fait avec une simple opération de copie ou en traitant l'entrée (concaténation, somme, etc.). Cette affectation est représentée par un autre outil NetBeans : BPEL Mapper (voir figure 3.1)

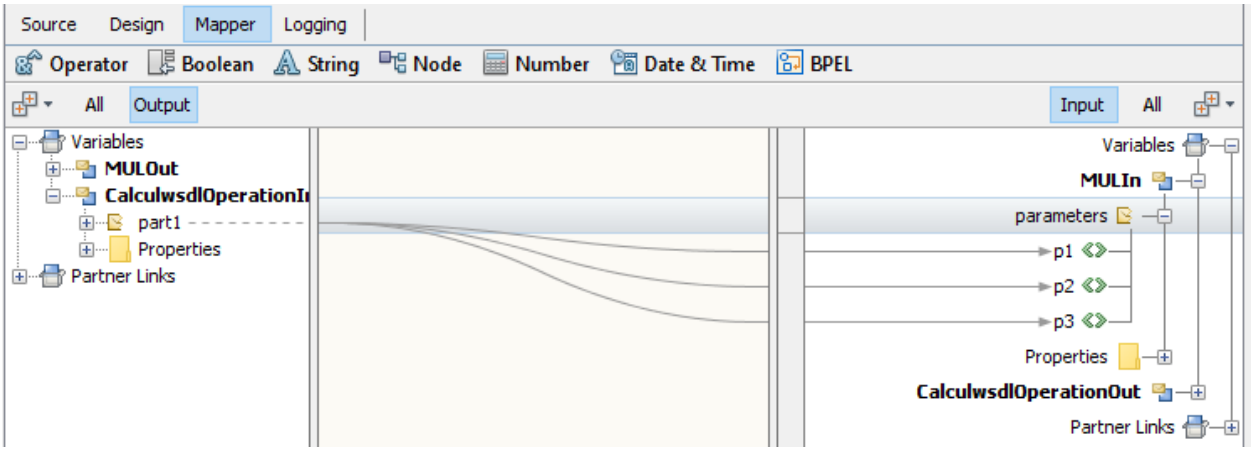


Figure 3.1 - Le Netbeans BPEL Mapper

3.4.1 Plan de conception

Dans cette application, nous allons réaliser les tâches suivantes :

- Un processus métier **BPEL(*Business* Process Execution Language)** permettant la lecture en entrée 16 variables :
  - V1 : Vitesse de rotation du four
  - V2 : Débit d'entrée de la charge du four
  - V3 : Débit de sortie de la charge du four
  - V4 : Température d'entrée de la charge du four
  - V5 : Température de sortie de la charge du four
  - V6 : Débit d'entrée du gaz du four
  - V7 : Débit de sortie du gaz du four
  - V8 : Température d'entrée du gaz du four
  - V9 : Température de sortie du gaz du four
  - V10 : Débit d'entrée du charge cyclone
  - V11 : Débit de sortie du charge cyclone

- V12 : Température d'entrée du charge cyclone
- V13 : Température de sortie du charge cyclone
- V14 : Débit d'entrée du gaz du cyclone
- V15 : Débit de sortie du gaz du cyclone
- V16 : Température d'entrée du gaz cyclone

- Ce processus invoque des service web externe qui va faire le calcule et envoi du résultat en sortie.
- Les services partenaires : sont des services avec lesquelles le processus BPEL **(B*usiness* Process Execution Language)** va interagir.
- Ensuite une application composite qui appelle de processus BPEL, nous le donnons en entrée une requête **SOAP (*Simple Object Access Protocol)*** contenant les 16 variables, et recevant en sortie une réponse SOAP **(*Simple Object Access Protocol)*** avec le résultat de calcule.

3.4.2 Création du processus métier de RUL Global

Pour créer notre processus BPEL, nous suivons les étapes suivantes :

1. On va créer un nouveau Projet,
2. Ensuite, on va sélectionner la catégorie SOA puis BPEL Module,
3. On va choisir **BpelRUL** comme nom de projet.

On obtint le résultat suivant :

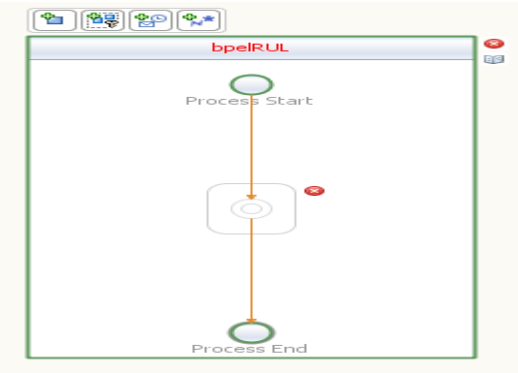


Figure 3.2 - Application BpelRUL : BPEL

3.4.3 Création de fichier XSD (XML Schema Definition)

Les services qui fournissent des données d'entrée au processus seront décrits par WSDL. Mais d'abord, par bonne pratique, nous allons créer un fichier XSD (XML Schema Definition) où nous définissons les types de messages qui seront échangés au cours de ce processus.

Pour créer un nouveau fichier XSD, nous suivons les étapes :

1. On va faire un clic droit sur le répertoire Process Files,
2. On va choisir New XML Schema,
3. Nous le nommons ce schéma **RULXsd**,
4. On va choisir l'onglet Source pour afficher notre schéma,
5. Pour créer des types complexes on va choisir la balise :

   <xsd:complexType name="name1">

   Pour créer des éléments simple on va choiser la balise :

   <xsd:element name="nam1" type="type1"/>

   nous avons besoin deux types complexes **varType** et **minType**.

   - **varType** avec 16 éléments Float .
   - **minType** avec un éléments Float, nous l'appelons (**RUL**).

6. On va créer ensuite deux éléments simples :

   var et **min** de type respectivement **varType** et **minType.**

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://xml.netbeans.org/schema/RULXsd"
            xmlns:tns="http://xml.netbeans.org/schema/RULXsd"
            elementFormDefault="qualified">
    <xsd:complexType name="varType">
        <xsd:sequence>
            <xsd:element name="VRF" type="xsd:float"/>
            <xsd:element name="DECF" type="xsd:float"/>
            <xsd:element name="DSCF" type="xsd:float"/>
            <xsd:element name="TECF" type="xsd:float"/>
            <xsd:element name="TSCF" type="xsd:float"/>
            <xsd:element name="DEGF" type="xsd:float"/>
            <xsd:element name="DSGF" type="xsd:float"/>
            <xsd:element name="TEGF" type="xsd:float"/>
            <xsd:element name="TSGF" type="xsd:float"/>
            <xsd:element name="DECC" type="xsd:float"/>
            <xsd:element name="DSCC" type="xsd:float"/>
            <xsd:element name="TECC" type="xsd:float"/>
            <xsd:element name="TSCC" type="xsd:float"/>
            <xsd:element name="DEGC" type="xsd:float"/>
            <xsd:element name="DSGC" type="xsd:float"/>
            <xsd:element name="TEGC" type="xsd:float"/>
        </xsd:sequence>
    </xsd:complexType >
    <xsd:complexType name="minType">
        <xsd:sequence>
            <xsd:element name="RUL" type="xsd:float"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="var" type="tns:varType"/>
    <xsd:element name="min" type="tns:minType"/>
</xsd:schema>
```

Figure 3.3 - Le fichier XSD final

3.4.4 Création de services partenaires

Ce processus BPEL interagit avec trois services.

1- Un service client qui va fournir les entrées au processus sera décrit par un fichier WSDL (Web Services Description Language).

2- Un service externe qui calcule RUL1, RUL2, RUL3, RUL4.

3- Un service externe qui calcule le minimum RUL.

3.4.4.1 Création de service client

Ce service est décrit par le fichier WSDL, nous suivrons les étapes suivantes pour le créer :

1. On va faire un clic droit sur Process Files,
2. On va choisir New WSDL Document,
3. Nous l'appelons ce fichier **RULWsdl.**

Dans la partie Abstract Configuration, nous Définirons le type des entrées et des sorties respectivement dans les cases Input et Output, comme suit :

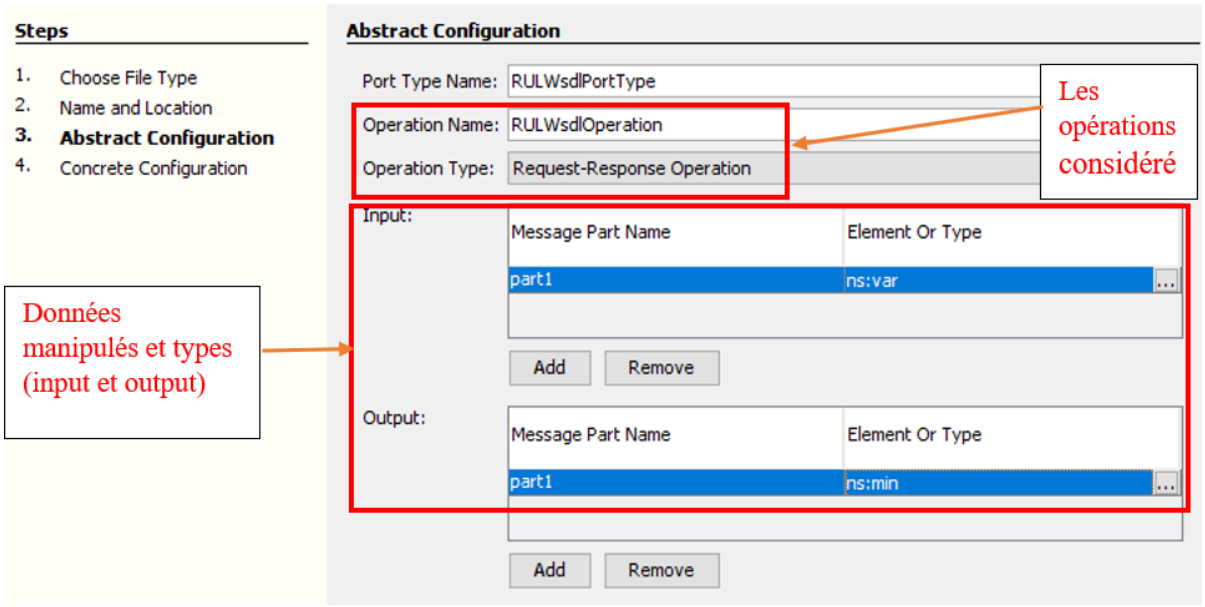


Figure 3.4 - Configuration du Fichier WSDL

3.4.4.2 Création de service web externe

Service web RUL

1. On va créer un projet web :

New → Java web → Web Application

2. On va créer ensuite un nouveau service web intitulé **RUL**.

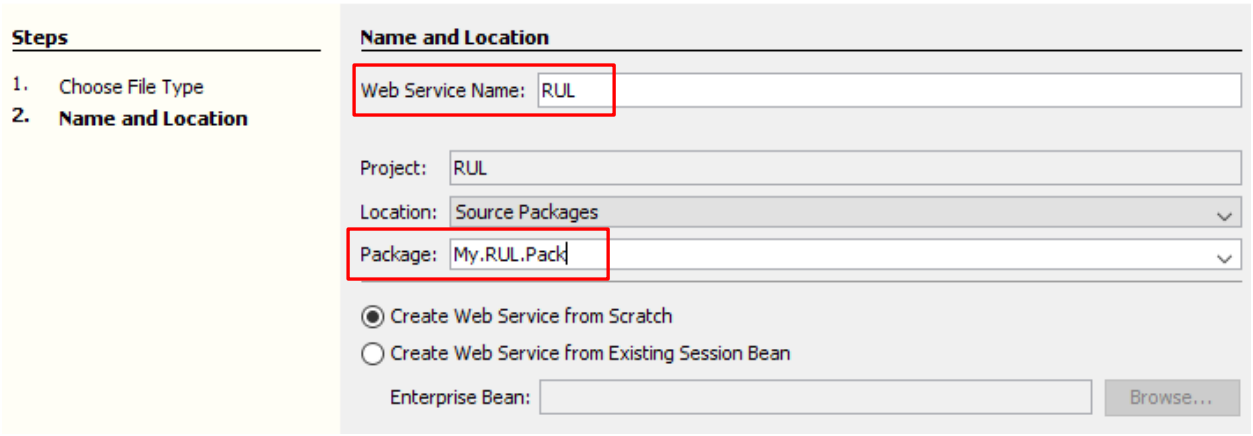


Figure 3.5 - Création de service web RUL

- Ce service contient quatre opérations, *calculRUL1*, *calculRUL2*, *calculRUL3*, *calculRUL4*. Chacun d'eux récupère quatre float, et retourne le résultat de calcule, ces opérations sont représentées par des méthodes java.

```
Source  Design
package My.RUL.Pack;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
/**
 *
 * @author Administrator
 */
@WebService()
public class RUL {

    float RUL1;

    @WebMethod(operationName = "calculRUL1")
    public float RUL1(
            @WebParam(name = "VRF") float v1,
            @WebParam(name = "DECF") float v2,
            @WebParam(name = "DSCF") float v3,
            @WebParam(name = "TECF") float v4) {
        if (v1 >= 0.70 & v1 <= 2.2) {
            if (v2 >= 1500 & v2 <= 1700) {
                if (v3 >= 1400 & v3 <= 1680) {
                    if (v4 >= 2400 & v4 <= 2900) {
                        RUL1 = 40;
                    }
                }
            }
        } else {
            RUL1 = (float) 0.00;
        }
        return RUL1;
5:21  INS
```

Figure 3.6 - Méthode java avec l'opération qui calcul *Rul1*

3. Nous enregistrons et publions ce service.

Service web MIN

- Pour créer ce service, nous suivons les mêmes étapes de création le service **RUL.**

```
@WebService()
public class MIN {

    float MIN;

    @WebMethod(operationName = "calculMIN")
    public float MIN(

            @WebParam(name = "rul1") float rul1,

            @WebParam(name = "rul2") float rul2,

            @WebParam(name = "rul3") float rul3,

            @WebParam(name = "rul4") float rul4) {
        if (rul1 < rul2) {
            MIN = rul1;
        } else {
            MIN = rul2;
        }
        if (MIN > rul3) {
            MIN = rul3;
        }
        if (MIN > rul4) {
            MIN = rul4;
        }

        return MIN;
    }
}
```

Figure 3.7 - Classe java correspond au service web (MIN)

- Donc pour finaliser notre service web nous devons enregistrer et le déployé.

3.4.5 Préparation du processus BPEL pour RUL

- Pour représenter un nouveau service web dans le processus BPEL, il faut le représenter sous forme de fichier WSDL dans le projet BPEL.
- On va faire un clic-droit sur *Process Files de* l'application ***BpelRUL***, et on va choisir New → *External WSDL Document(s)*
- Nous le donnons comme URL le chemin vers le fichier WSDL de nos services web:

  http://localhost:8080/RUL/RULService?WSDL

  http://localhost:8080/MINRUL/MINService?WSDL

- On ouvre le fichier Bpel,
- On glisse ensuite nos fichiers WSDL importé vers la partie droite de la fenêtre principale, et le fichier WSDL du service client créer déjà à gauche du processus BPEL (service qui fournit les entrées),
- Ensuite les activités trouver dans la palette à droite : Receive , Assign , Invoke, Reply et Flow entre les activités Processstart et Processend.(voir figure 3.8)

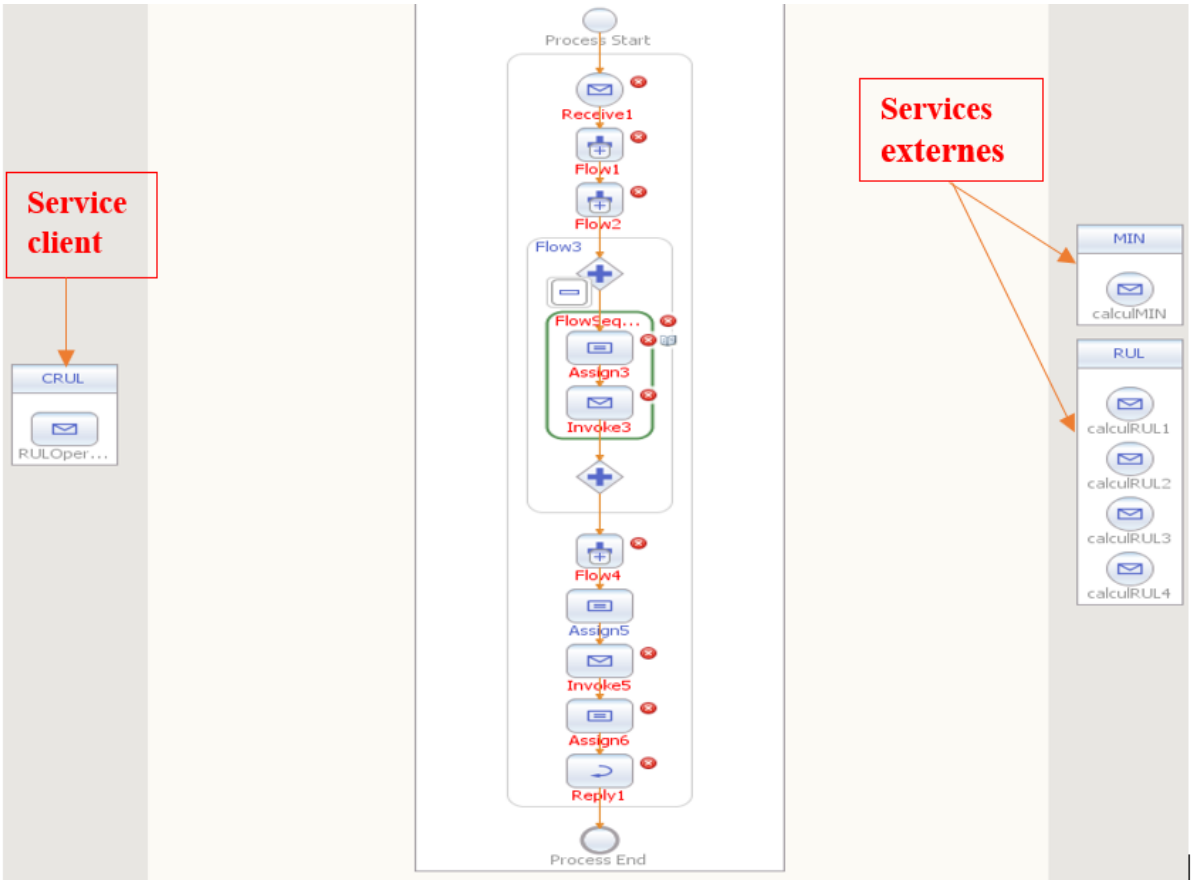


Figure 3. 8 - processus BPEL (avant la configuration)

- Nous allons maintenant faire la configuration de chaque activité :
  - **Receive** : pour récupérer les seize (16) paramètres en entrée à partir du service décrit par *RULWsdl*.

1) Nous double-cliquons sur cette activité pour la configurer,
2) On définit le Partner Link à partir duquel les données seront reçues (**CRUL**), Partener link représente le service,
3) On définit l'opération (RULWsdlOperation),
4) Ensuite, nous cliquons sur le bouton "Creat…" pour générer la variable en entrée.

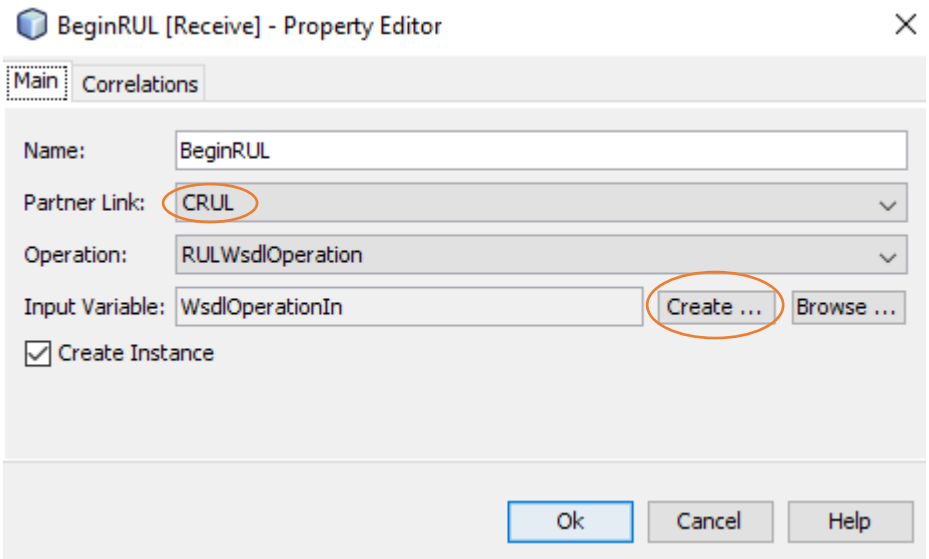


Figure 3. 9 - la configuration de l'activité **Receive**

- **Reply :** pour envoyer le résultat du processus au service client. Nous le configurons de la même manière que l'activité *Receive*.
- **Invoke :** cette activité permet de faire l'appel à un service externe.

Nous configurons l'activité *InvokeRUL1* comme indiqué ci-dessous :

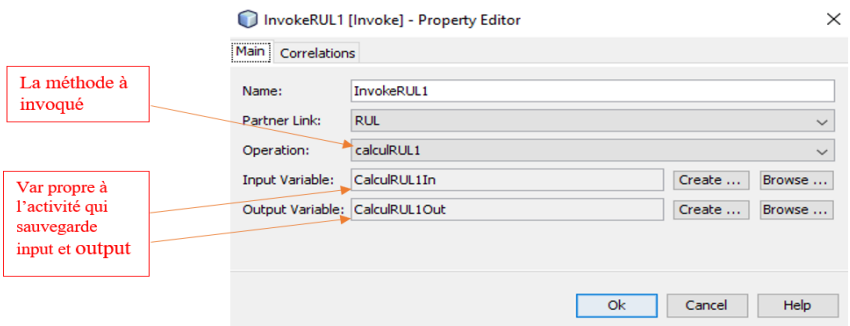


Figure 3. 10 - la configuration de l'activité **InvokeRUL1**

- Afin de configurer les activités InvokeRUL2, InvokeRUL3, InvokeRUL4, InvokeMIN, nous suivons les mêmes étapes que la configuration de l'activité InvokeRUL1.

➢ **Assign** : permet d'affecter les variables en entrée aux variables en sortie du processus.

- Nous configurons les activités Assign1 comme indiqué ci-dessous :

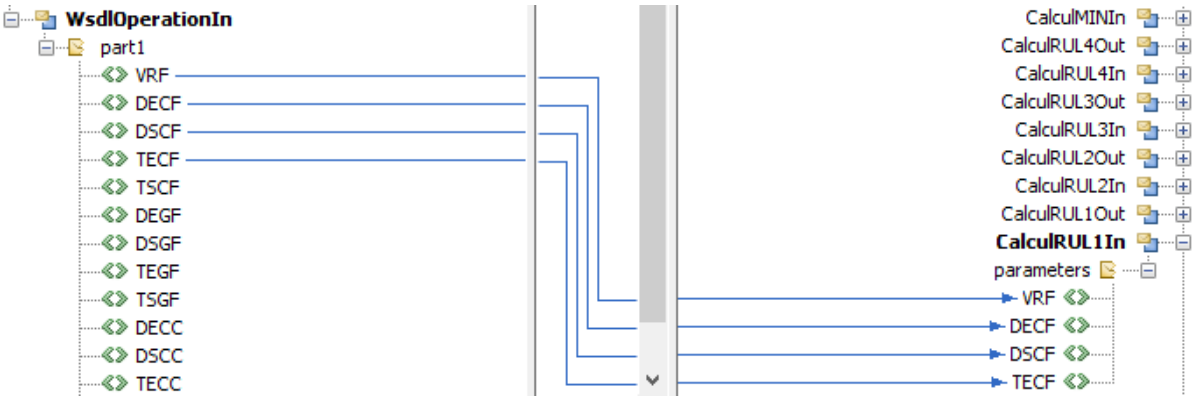


Figure 3. 11 - la configuration de l'activité **Assign1**

- Nous configurons les activités Assign2, Assign3, Assign4, Assign6 de la même manière que l'activité Assign1.

- Pour l'activité Assign5 :

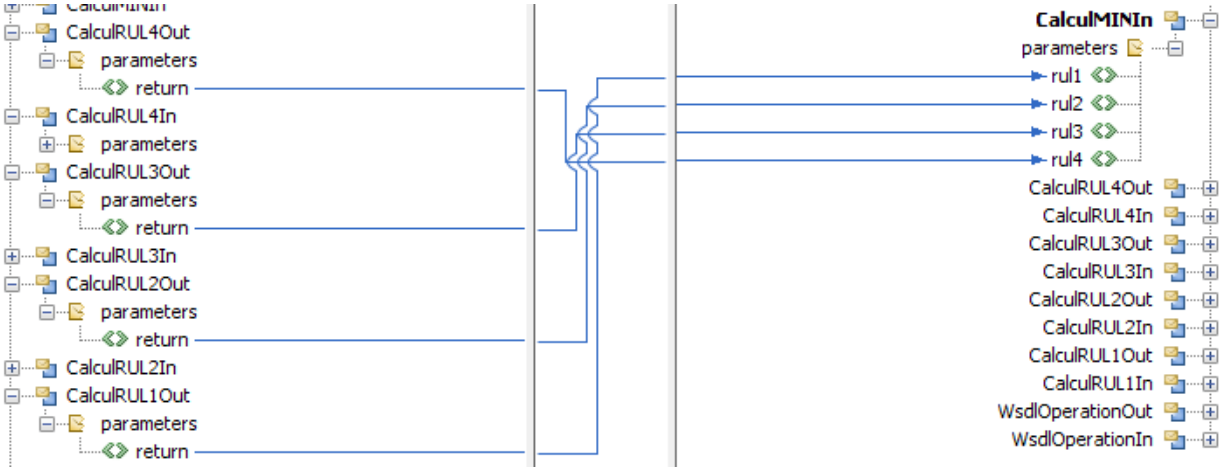


Figure 3. 12 - la configuration de l'activité **Assign5**

- Le processus résultant ressemblera à ceci :

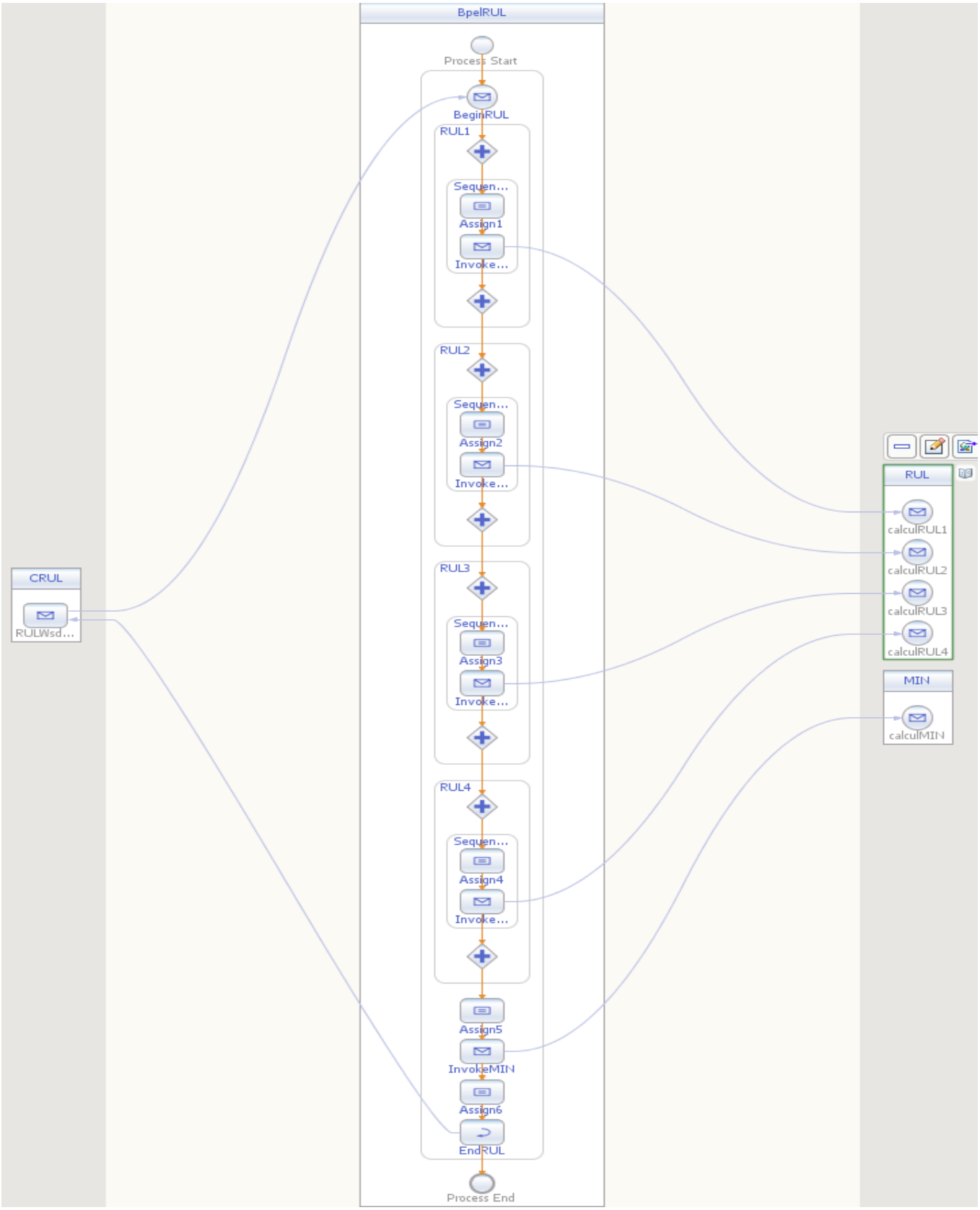


Figure 3. 13 -Processus BPEL final


```
name="BpelRUL"
targetNamespace="http://enterprise.netbeans.org/bpel/BpelRUL/BpelRUL"
xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://enterprise.netbeans.org/bpel/BpelRUL/BpelRUL" xmlns:ns0="http://xml.netbeans.org/schema/R
<import namespace="http://j2ee.netbeans.org/wsdl/RULWsdl" location="RULWsdl.wsdl" importType="http://schema
<import namespace="http://enterprise.netbeans.org/bpel/MINServiceWrapper" location="localhost_8080/MINRUL/M
<import namespace="http://Pack.RUL.My/" location="localhost_8080/MINRUL/MINService.wsdl" importType="http:/
<import namespace="http://enterprise.netbeans.org/bpel/RULServiceWrapper" location="localhost_8080/RUL/RULS
<import namespace="http://Pack.RUL.My/" location="localhost_8080/RUL/RULService.wsdl" importType="http://sc
<partnerLinks>
    <partnerLink name="RUL" xmlns:tns="http://enterprise.netbeans.org/bpel/RULServiceWrapper" partnerLinkTy
    <partnerLink name="MIN" xmlns:tns="http://enterprise.netbeans.org/bpel/MINServiceWrapper" partnerLinkTy
    <partnerLink name="CRUL" xmlns:tns="http://j2ee.netbeans.org/wsdl/RULWsdl" partnerLinkType="tns:RULWsdl
</partnerLinks>
<variables>
    <variable name="CalculMINOut" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculMINResponse"/>
    <variable name="CalculMINIn" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculMIN"/>
    <variable name="CalculRUL4Out" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculRUL4Response"/>
    <variable name="CalculRUL4In" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculRUL4"/>
    <variable name="CalculRUL3Out" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculRUL3Response"/>
    <variable name="CalculRUL3In" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculRUL3"/>
    <variable name="CalculRUL2Out" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculRUL2Response"/>
    <variable name="CalculRUL2In" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculRUL2"/>
    <variable name="CalculRUL1Out" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculRUL1Response"/>
    <variable name="CalculRUL1In" xmlns:tns="http://Pack.RUL.My/" messageType="tns:calculRUL1"/>
    <variable name="Wsdl0perationOut" xmlns:tns="http://j2ee.netbeans.org/wsdl/RULWsdl" messageType="tns:RU
    <variable name="Wsdl0perationIn" xmlns:tns="http://j2ee.netbeans.org/wsdl/RULWsdl" messageType="tns:RUL
</variables>
```

Figure 3. 14 -Processus BPEL Source

3.4.6 Création de l'Application Composite

La création d'une application composite passe par plusieurs étapes, à savoir :

1. On va choisir *New Project→SOA→Composite Application,*
2. Nous l'appelons **MyCompositeApp**,
3. Notre Application Composite est divisée en 3 parties : une pour les ports WSDL, une pour les modules JBI, et une autre pour les modules externes,
4. On va faire un clic-droit sur *JBI Modules de* l'application ***MyCompositeApp***, et on va choisir *→Add JBI Module,*
5. On va sélectionner notre projet puis clic sur Add Project JAR Files,
6. Ensuite, nous cliquons sur *build* pour voir son contenu,
7. L'étape clé de cette partie est l'ajout de Protocol SOAP, nommé **casaPort1** ensuit, nous devons le relie avec notre Bpel Project (voir figure 3.15).

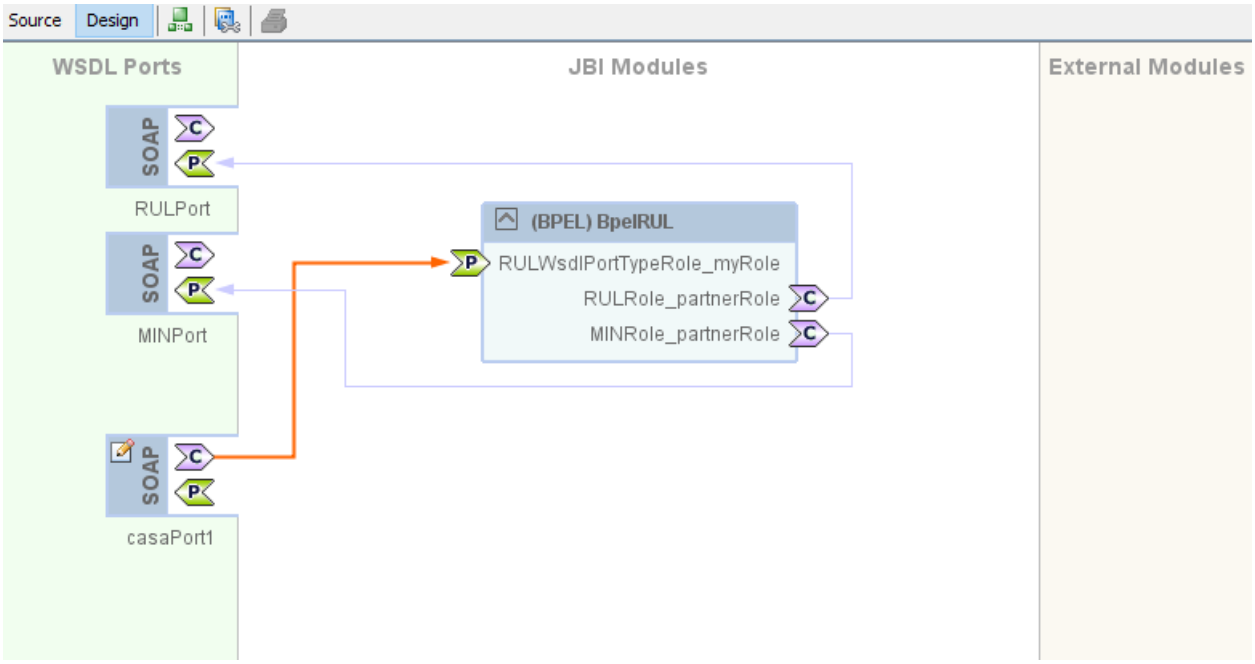


Figure 3. 15 - Composite Application finale

- Nous enregistrons et publions cette application.

3.4.6.1 Test de l'application composite

- Une fois notre application déployée, il possible de la tester.
  1. On va faire un clic droit sur le répertoire *Test de l'application composite,* et nous choisissons : *New Test Case (TestCase1),*
  2. Dans la fenêtre suivante, on va sélectionner le fichier WSDL de notre application composite, puis l'opération *RULWsdlOperation.*
  3. Nous double-cliquons sur le fichier *Input* crée, puis on modifie les paramètres en entrée notre variables.

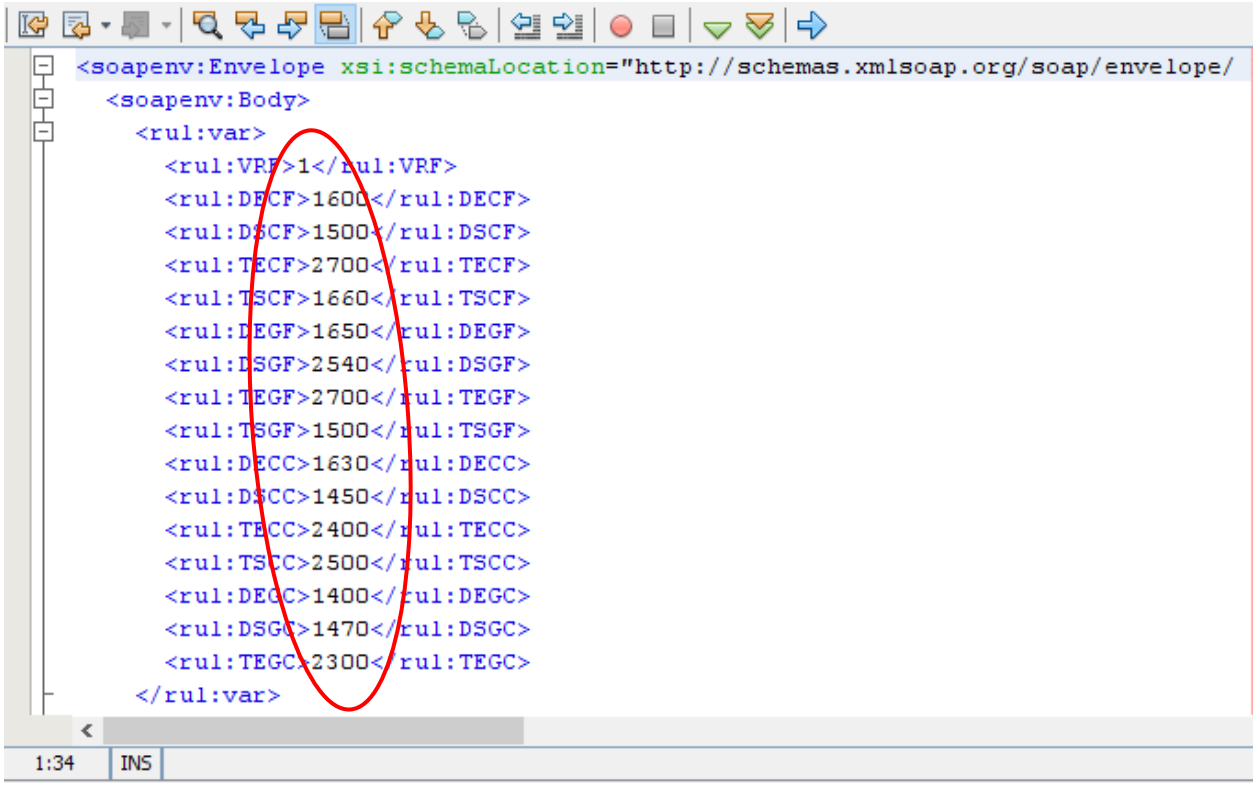


Figure 3. 16 - Fichier SOAP (Request) : Input avec les valeurs optimales

4. Clic-droit sur *TestCase1* → *RUN:*
   - ❖ Quand on fait le test pour la première fois. Généralement, une fenêtre d'erreur apparaît car le fichier output est vide

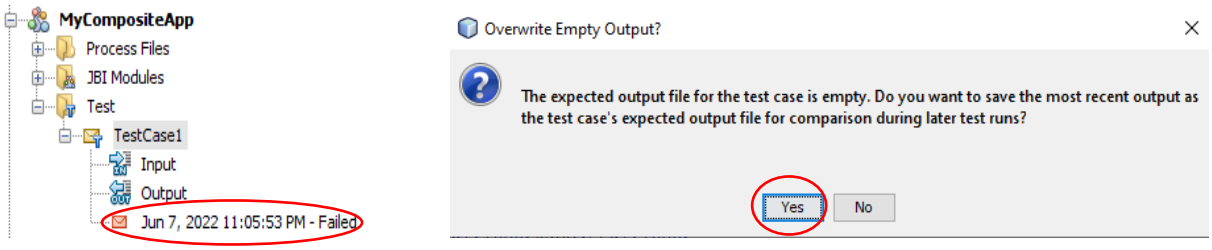


Figure 3. 17 - Test échoué

   - ❖ On clique sur Oui, ce fichier sera rempli avec le résultat obtenu et une deuxième exécution le test doit indiquer que le test a été réussi.
5. Le fichier Output va contenir le résultat de calcule.

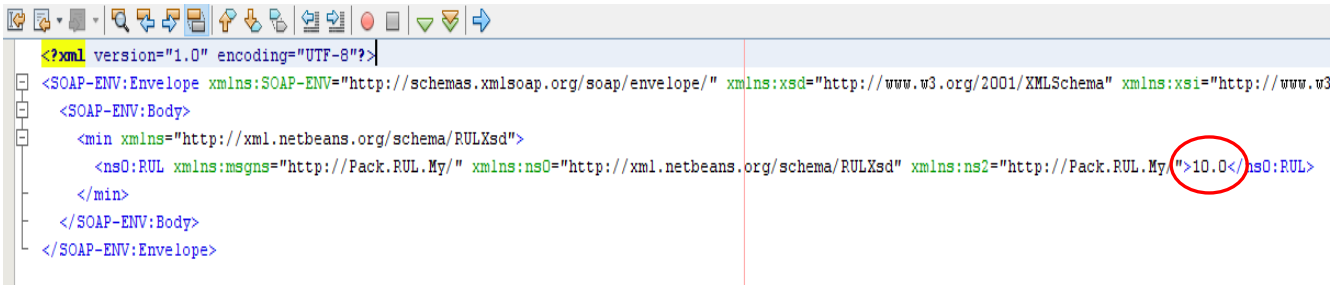


Figure 3. 18 - Fichier SOAP (Response) : Output avec le résultat de calcul

3.5 Conclusion

Dans ce chapitre nous avons vue comment nous pouvons créer une application composite web, premièrement nous avons créé nos fichiers Wsdl à partir du fichier XSD, ensuite nous avons essayé de créer les services web partenaire à savoir RUL et MIN, la troisième étape nous avons intégré ces fichiers dans notre processus métier BPEL après la configuration de chaque activité. A la fin nous avons tester notre projet par une application composite.

Conclusion générale :

Afin d'atteindre notre objectif, nous avons commencé notre travail par le pronostic et ses méthodes, et à travers cela nous avons appris que son objectif principal est d'estimer le temps de disponibilité jusqu'à la panne ou la durée de vie résiduelle (RUL) ainsi que le risque d'existence ou d'émergence ultérieure d'un ou plusieurs modèles de panne.

Autour de ce point, nous avons proposé une méthode de calcul ou d'estimation du temps restant jusqu'à la défaillance à l'aide de services Web .

Dans la deuxième partie de notre travail, nous avons étudié en détail les services Web et leurs caractéristiques, et avons également parlé de la composition des services Web, ce dernier est le pilier de la troisième partie de notre travail.

Ce travail nous permet d'estimer le temps restant avant la défaillance, en calculant le temps de défaillance de notre système (RUL Globale), en utilisant la composition des services Web, ces services interagissent avec un processus BPEL (Business Process Execution Language).

Bibliographie :

[1] AFNOR. Norme AFNOR NF EN 13306 "Terminologie de la maintenance". Ed. Afnor,

Paris, 2001.

[2] P. Wang, G. Vachtsevanos, ”Fault prognostics using dynamic wavelet neural net works”, AAAI Technical Report, 1999.

[3] A. Mathur, K.F. Cavanaugh, K.R. Pattipati, P.K. Willett, T.R. Galie, ”Reasoning

and modeling systems in diagnosis and prognosis”, Aerospace/Defense Sensing, Si mulation, and Controls, International Society for Optics and Photonics, pp. 194-203,Jul. 2001

[4] A. Muller, M.C. Suhner, B. Iung ”Formalisation of a new prognosis model for sup porting proactive maintenance implementation on industrial system”, Reliability En gineering & System Safety, vol. 93, no. 2, pp. 234-253, 2008.

[5] C.S. Byington, M.J. Roemer and T. Galie, ”Prognostic enhancements to diagnostic systems for improved condition-based maintenance [military aircraft]”, Aerospace Conference Proceedings, IEEE,vol. 6, pp. 2815-2824, 2002.

[6] C. Byington ; M. Roemer ; M. Watson ; and T. Galie. Prognostic enhancements to gas turbine diagnostic systems. Aerospace Conference, Proceedings. 2003 IEEE, 7 :3247–3255, March 2003.

[7] J. Luo ; M. Namburu ; K. Pattipati ; L. Qiao ; M. Kawamoto and S. Chigusa. Model-based prognostic techniques. In AUTOTESTCON 2003. IEEE Systems Readiness Technology Conference, 6 :330–340, California, USA, September 2003.

[8] Mickaël DIEVART. Architectures de diagnostic et de pronostic distribuées de systèmes techniques complexes de grande dimension, Thèse de doctorat, Université de Toulouse, France, Décembre 2010.

[9] Otilia Elena VASTILE. Contribution au pronostic de défaillances par réseau neuro-flou : maitrise de l’erreur de prédication, Thèse de doctorat, Université de Franche-Comte,France,2008.

[10] Asmaa MOTRANI, Rachid NOUREDDINE,METHODOLOGIE DE CHOIX D’UNE APPROCHE DE PRONOSTIC,Université Oran 2 Mohamed ben Ahmed,March 2016

[11]: BELOUAAR, Houcine. Modélisation d'une approche basée agent et logique floue pour la qualité des services Web. 2018. Thèse de doctorat. Université Mohamed Khider Biskra.

[12]: Chouhal Ouahiba. Calcul Orienté Service. 2021. Cours Master. Université Abbes Laghrour de Khenchela.

[13]: W3C World Wide Web Consortium; "Web Services Architecture"; W3C Working Group Note 11; February 2004; http://www.w3.org/TR/ws-arch

[14]: O'Sullivan Justin, Edmond David, Ter Hofstede Arthur, What's in a service Distrib. Parallel Databases ?, 12(2-3): 117–133, 2002.

[15]: IBM.BPEL4WS (version1.1), http://www.ibm.com/, 2003. Consulté le 19/05/2021.

[16] :http://deptinfo.cnam.fr/Enseignement/CycleSpecialisation/ISA/ISCS/Laloum/CORBA-cnam.pdf(derniere consultation 22 mars 2022)

[17] :https://www.researchgate.net/publication/261074660_Evaluation_of_CORBA_and_Web_Services_in_distri buted_applications (derniere consultation 22 mars 2022)

[18]: AOUICHE Ahmed. Sélection des services Web dans les systèmes distribués : étude comparative. 2019/2020. Mémoire MASTER. Université larbi ben mhidi oum elbouaghi.

[19]: MANAA said. La Logique Floue pour La Sélection de service web à base QoS. 2020/2021. Mémoire MASTER. Université Abbes Laghrour de Khenchela.

[20]: Mihi Europa. Une approche de sélection des services Web basée sur la logique floue. 2019. MEMOIRE DE MASTER. Université Mohamed Khider Biskra.

[21] : Matjaz Juric,Benny Mathew,Poornachandra Sarang,Business Process Execution Language for Web Services : BPEL and BPEL4WS, 2004