

A photograph of modern skyscrapers with curved, glass facades, viewed from a low angle looking up. The image is partially covered by a large purple diagonal shape on the right side.

WAVESTONE

Module Web Services et autres briques SOA

Version française

Octobre 2018

AGENDA

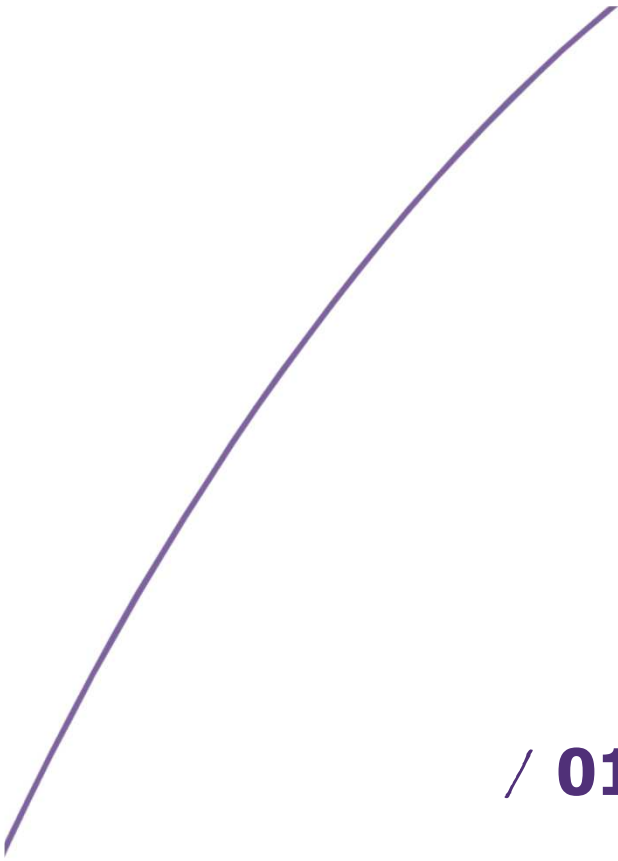
/ 01 Web Services

Page 3

/ 02 Briques SOA

Page 22

1. L'Entreprise Service Bus
2. Le Business Process Management
3. L'annuaire de Service
4. Le Business Activity Monitoring



/ **01**

Web Services

XML et son rôle dans la SOA

Rappel : Il ne doit **pas y a voir d'adhérence technologique** entre le fournisseur et le consommateur

- / **Usage d'un format d'échange pivot.**
- / **Usage d'intermédiation**

Pourquoi XML (eXtended Markup Language) ?

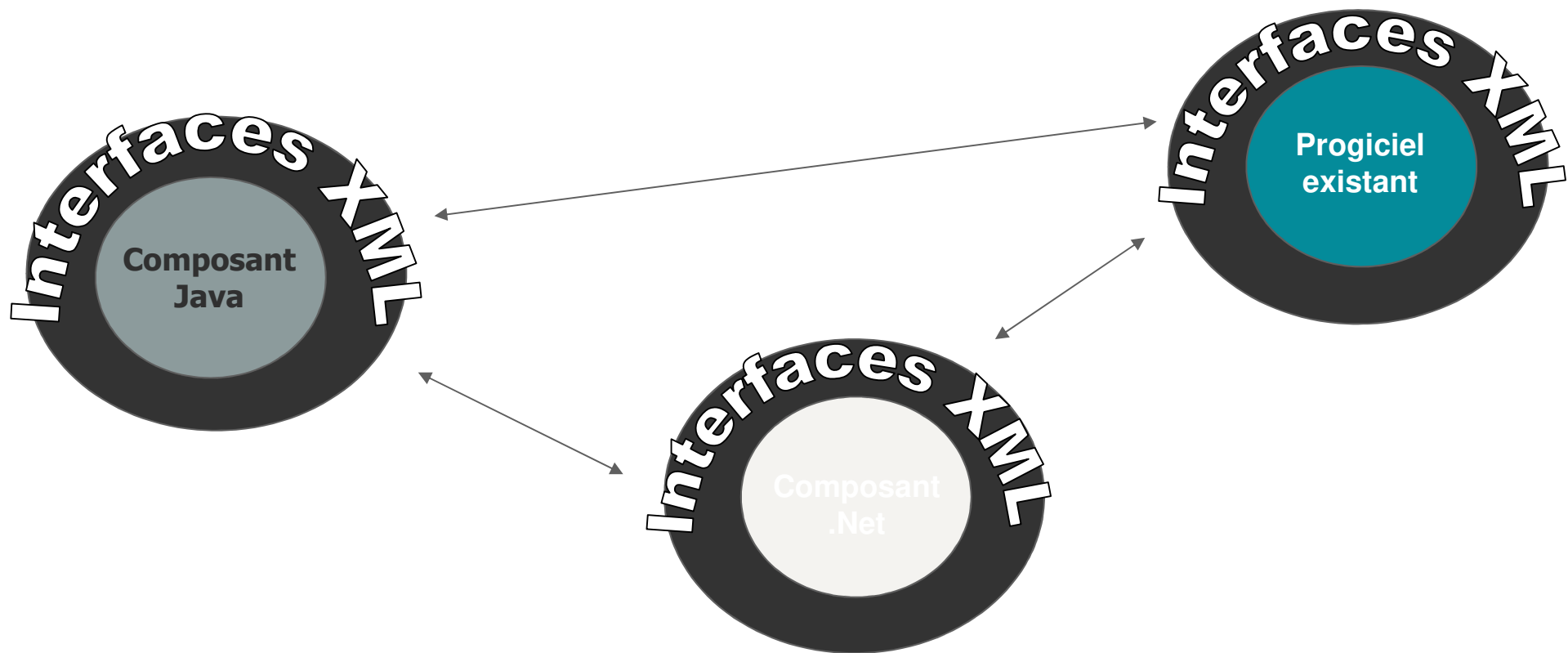
- / **Une qualité d'auto-description**
- / **Un format texte qui répond aux besoins d'échange avec l'extérieur**
- / **Capacité à intégrer des structures complexes**
- / **Validation de la conformité grâce aux schémas (XSD)**
- / **La corrélation (mapping) entre structure de donnée est facilitée (XSLT)**
- / **Un standard largement admis**

C'est un format désormais banalisé, utilisé par :

- / **Les Web Services**
- / **Le BPEL (Business Process Execution Language)**
- / **Les descripteurs de déploiement des serveurs d'applications**
- / **...**

Pourquoi XML est pertinent pour les services

La galaxie XML permet d'élaborer un jeu de protocoles qui représentent un type d'interfaces standardisé pour accéder à des services



Les services Web

Définition

Les Web Services sont une technologie permettant d'**exposer** et de **solliciter** des fonctions applicatives **distribuées** et **hétérogènes** indépendamment des plateformes et des langages par l'intermédiaire d'un échange typiquement basé sur des **documents XML**.

- La pile est bâtie sur les standards W3C et OASIS :
 - Format des messages : **X**ML (*eXtended Markup Langage*)
 - Contrat d'utilisation du service : **W**SDL (*Web Service Description Langage*)
 - Annuaire de services : **U**DDI (*Universal Description, Discovery and Integration*)
 - Enveloppe de transport : **S**OAP (*Simple Object Access Protocol*)
- Activable sur les réseaux IP via HTTP
 - Facilite l'administration de la sécurité (ex : firewall)
 - Capitalisation des infrastructures existantes

SOA	Web Services
Contrat de service	WSDL: description au format XML
Annuaire	UDDI : Interface de service normalisée/structurée
Protocole applicatif (messages échangés)	SOAP : protocole technique

Les services Web

Cas d'usage

Consommation d'un service grâce au triptyque fondateur **WSDL / SOAP / UDDI**

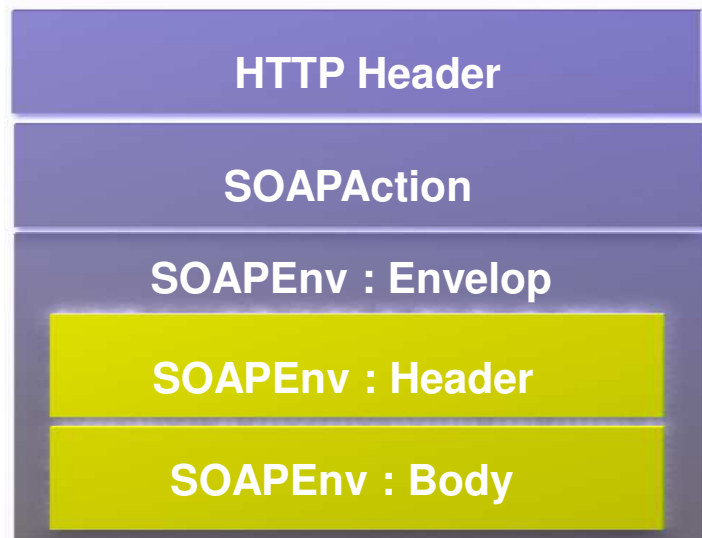


Les services Web

SOAP

▪ SOAP (Simple Object Access Protocol)

- L'enveloppe des services Web
- Organisme de tutelle : W3C



```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-
envelope">
  <env:Header>
    <m:reservation xmlns:m="/next" env:mustUnderstand="true">
      <m:reference>uuid:093a2da1-q345-739r-ba5d-pqff98fe8j7d</m:reference>
    </m:reservation>
  </env:Header>
  <env:Body>
    <p:itinerary xmlns:p="http://travelcompany.example.org/reservation/travel">
      <p:departure>
        <p:departing>New York</p:departing> <p:arriving>Los Angeles</p:arriving>
        <p:departureDate>2001-12-14</p:departureDate> <p:departureTime>late
        afternoon</p:departureTime>
        <p:seatPreference>aisle</p:seatPreference>
      </p:departure>
      <p:return>
        <p:departing>Los Angeles</p:departing> <p:arriving>New
        York</p:arriving>
        <p:departureDate>2001-12-20</p:departureDate>
        <p:departureTime>mid-morning</p:departureTime>
        <p:seatPreference/>
      </p:return>
    </p:itinerary>
  </env:Body>
</env:Envelope>
```


Les services Web

WSDL

- **WSDL** (Web Services Description Language)
 - Organisme de tutelle : W3C
 - Il fournit la description des messages SOAP échangés

```
<?xml version="1.0"?>
<definitions name="StockQuote"
  targetNamespace="http://example.com/stockquote.wsdl"
  xmlns:tns="http://example.com/stockquote.wsdl"
  xmlns:xsd1="http://example.com/stockquote.xsd"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <schema targetNamespace="http://example.com/stockquote.xsd"
      xmlns="http://www.w3.org/2000/10/XMLSchema">
      <element name="TradePrice">
        <complexType>
          <all> <element name="price" type="float"/> </all>
        </complexType>
      </element>
    </schema>
  </types>
  <message name="GetLastTradePriceInput">
    <part name="body" element="xsd1:TradePriceRequest"/>
  </message>
  <message name="GetLastTradePriceOutput">
    <part name="body" element="xsd1:TradePrice"/>
  </message>
```

```
<portType name="StockQuotePortType">
  <operation name="GetLastTradePrice">
    <input message="tns:GetLastTradePriceInput"/>
    <output message="tns:GetLastTradePriceOutput"/>
  </operation>
</portType>
<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetLastTradePrice">
    <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>
<service name="StockQuoteService">
  <documentation>My first service</documentation>
  <port name="StockQuotePort" binding="tns:StockQuoteBinding">
    <soap:address location="http://example.com/stockquote"/>
  </port>
</service>
</definitions>
```

Les services Web

L'interopérabilité

L'interopérabilité n'était historiquement pas garantie par les Web Services (implémentations variantes des standards)

Le consortium **WS-I** (*Web Services Interoperability*) a édicté un ensemble de recommandations afin de favoriser l'interopérabilité

Les règles importantes concernent les domaines suivants

- / Exclusion du SOAP encoding
- / Utilisation du mode RPC-literal ou Document-literal (WSDL)
- / Utilisation de SOAP HTTP
- / Utilisation de HTTP 500 pour la gestion des messages d'erreur SOAP Fault
- / Utilisation du HTTP Post
- / Utilisation du WSDL 1.1 pour décrire le Web service

Les implémentations comme JAX-WS et .NET peuvent être paramétrées afin de respecter le **Basic Profile** WS-I

Les services Web

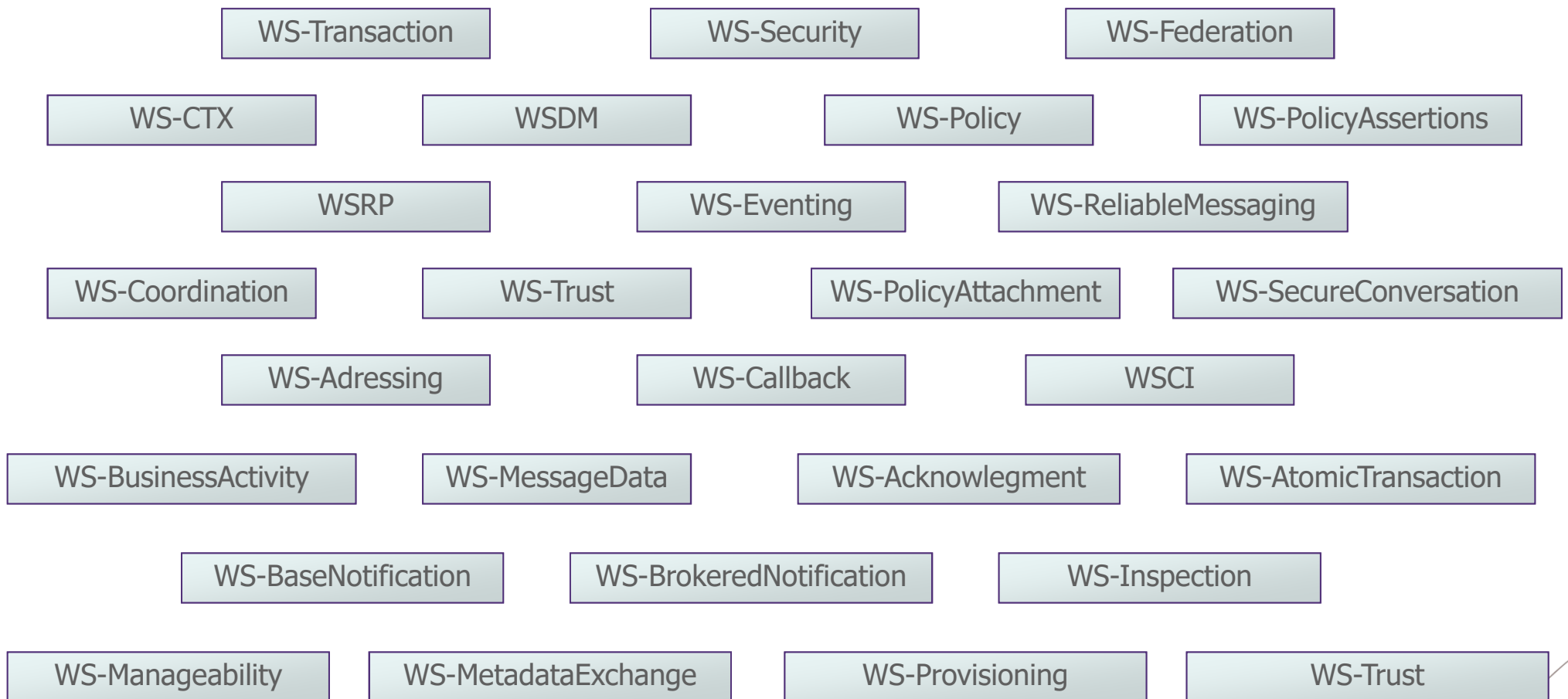
Les standards

- Si SOAP, WSDL et UDDI représentent une première étape dans la standardisation des services Web, de nombreuses questions restent en suspens :
 - Comment implémenter le **SLA** (contrat de service) ?
 - Comment mettre en œuvre des **transactions** à travers un tel procédé de couplage lâche ?
 - Comment garantir la **sécurité** de ces services ?
 - Comment garantir la **fiabilité** de ces services ? (Haute disponibilité)
 - Comment garantir les **performances** ?
 - Comment **superviser** les services web ?
 - Comment **orchestrer/composer** des processus à base de services Web ?
 - Comment **agréger** ces services à travers un portail d'entreprise ?

Les services Web

Les standards (WS-*)

- Un ensemble de standards gravitent autour du triptyque **WSDL / SOAP / UDDI** :



Les services Web

Les standards (WS-*)

Parmi cette multitude de standards, seuls certains sont reconnus et supportés par la majorité des implémentations de Web Services

On peut retenir les standards suivants :

Annuaire, UDDI, WSIL (Web Services Inspection Language)

Méta-données

WS-MetadataExchange

Processus, BPEL

Contrat, WSDL, XSD,

WS-Policy,

WS-Agreement

Sécurité,

WS-Security

Fiabilité,

WS-Transaction,
WS-ReliableMessaging

Message, SOAP, WS-Addressing, WS-Eventing

Réseaux, http, smtp, ftp, iiop...

Les services Web

WS-Agreement

WS-Agreement

- / Définition des contrats de services mettant en relation les consommateurs et les fournisseurs de services
- / Cela offre des garanties sur des ressources comme :
 - Les temps de réponse.
 - La capacité d'enregistrement.
 - La bande passante.
 - La mémoire.

```
<wsag:Variable name="CPUcount" metric="job:numberOfCPUs">  
  <wsag: Location>  
    //wsag:AgreementOffer/wsag:Terms/wsagAll/wsag:ServiceDefinitionTerm/job:executable  
  </wsag: Location/>  
</wsag:Variable/>
```

Les services Web

WS-Transaction

WS-Transaction

- / Définit les mécanismes d'interopérabilité des transactions entre Web Services
- / Ce standard est composé de :
 - WS-Coordination
 - WS-AtomicTransaction
 - WS-BusinessActivity

Les services Web

La sécurité (1/3)

L'utilisation des Web Services a des conséquences sur la sécurité :

- / Exposition à l'extérieur du SI
- / Transit par plusieurs nœuds intermédiaires entre l'émetteur et le destinataire du message
- / Potentialité d'une facturation

Les Web services posent donc les contraintes de sécurité suivantes :

- / Contrôle des accès aux services exposés
- / Confidentialité des messages
- / Intégrité des messages (pas de modification lors du cheminement du message)
- / Non répudiation des messages par les clients

La sécurisation des Web Services peut se faire sur deux niveaux :

- / HTTP & HTTPS : sécurisation au niveau du protocole de transport
- / **WS-Security** : sécurisation au niveau des messages SOAP

Les services Web

La sécurité (2/3)

HTTP :

- / Fournissent juste un mécanisme simple d'authentification par login / mot de passe
- / Insuffisants pour répondre aux besoins de sécurisation des Web Services

HTTPS :

- / Évolution d'HTTP basée sur SSL (Secure Sockets Layer)
- / Permet :
 - / L'authentification
 - / La certification du destinataire et de l'émetteur
 - / Le chiffrement au niveau transport
- / Fonctionnellement limité
- / Point à point : inadapté à un traitement de messages multi-acteurs

WS-Security :

- / Mise en place des informations de sécurité de base dans les headers SOAP
- / Supporte de multiples méthodes de signature et cryptage

```
<wsp:Policy>
  <wsp:ExactlyOne>
    <wsp:All>
      <wsse:SecurityToken>
        <wsse:TokenType>wsse:Kerberosv5TGT</wsse:TokenType>
      </wsse:SecurityToken>
    </wsp:All>
    <wsp:All>
      <wsse:SecurityToken>
        <wsse:TokenType>wsse:X509v3</wsse:TokenType>
      </wsse:SecurityToken>
    </wsp:All>
  </wsp:ExactlyOne>
</wsp:Policy>
```

Les services Web

La sécurité (3/3)

Authentification :

- / Basée sur l'insertion d'un token de sécurité dans les entêtes SOAP
- / WS-Security décrit la représentation XML :
 - D'une authentification classique par couple login / mot de passe,
 - Des certificats X509,
 - Des tickets Kerberos,
 - Des tokens binaires,
- / Les mécanismes d'authentification définis par WS-Security sont extensibles

Chiffrement :

- / Basé sur **XML-Encryption**
- / Définition des paramètres de cryptages dans les entêtes SOAP (algorithme, clef de chiffrement, ...)
- / Représentation des données chiffrées par des balises spécifiques dans le corps des messages SOAP

Signature :

- / Basée sur **XML-Signature**
- / Signature partielle d'un message possible
- / Définit la signature et ses paramètres de calcul dans les entêtes SOAP
- / Pointe vers le bloc du message à signer

Les services Web


WS-Addressing

WS-Addressing

- / Définition des mécanismes d'adressage indépendant de la couche de transport
- / Permet l'implémentation de points de routage intermédiaires entre source et destinataire

```
POST /Dev/WebServices/BasicServices.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/GetAllDocumentsByFilter"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
.....
</soap:Envelope>
```



```
<wsa:MessageID> xs:anyURI </wsa:MessageID>
<wsa:To>xs:anyURI</wsa:To>
<wsa:Action> http://tempuri.org/GetAllDocumentsByFilter </wsa:Action>
<wsa:From>endpoint-reference</wsa:From>
<wsa:ReplyTo>endpoint-reference</wsa:ReplyTo>
<wsa:FaultTo>endpoint-reference</wsa:FaultTo>
```

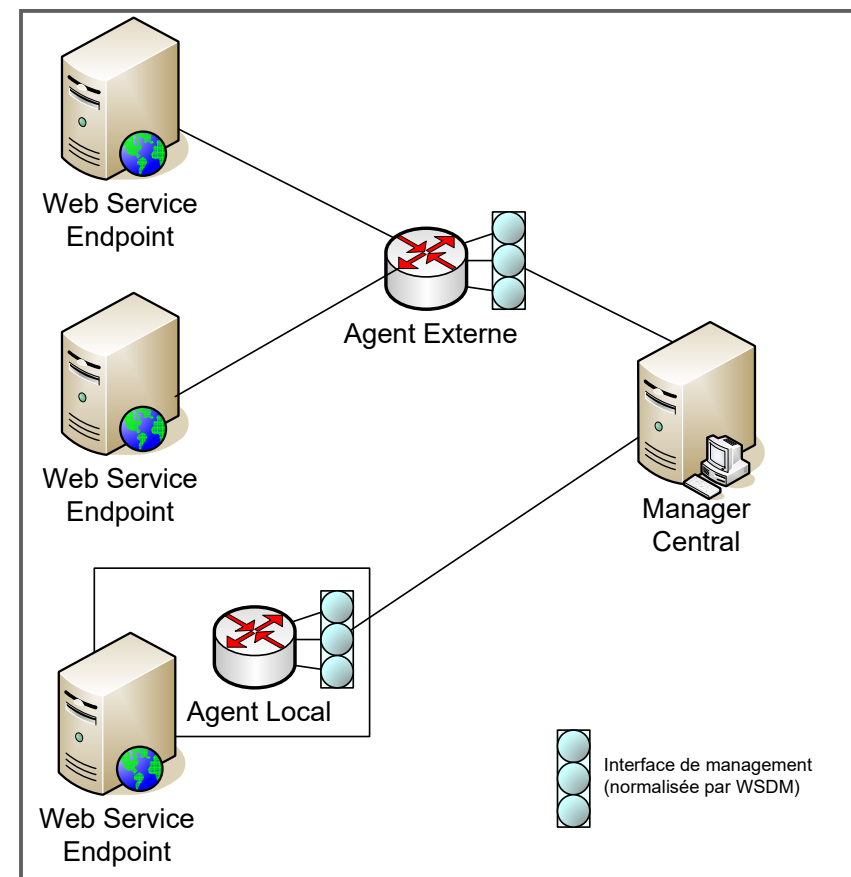
Les services Web

WS-DistributedManagement

WS-Distributed Management (WSDM)

/ Définition de l'architecture et des opérations pour le management des endpoints : identification, state, dépendance entre les ressources managées (relationship)...

- Exposition d'une interface de management par chaque endpoint
- Chaque endpoint expose une interface de services qui contient les opérations d'administration disponibles :
 - Identification du service,
 - État (state),
 - Configuration,
 - Métriques (Metrics),
 - Relation (Relationship).





/ **02**

Briques SOA



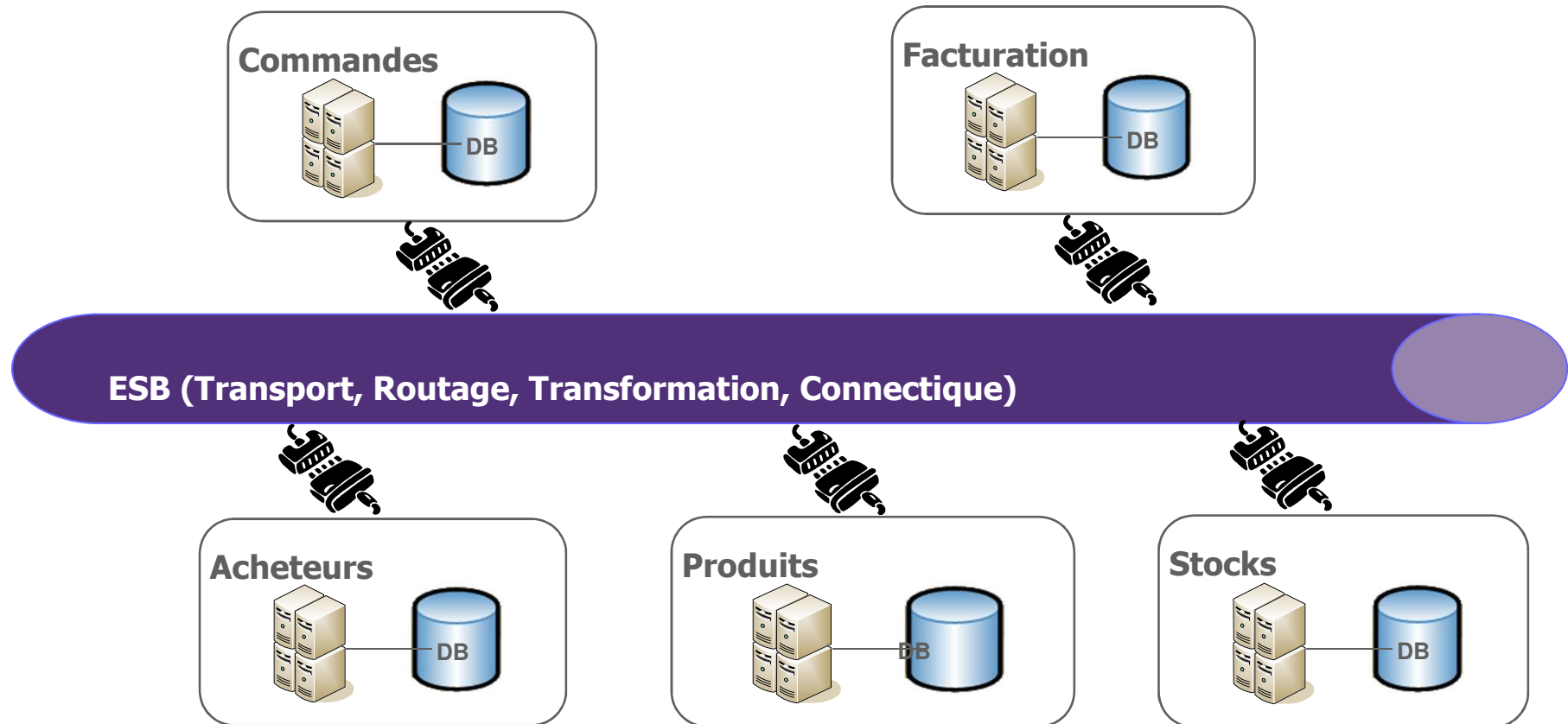
/ **02.1**

L'Entreprise Service Bus



L'ESB : Enterprise Service Bus

L'ESB : Une solution constituée d'un bus d'intégration reposant sur **les standards Web et J2EE**



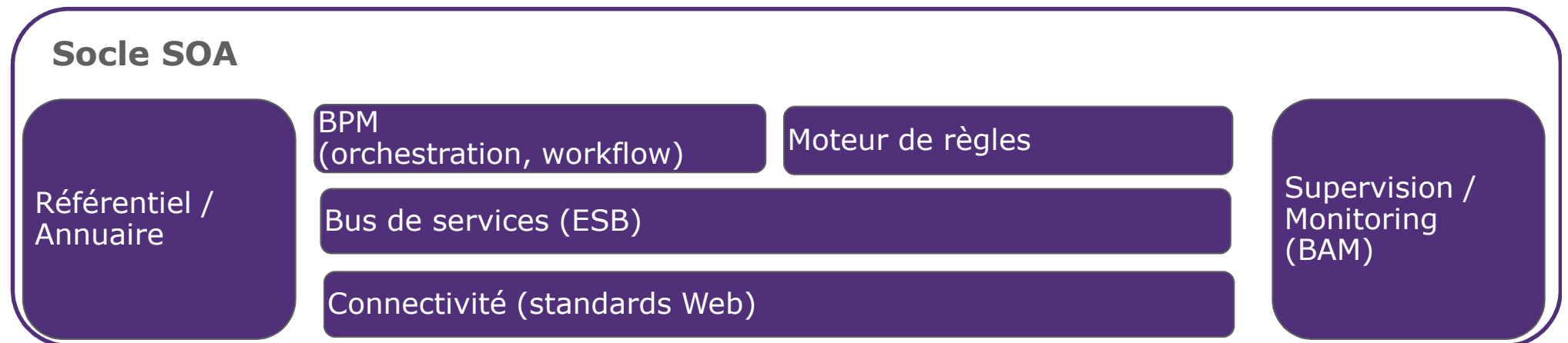
Positionnement et couverture de l'ESB dans la SOA

L'ESB a la responsabilité des échanges

- / Transport
- / Routage
- / Connecteurs (Sous la forme d'appel de services)

Autres rôles de l'ESB

- / Orchestrateurs de services techniques
- / Moteur de règles techniques



Concept d'orchestration

Les appels de services sont orchestrés dynamiquement par un moteur qui invoque les services

L'orchestration est définie comme étant un ensemble d'activités privées ou étapes d'un processus ou Workflow dont le franchissement est contrôlé par le dépositaire. Elles ont pour rôle :

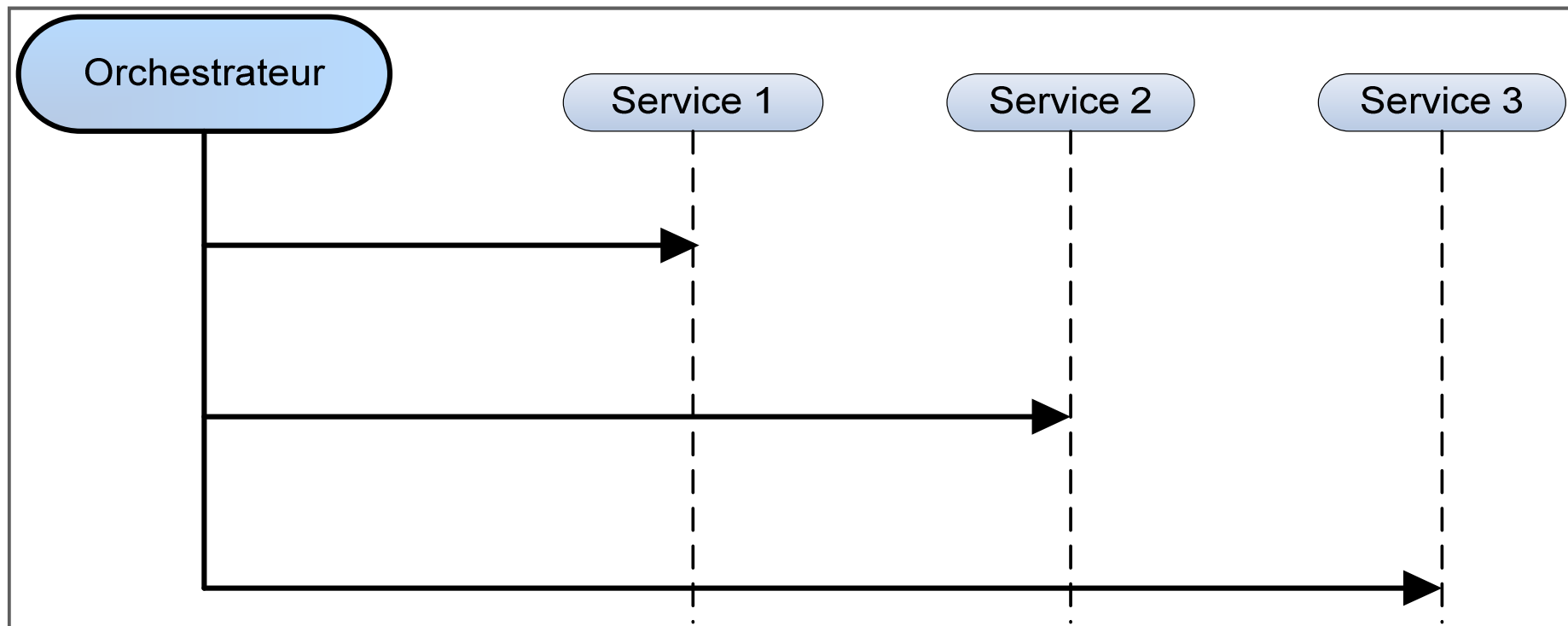
- / De représenter la logique applicative. La fonction d'orchestration comprend toute la logique métier et fonctionnelle (Enchaînement de services).
- / De conserver le contexte, et les données entre plusieurs appels (maintien de l'information entre les services).
- / De gérer les transactions (2pc, stratégie de Compensation)
- / Éventuellement de maîtriser la QoS et le SLA

Concept d'orchestration - Gestion de la logique applicative

Contient la logique fonctionnelle d'assemblage des éléments

Les règles métiers restent localisées dans les éléments assemblés

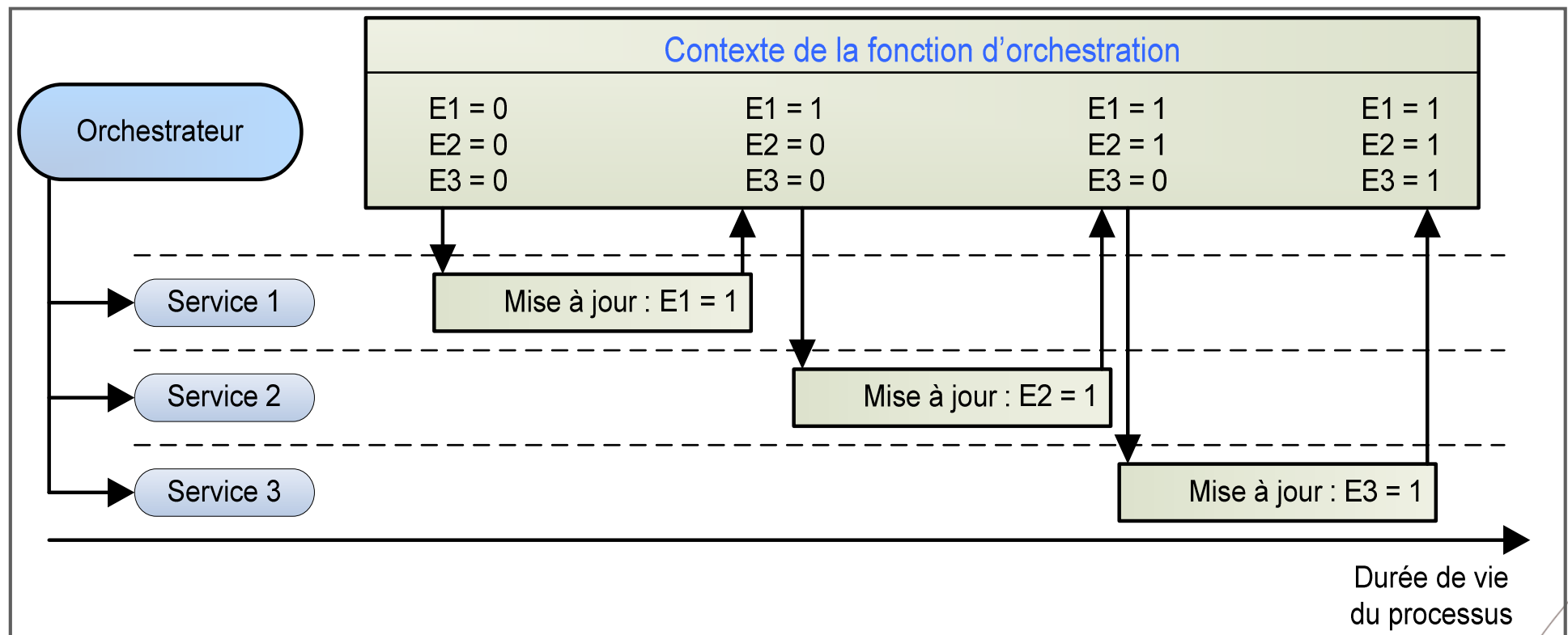
Partie « workflow » de la fonction d'orchestration



Concept d'orchestration - Gestion du contexte

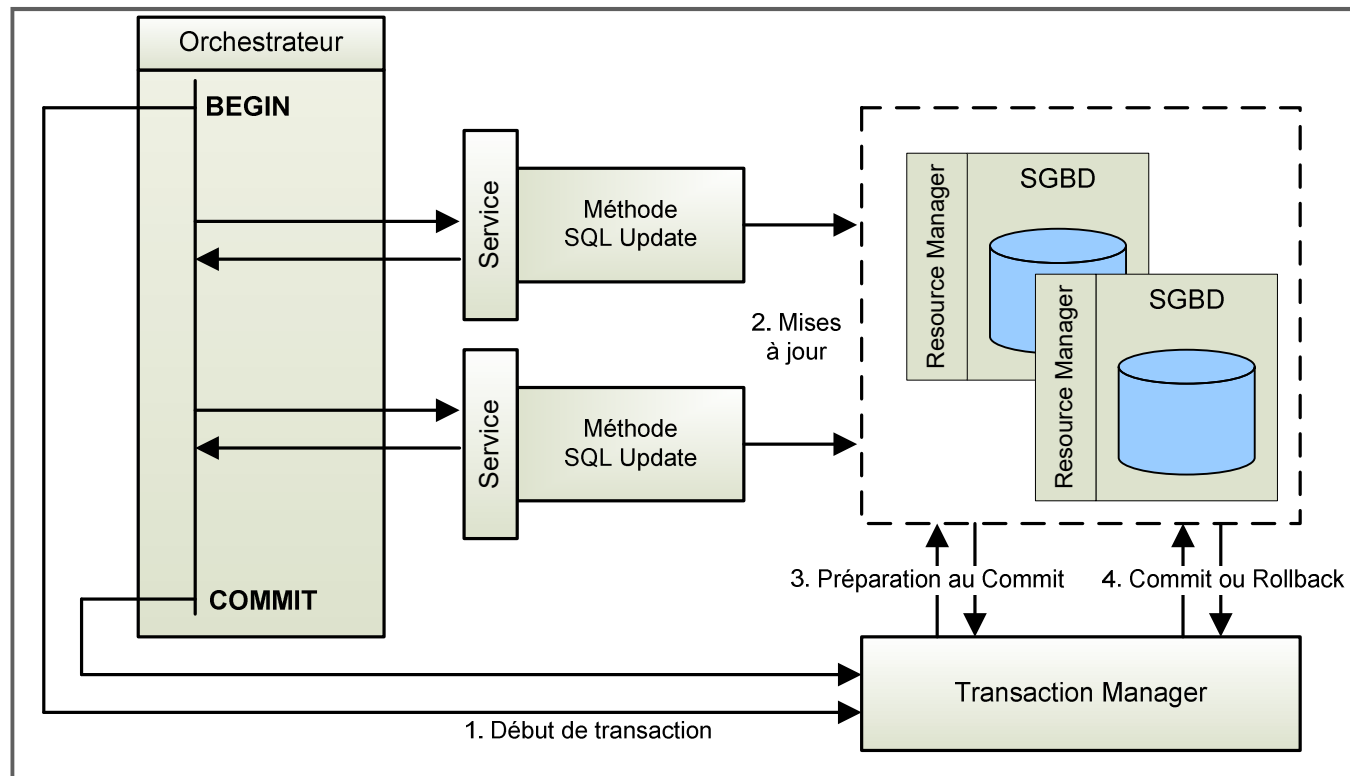
Prend en charge la conservation de certaines données qui sont nécessaires tout au long de la durée de vie d'un enchaînement

Un bon orchestrateur doit avoir la capacité de persister son état



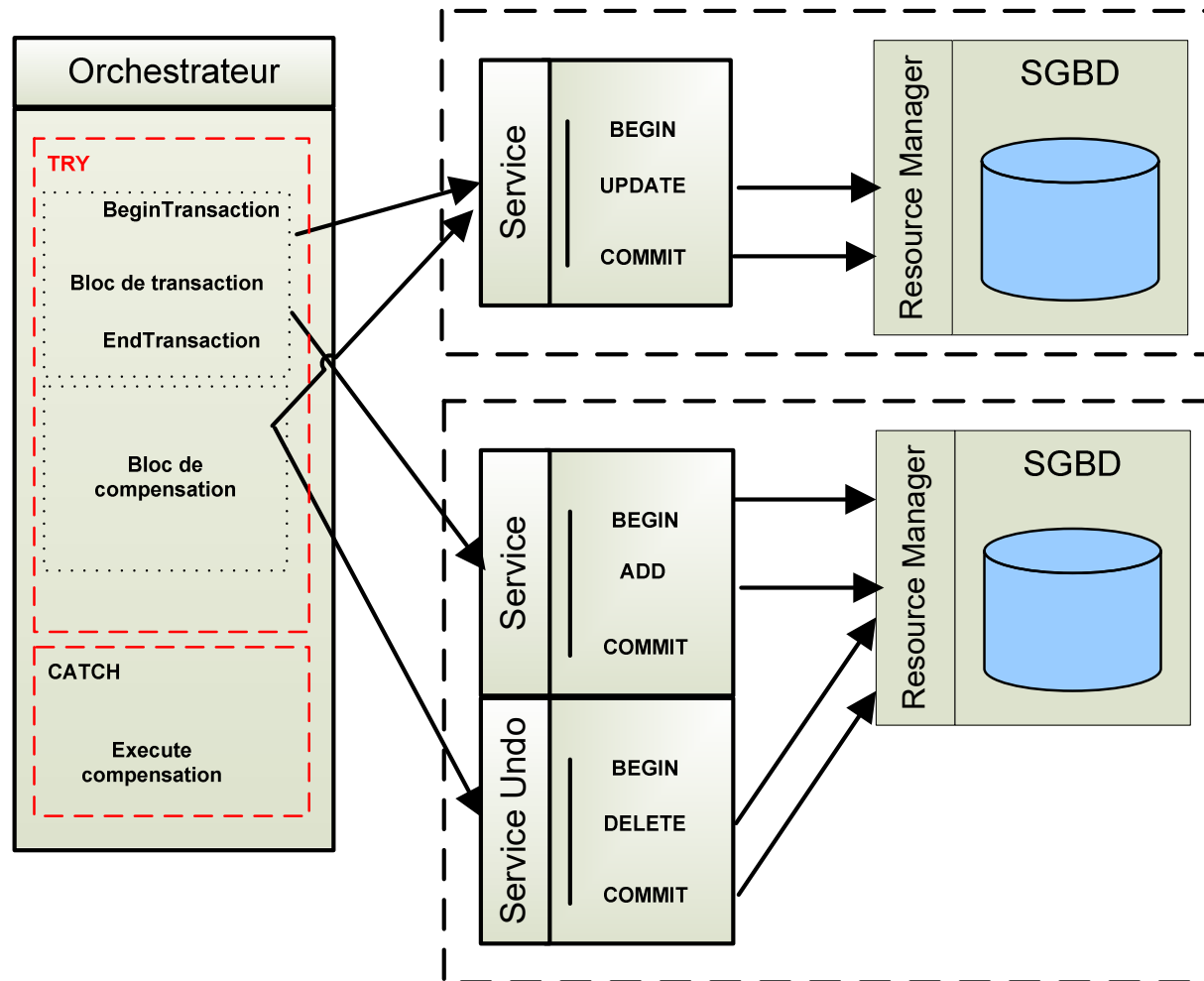
Concept d'orchestration - Gestion transactionnelle (1/2)

- Gère les opérations de Commit/Rollback dans le cas d'une transaction technique
- Gère le processus de compensation lorsqu'un rollback technique ne peut être appliqué



Gestion transactionnelle (via Rollback)

Concept d'orchestration - Gestion transactionnelle (2/2)



Gestion transactionnelle avec compensation

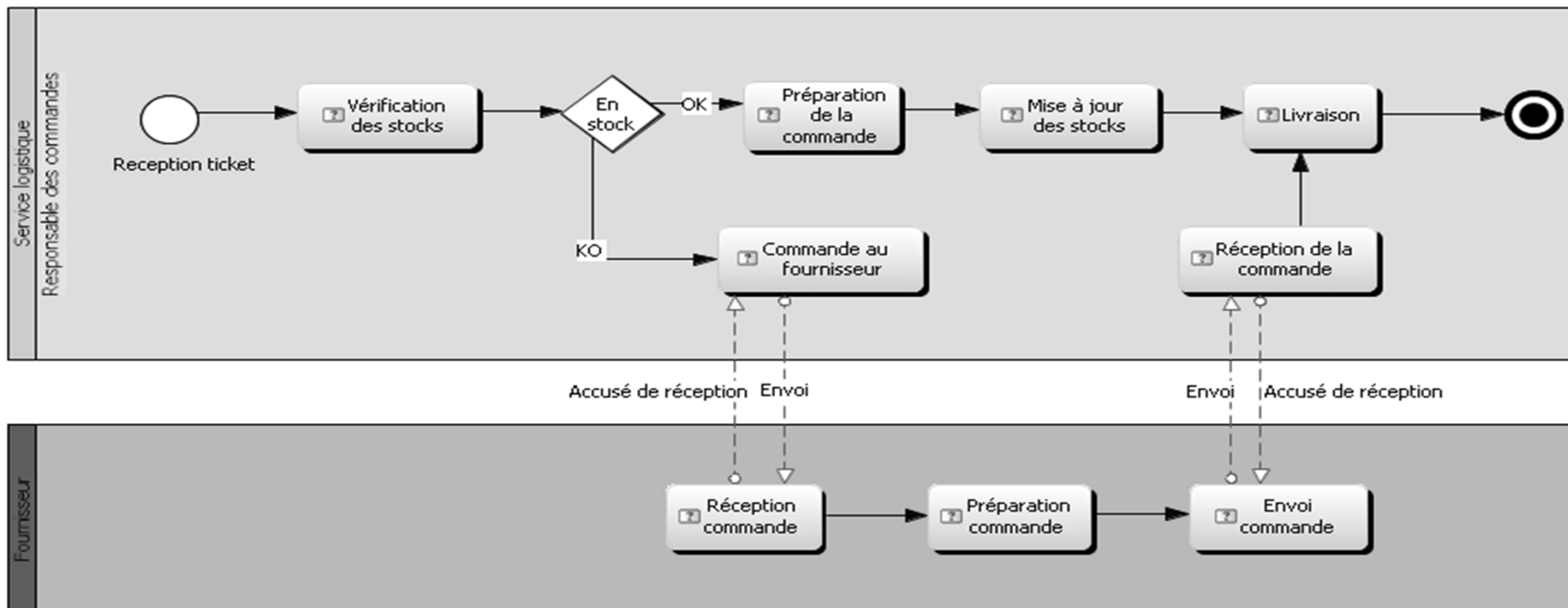
A decorative purple curve starts from the left edge of the slide, curves upwards and to the right, and ends near the top edge.

/ **02.2**

Le Business Process Management

Le BPM : Définition

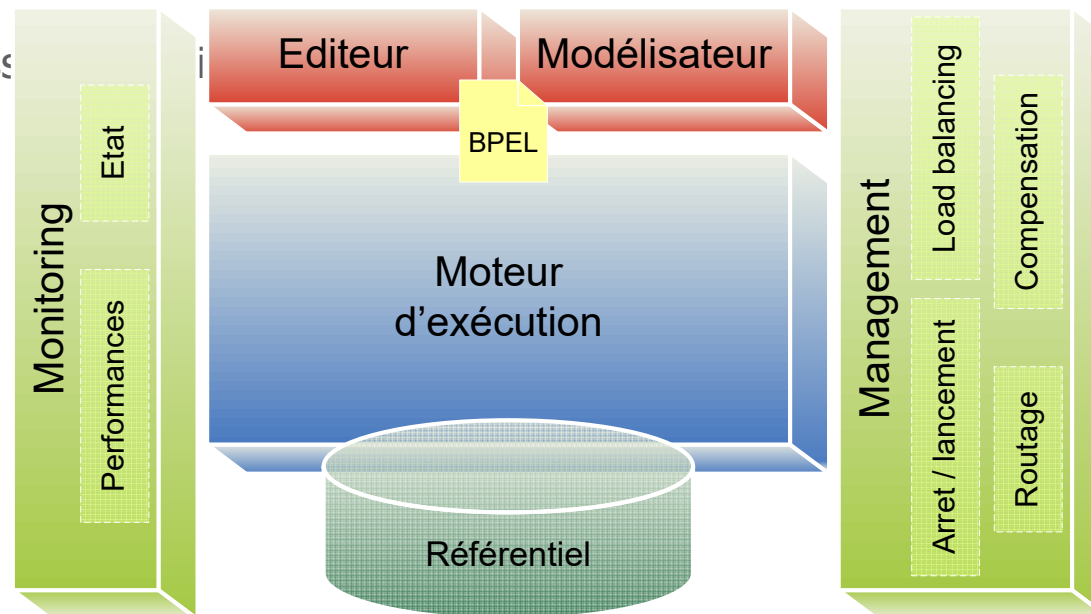
- **BPM** : système permettant de gérer des processus métier de bout en bout
- **Processus métier** : traitement de haut niveau organisationnel reflétant l'activité métier de l'entreprise ou du domaine d'activité étudié



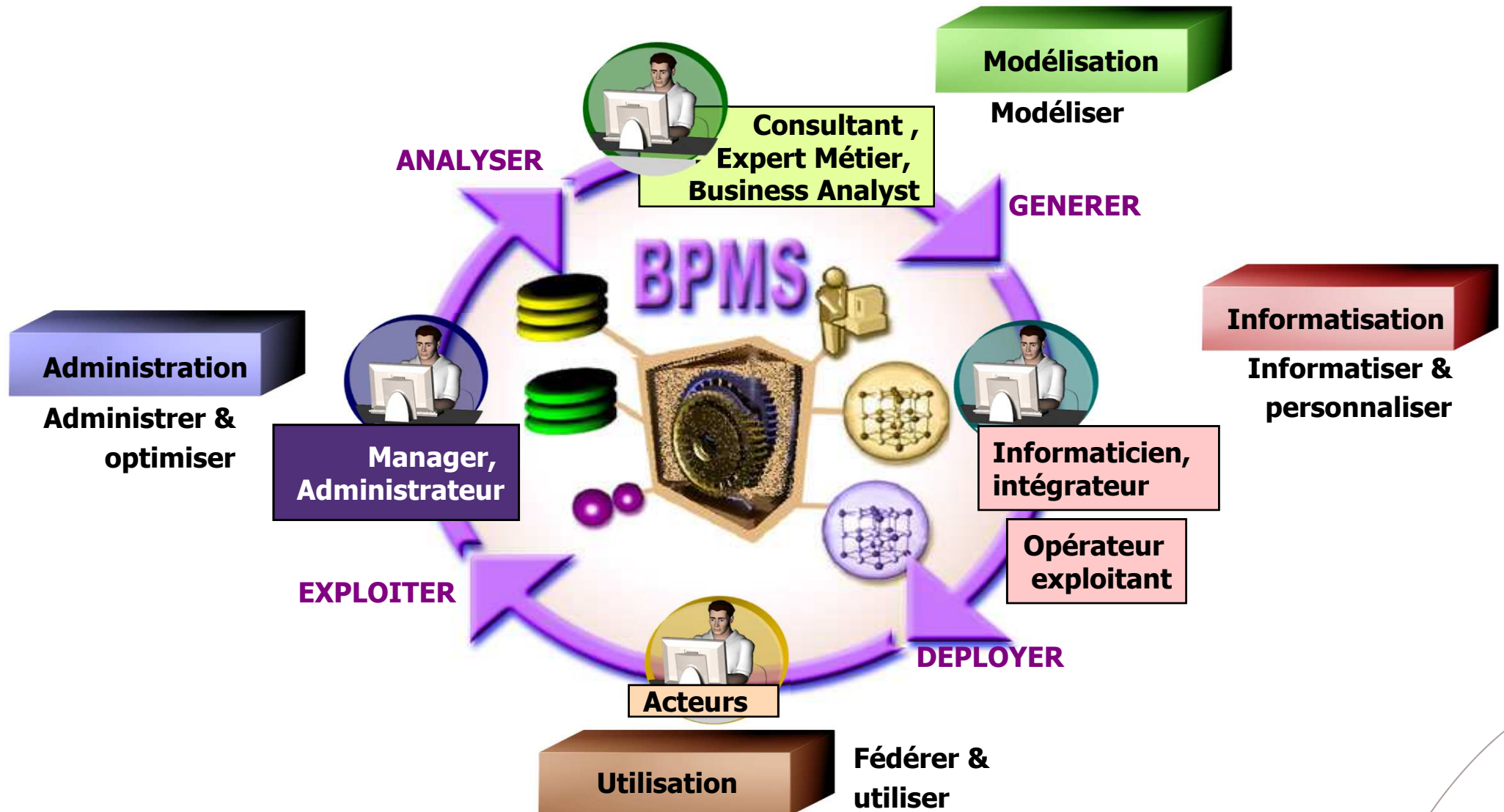
Composants d'une offre BPM

Dans une vision large le BPM se décompose de la manière suivante :

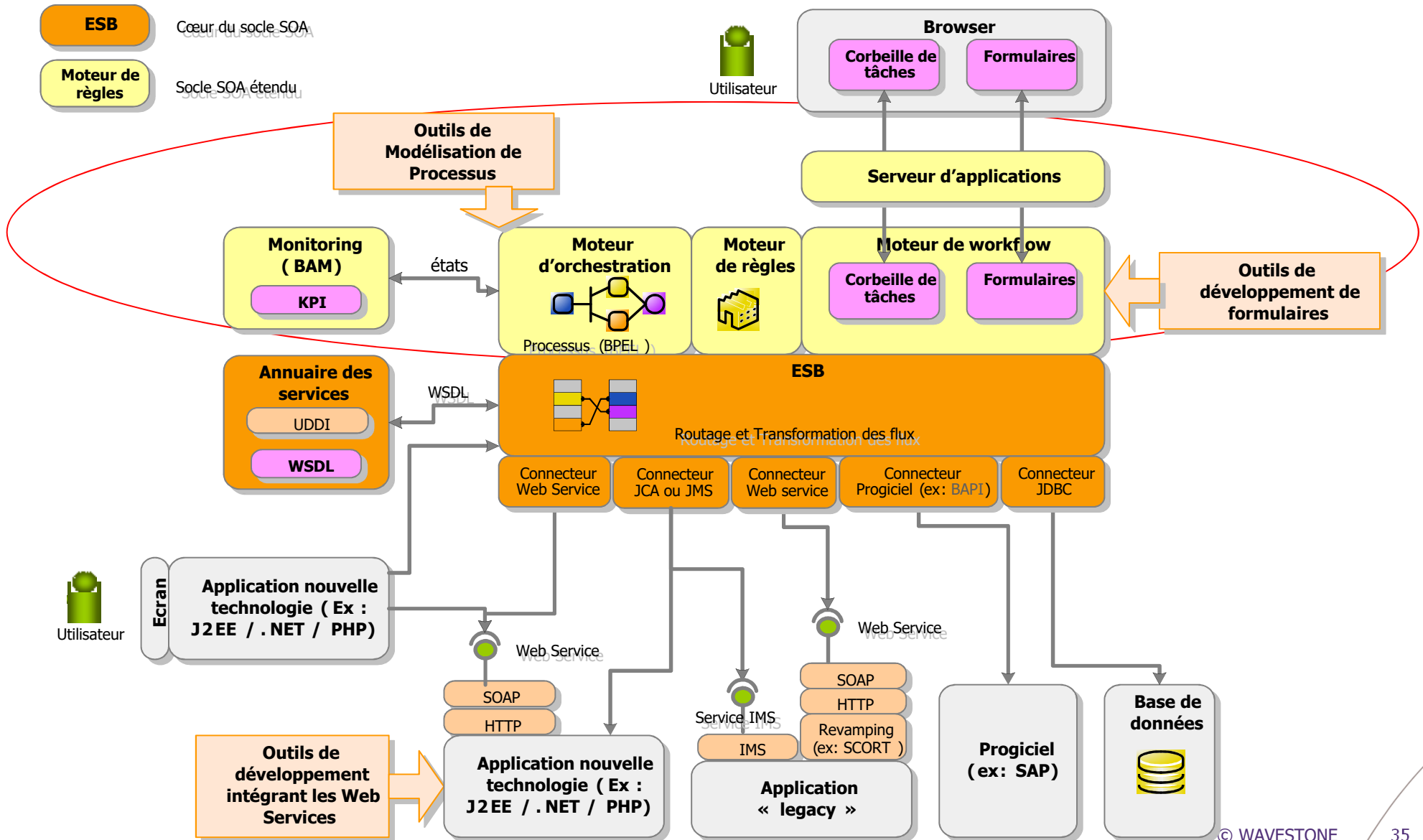
- / La modélisation des processus métier
- / L'automatisation des processus, c'est-à-dire l'intégration, l'orchestration, le Workflow...
- / La performance : suivi des coûts (**A**ctivity **B**ased **C**osting), simulation (Process Simulation) et contrôle continu de la performance des processus (**B**usiness **A**ctivity **M**onitoring)
- / Etat des processus



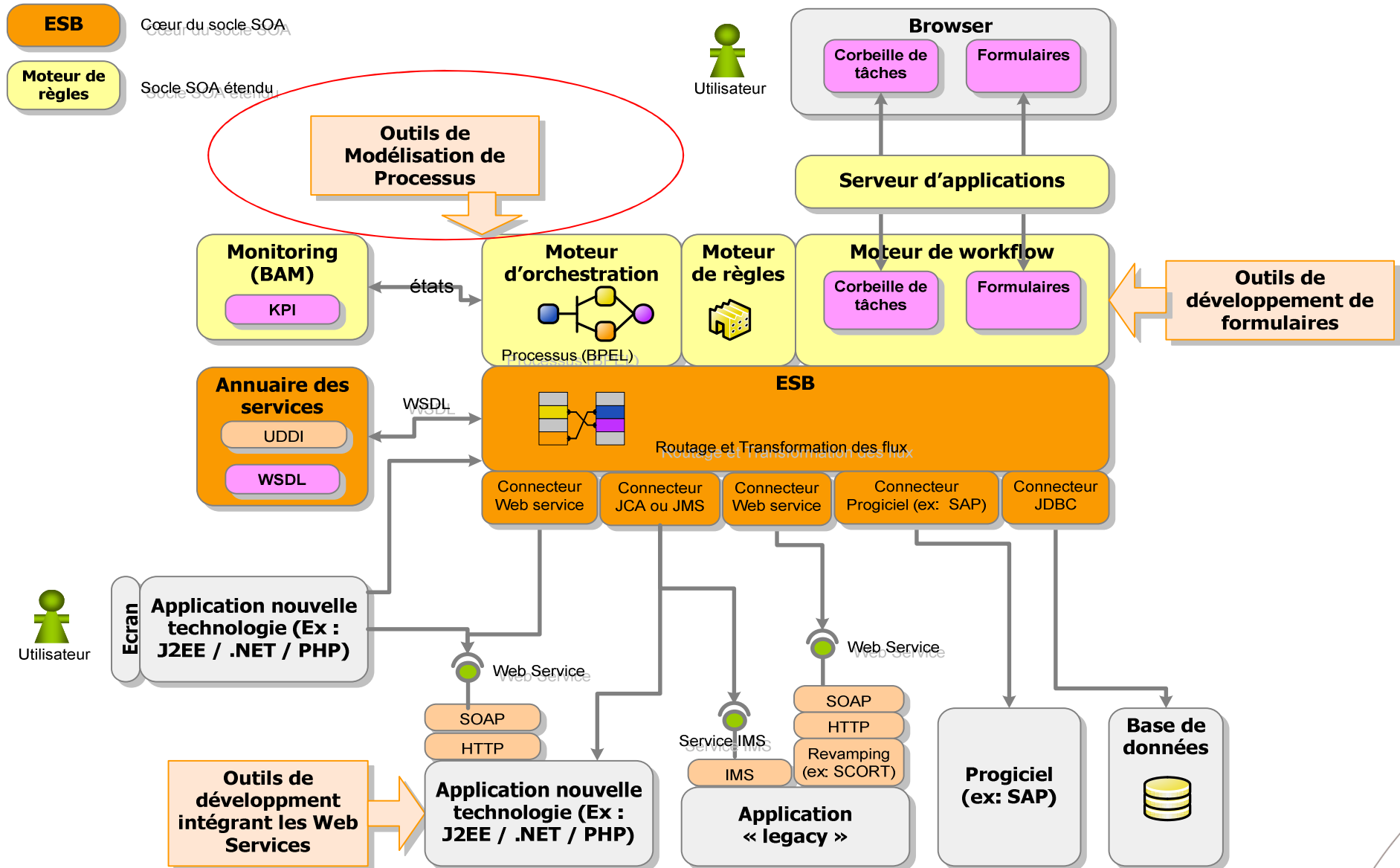
Le cycle projet BPM



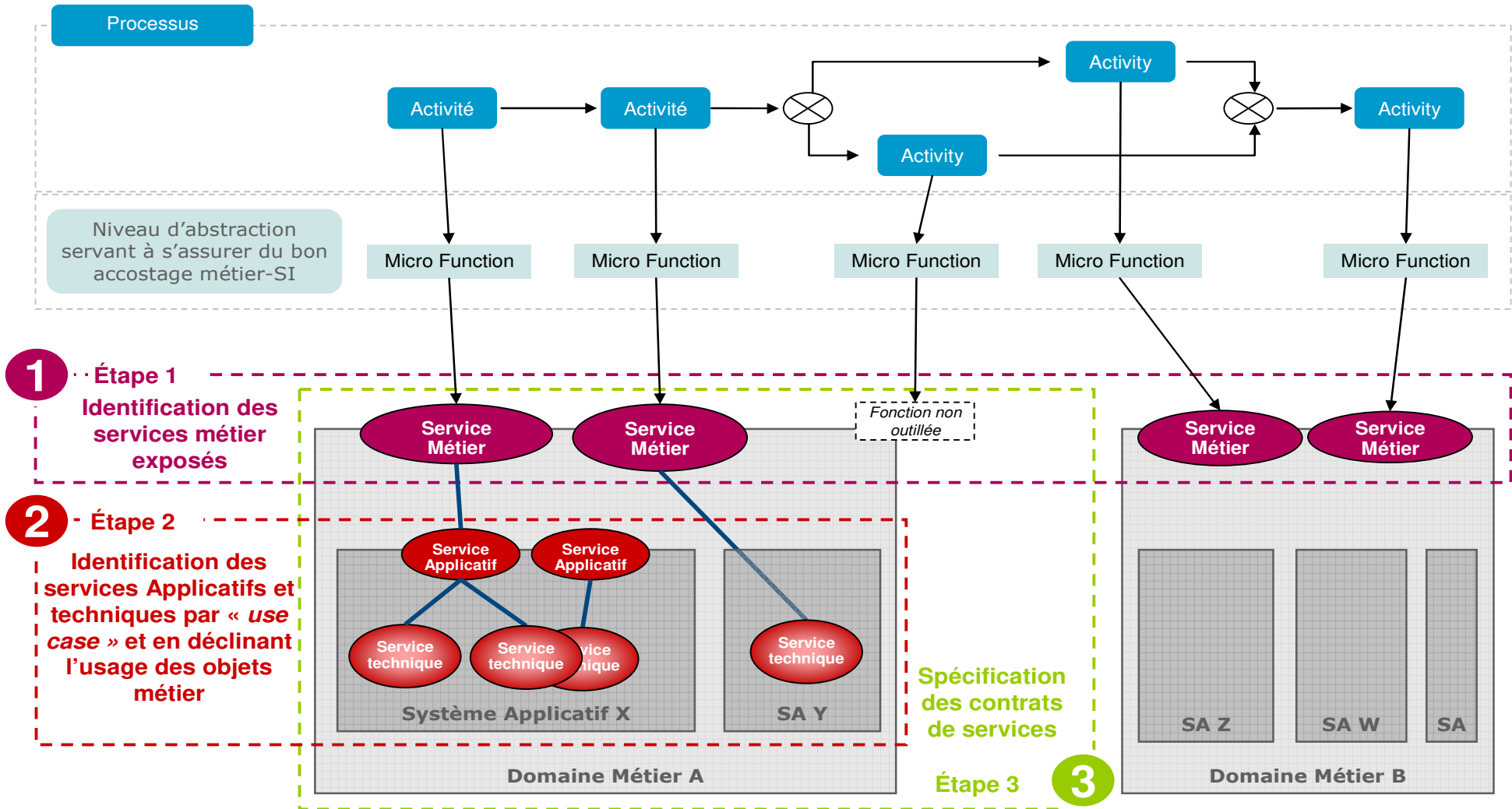
Architecture technique de Référence (Rappel)



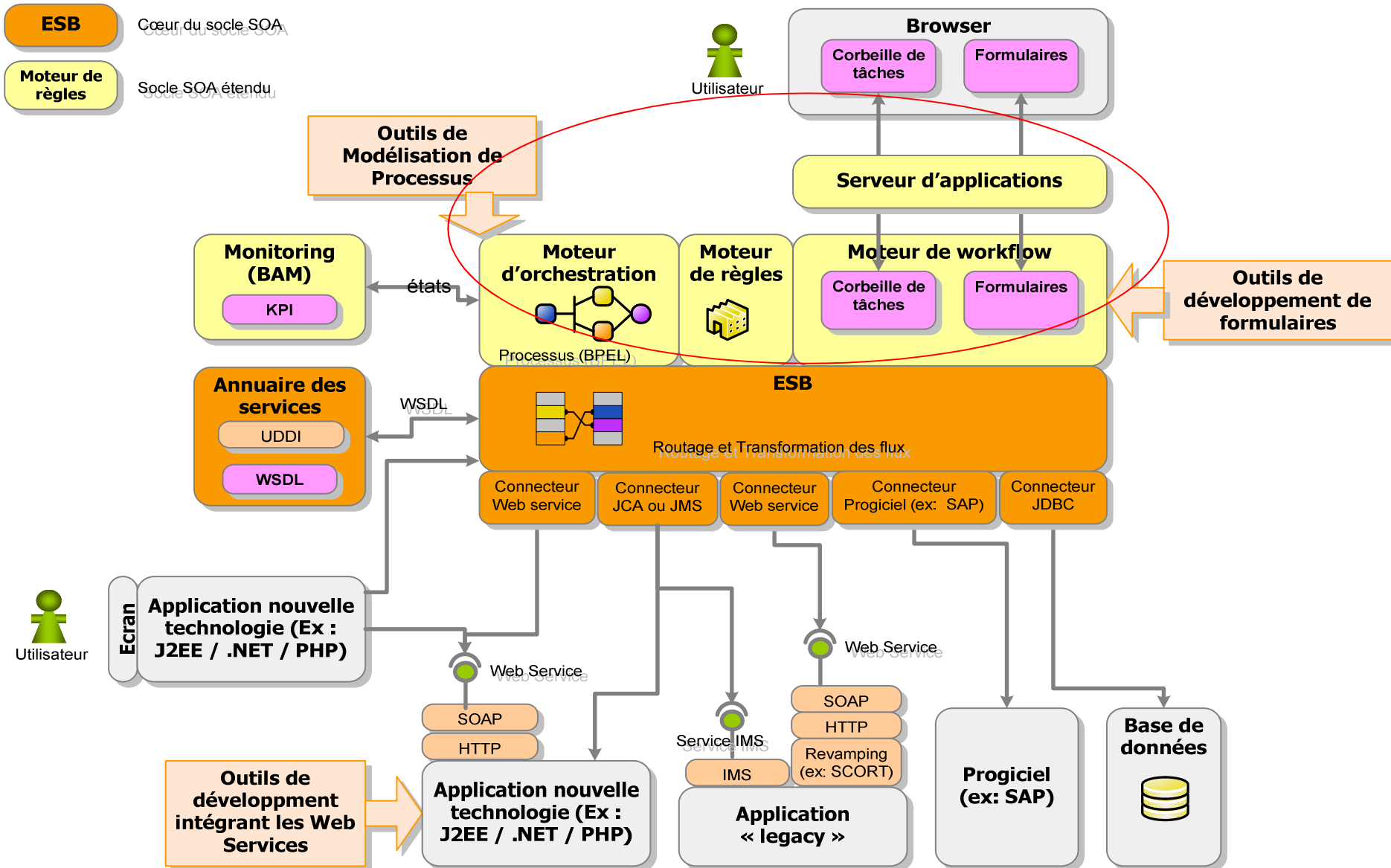
La modélisation des processus



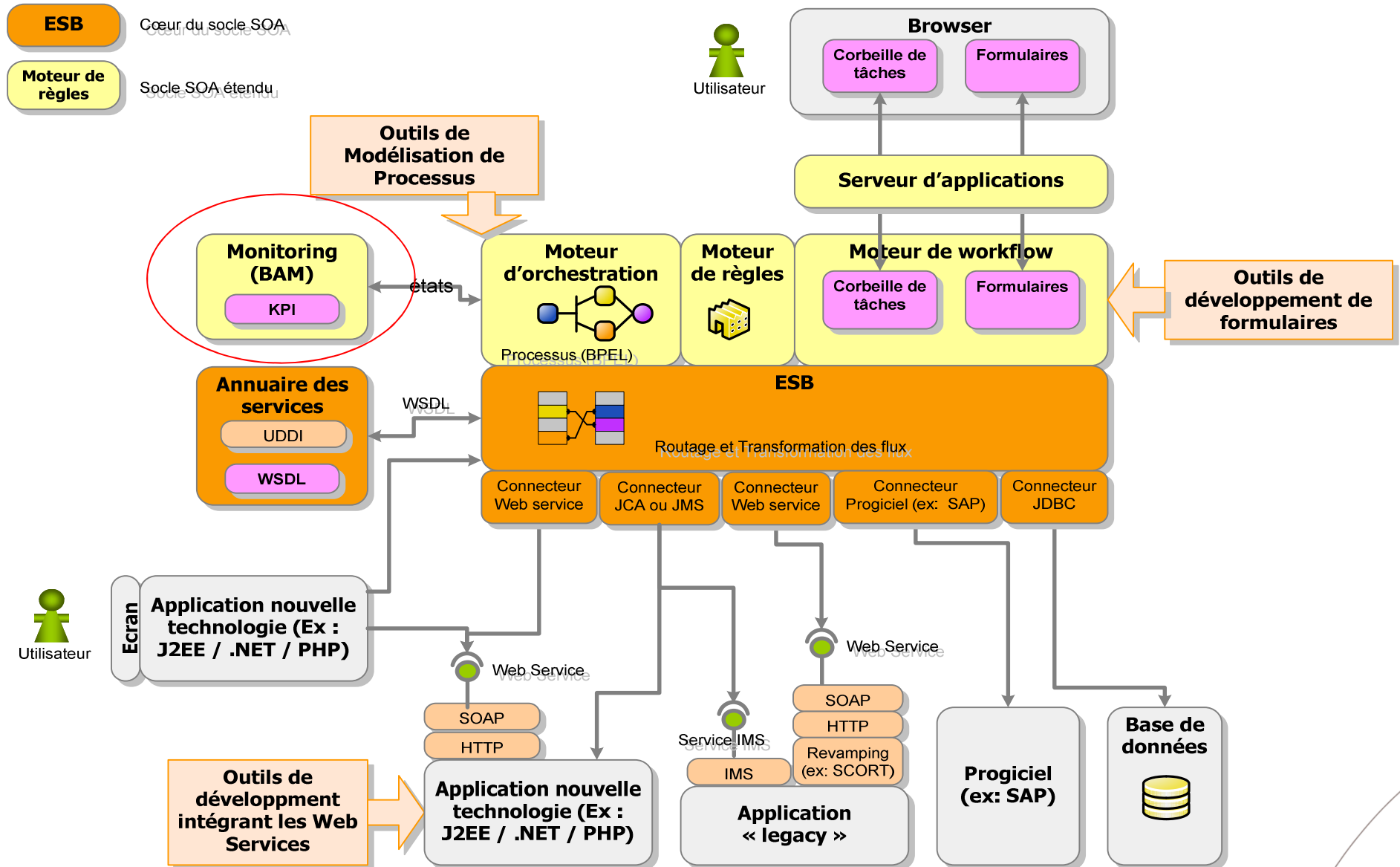
Association Activité / Service



Déploiement et exécution des processus



Supervision des processus



Les composants du BPMS

Idéalement, l'exécution des processus sur l'infrastructure devrait nécessiter la mise en œuvre des composants suivants :

- Moteur de processus
- Coordinateur de processus distribués
- Gestionnaire de ressources
- Ordonnanceur
- Gestionnaire d'audit
- Gestionnaire d'erreurs
- Gestionnaire de sécurité

Les contraintes majeures de l'infrastructure d'exécution sont :

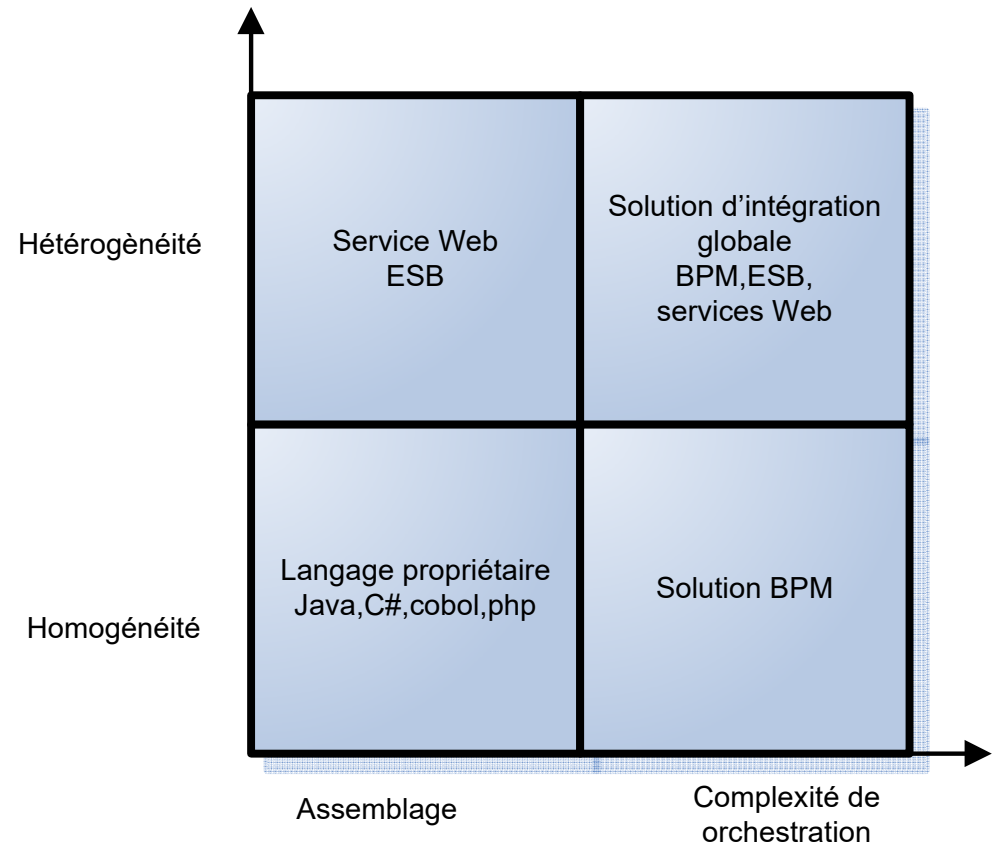
- La disponibilité
- L'extensibilité (scalabilité)

Le BPM et les orchestrateurs

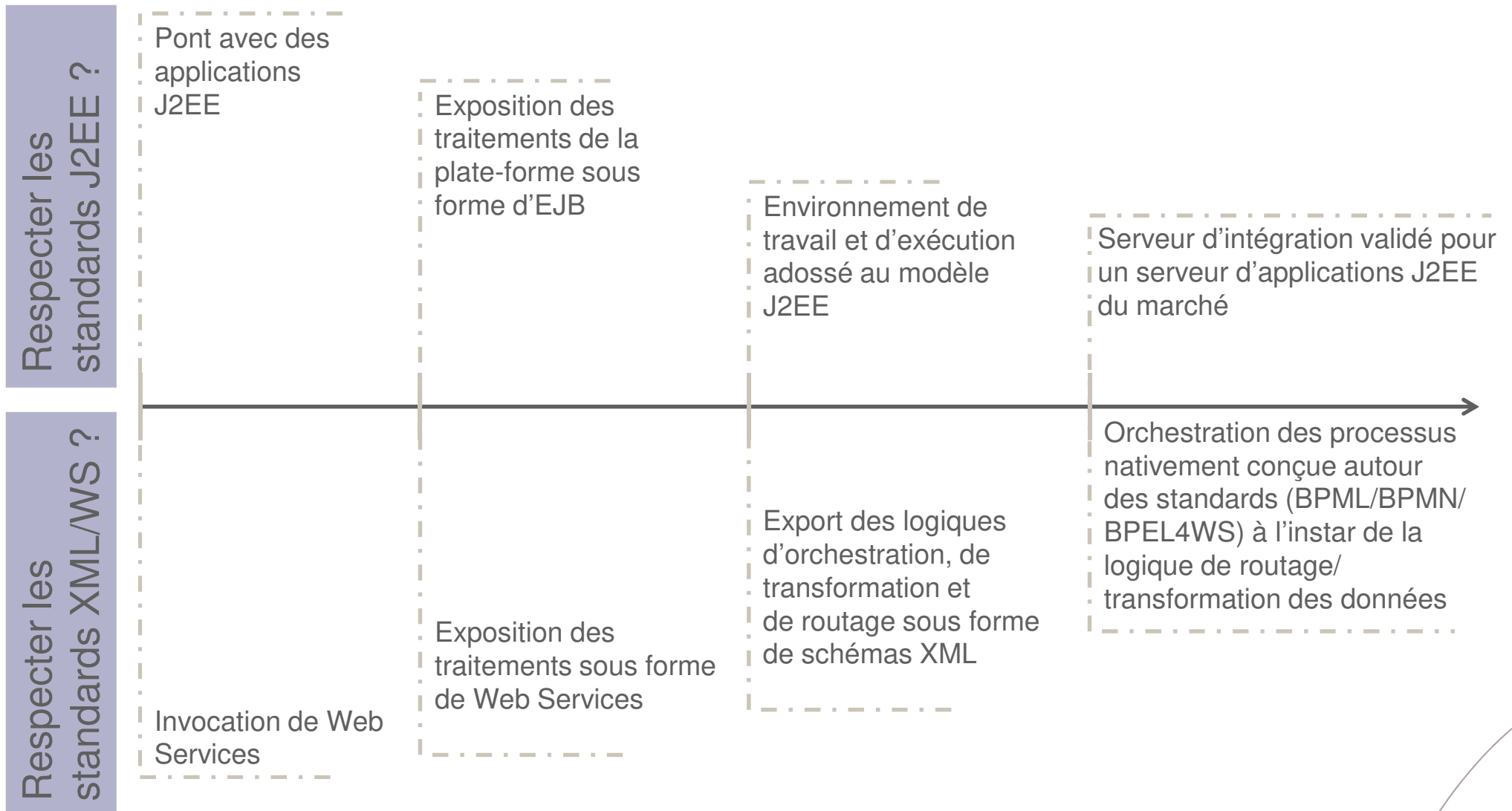
Chaque niveau d'orchestration demande une réponse technologique appropriée

■ Le choix de l'orchestrateur est conditionné par

- Le niveau de cohésion technique,
- Le niveau d'orchestration,
- La complexité de l'orchestration
- L'évolutivité attendue du système



La typologie des solutions d'intégration : vue technique





/ **02.3**

L'annuaire de service

Brique SOA – Annuaire de services

Définition

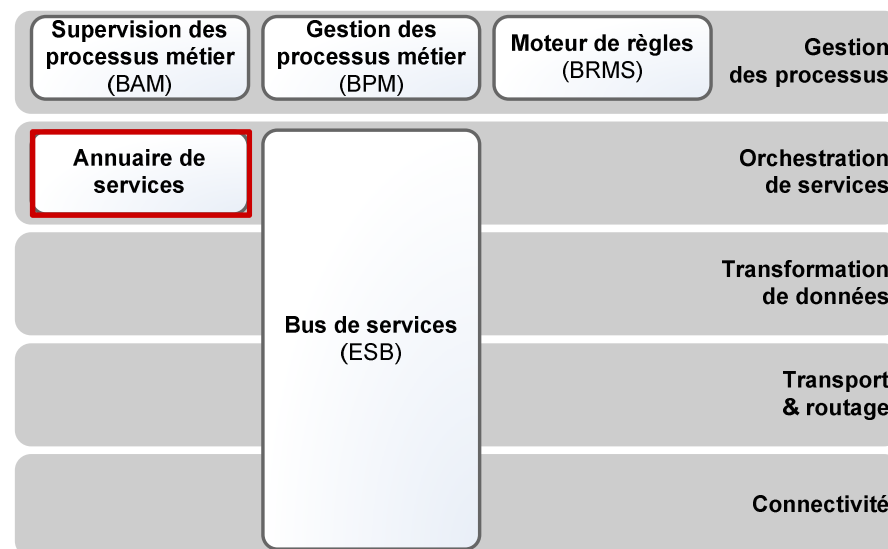
Définition

Un annuaire référence l'ensemble des services (et des contrats associés) disponibles au sein du SI. Les annuaires UDDI forment aujourd'hui le standard de référencement des services.

Dans le cadre de la SOA, l'annuaire est une brique incontournable en tant que **tiers d'intermédiation** et **gestionnaire du catalogue de services** de l'entreprise.

Depuis 2007, les annuaires de services ont fortement évolué afin d'intégrer **la gestion du cycle de vie** des services allant de la conception à l'exécution. Le marché parle aujourd'hui de repository / registry.

Les annuaires de services s'appuient principalement sur les standard décrit ci-dessous.

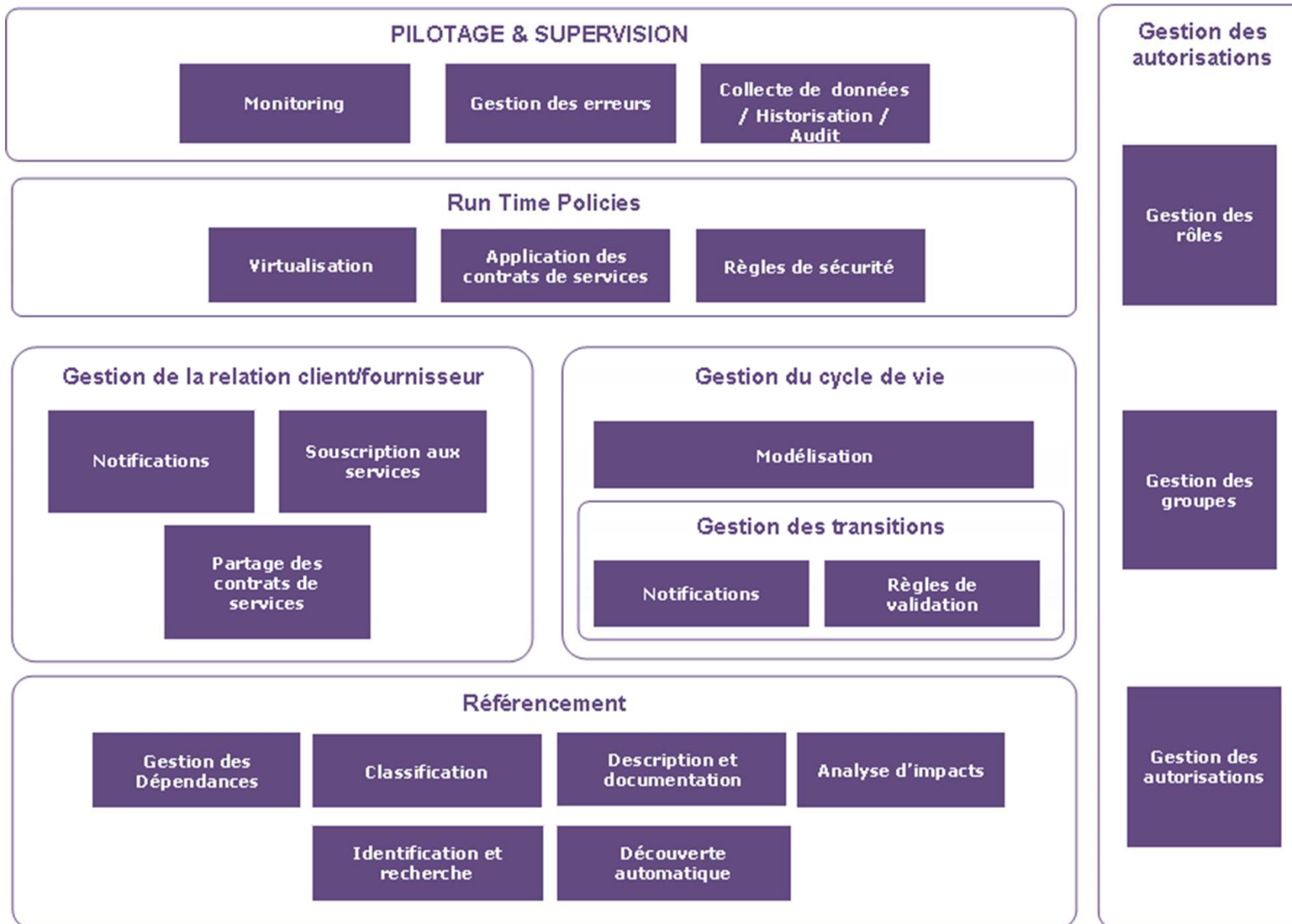


Standard

- **Web Services** pour gérer l'exposition des services.
- **WSDL (Web Services Description Language)** contrat de service décrivant l'interface d'accès au Web Service.
- **UDDI (Universal Description Discovery and Integration)** correspond à une spécification du consortium OASIS du registre des services web. UDDI permet la découverte des services.
- **WS-*** Ensemble de standards autour des Web Services.

Présentation du référentiel de services

Fonctions



A decorative purple curve starts from the left edge of the slide and curves upwards and to the right, ending near the top center.

/ **02.4**

Le Business Activity Monitoring

Brique SOA - BAM

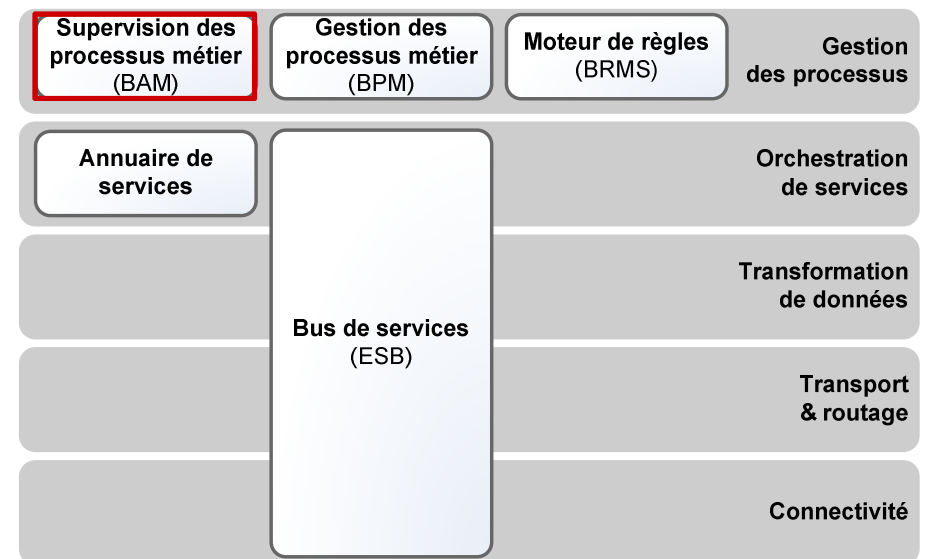
Définition

Définition

Le **BAM** (**Business Activity Monitoring**) a pour objet de suivre l'évolution des processus métier. Il permet de contrôler en permanence la performance des processus clés de l'entreprise.

Du point de vue technique, le BAM exploite les concepts d'intégration et de communication des applications, dont la SOA. Il s'intègre dans une dimension plus globale de BPM.

Du point de vue de l'utilisateur, responsable d'activités ou de processus, la solution BAM se concrétise en tableaux de bord de pilotage composés d'indicateurs de performance, KPI (Key Performance Indicators), rafraîchis en temps quasi-réel.



Brique SOA - BAM

Principales fonctionnalités

SUPERVISION DES PROCESSUS ET MESURE DU SLA

Suivi en temps-réel des processus et des activités métier afin de s'assurer de leur bonne exécution de manière massive ou bien unitairement pour chaque objet mis sous surveillance (objet métier, message, fichier, ...).

D'autre part, le suivi des processus permet aux outils de BAM de qualifier le SLA de chaque processus.

MESURE DE LA PERFORMANCE

Pilotage en temps-réel d'indicateurs de performance (KPI). Ces indicateurs peuvent prendre de multiples formes : durée d'exécution d'une ou de multiples étapes d'un processus, volumes d'objets (objet métier dans un statut métier donné, message parvenu à une étape du processus, fichier reçu ou émis), volumes et taux d'erreur, montants financiers liés à un volume d'objets métier traités...

Ces indicateurs peuvent être remontés via des alertes paramétrables au travers de l'IHM de la solution.

FONCTIONS PRÉDICTIVES

Capacité à fournir des fonctions d'aide à l'identification, la qualification, la résolution et l'anticipation des dysfonctionnements.

Il s'agit par exemple :

- de fonctions d'analyse de cause (identification des instances de processus impactées par un même dysfonctionnement ...).
- d'analyse d'impact (détermination automatique du niveau de sévérité d'un problème, indicateurs de suivi du respect du niveau de service ...).
- d'analyse de risque (non-réception d'une donnée de la part d'un système dans un délai suffisant pour effectuer un traitement métier programmé ...).

GÉNÉRATION DE TABLEAUX DE BORD

Capacité à fournir des fonctions d'analyse de données et de restitution graphique avancée. Ces fonctions facilitent la génération de tableaux de bord.



/ **03.0**

Architecture Micro-service et approche
container

A decorative purple curve starts from the bottom left and arcs upwards and to the right, ending near the top left of the slide.

/ **03.1**

Architecture Micro-service

Micro-service value proposition : *Simpler, stronger, faster*



Agility

Ability to evolve/adapt the application **quickly** and easily to reduce time to market

- ✓ Increase speed of the evolution
- ✓ Increase ability to adapt the scale of the feature really needed



Efficiency

Ability to perform the expected features **in the best possible manner** with the least waste of time and effort

- ✓ Use the best suitable technology stack for a dedicated purpose
- ✓ Reduce adherence to a stack for easier migration and obsolescence management



Tackle complexity

Split the problem in collaborative small pieces of work **suitable for human understanding**

- ✓ Easier understanding of a system purpose
- ✓ Small team is able to manage the topic from end to end
- ✓ Easier coordination thanks to well bounded services

But micro-service bring back “Distributed Systems” challenges ...



1. **Architecture topics** : choose and implement an inter-process communication mechanism, handle partial failure, partitioned database consistency, tests strategy, understand the whole system and ensuring correctness
2. **Organisational topics** : align organisational unit, break the silo
3. **Ops topics** : manage the scale (deploy, configure, scale, monitor several moving parts),

Micro-service : principle

// A micro-service is a **small** software component that can be **deployed independently** from other services. //



Small

- ✓ Able to deliver a **useful feature** as much as possible in autonomy
- ✓ Can be **handle by a small team** (*one team*)
- ✓ **Data use case approach first** in order to be able to **ensure consistency** (transaction boundary or compensation mechanism)
- ✗ Does not rely on number of lines of code
- ✗ Is not one for one function/method called by a remote procedure call (RMI / HTTP)



Deployed independently

- ✓ Can **be technically deployed without any impact on other services**
- ✓ Is **as much as possible decoupled** (own database, features exposed with API, etc.)
- ✗ Deployment becomes easy without improvement on deployment process/tooling

Micro-service : small and deployable independently are not enough



Design for Failure

Design and develop micro-service in a way that it survive or can be re-deploy in any case of incident

- ✓ Breakdown prevention
- ✓ Breakdowns simulation in order to test resiliency

Implementation examples :

- ✓ Idempotent API
- ✓ Graceful degradation



Scalability

Design and develop micro-service in a way that it can handle a growing or decreasing amount of work

- ✓ Context management
- ✓ H-Scaling

Implementation examples :

- ✓ Stateless services
- ✓ Optimized stack



This requirements should be considered from **software** (caller and called micro-service) and **infrastructure** (multi-instances, LB, multi-area) **points of view**

Micro-service : When, where and difference with other paradigms

Monolithic (1980)



Tight coupling
Fully centralized

SOA (2000)



Looser coupling
Partially centralized
IS scope

Micro-services (2010)



Decoupled*
Fully distributed
Application scope



When

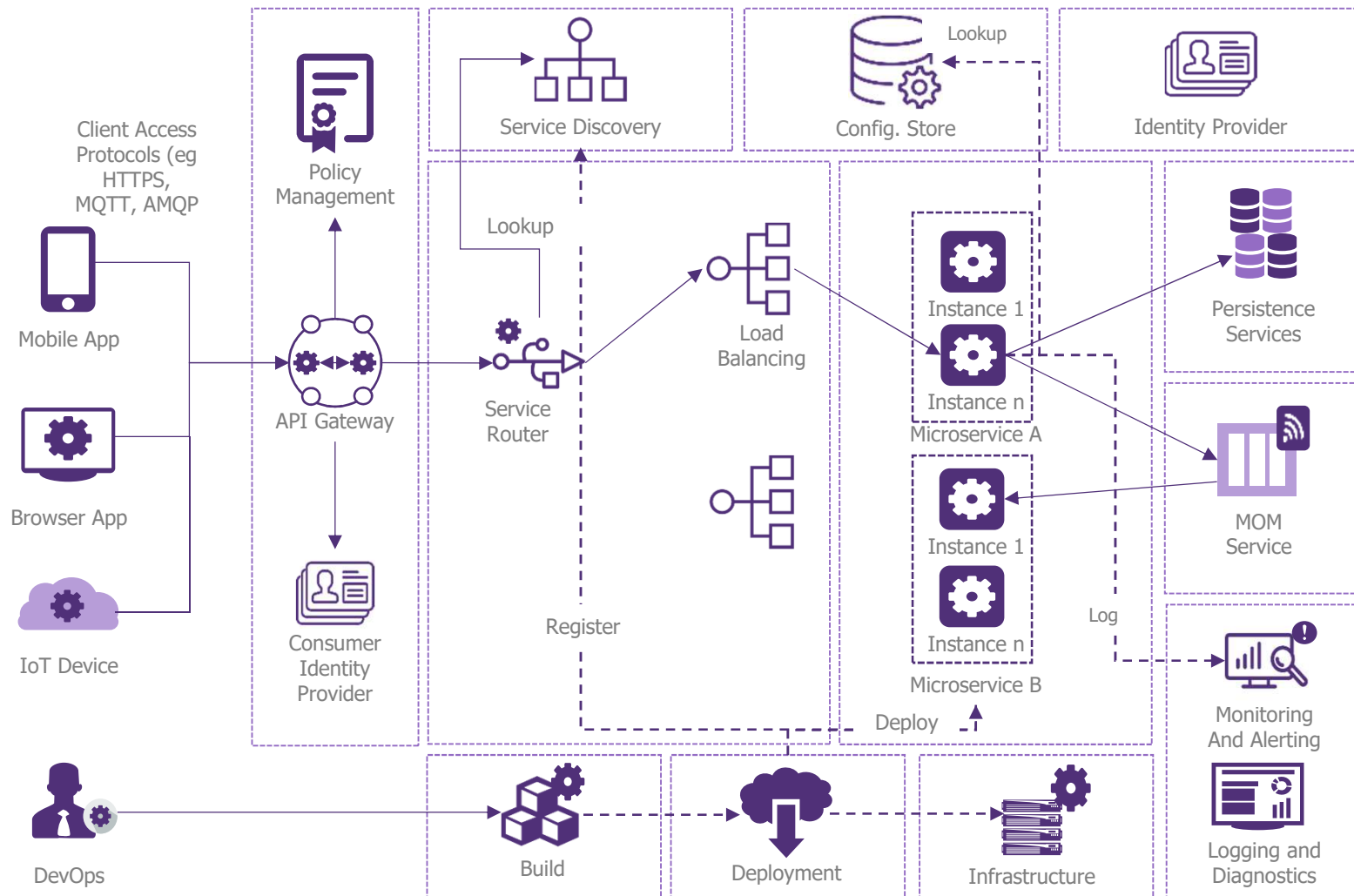
- ✓ **A important need of agility :**
 - ✓ Functional scope have to evolve very fast
 - ✓ Workload is very variable
- ✓ **A need of a specific technology stack**
- ✓ **Organizational issue (team size too big, etc.)**
- ✓ **Unable to maintain the application**



Where

- ✓ **New application**
- ✓ **Existing one (opportunity approach)**

Micro-service : Outer Architecture overview





/ **03.2**

Container

Docker & micro service architecture

Micro-services problematics

- Which tool and deployment process to set up ?
- How manage efficiently micro-services ?
- How to insure a well monitoring in an oriented micro-services architecture ?

Solution

Docker, actual **standard** of containerization on the market, simplify the implementation of an oriented micro-services architecture. Each micro-service can be **packaged in a container**. The **deployment** as well as the **maintenance in operational condition** are simplified

How does Docker help ?



Build

Package a micro-service in a container



Ship

Move that container from a machine to another



Run

Execute that container



Any app & anywhere

Anything that runs on Linux
Local VM, Public/Private Cloud

- Docker provides **an ideal environment** for this type of architecture, because it allows :

- ✓ Environment isolation
- ✓ Faster & seamless deployment
- ✓ Portability

- ✓ Cloud compatibility (public/private/hybrid)
- ✓ Scalability
- ✓ Simplified test



- Docker implementation can be made only with **a dedicated orchestration solution** (type PaaS/CaaS) to this ecosystem, in order to insure some features such as cluster management, service discovery, rolling updates, provisioning, monitoring etc.
- This solution must be able to **adapt in real time** to constant and fast evolution of containers (volatile & short-lived)
- Several solutions exist, some are more mature than others



Les principaux drivers de l'adoption de Docker

Agilité

- Réduction du time to market
- Disparition du problème « chez moi, ça marche »
- Plus de flexibilité pour les Devs et de stabilité pour les Ops
- Une gestion de l'infrastructure simplifiée

Utiliser Docker pour builder et/ou exécuter des applications



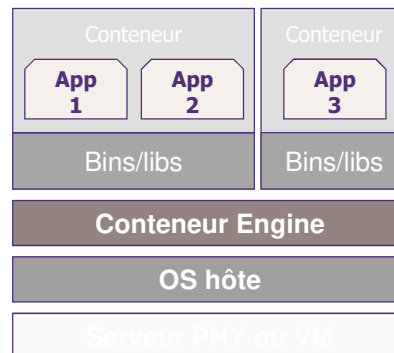
Build

Optimisation

- Un faible overhead : une baisse de la consommation de RAM 4 à 30 fois
- Rationalisation des coûts (moins de licences et d'OS)
- Des mises à jours simplifiées et sans temps d'arrêt



Cycle de déploiement d'un conteneur



Run

Déployer des services avec une scalabilité accrue en implémentant une solution d'orchestration

Portabilité

- Facilite la transition vers le Cloud
- Migration d'une plateforme à une autre de manière transparente
- Compatibilité avec le cloud public/privé, serveur physique ou virtuel



Ship

Intégrer Docker dans le process d'intégration continue et homogénéiser les environnements (dév, recette, prod)

Docker : un standard

- Une grande communauté autour de la technologie Docker
- Une évolution rapide : une adoption multipliée par 5 entre 2014 et 2015
- Adopté par les plus grands acteurs du Cloud

