2éme année Master Informatique

Option: **Génie Logiciel et Systèmes Distribués**

Université de Biskra

Année 2021/2022

Architectures Orientées Services

Dr. Mohamed Lamine KERDOUDI

Email : **l.kerdoudi@univ-biskra.dz**

Composition de services : Motivation

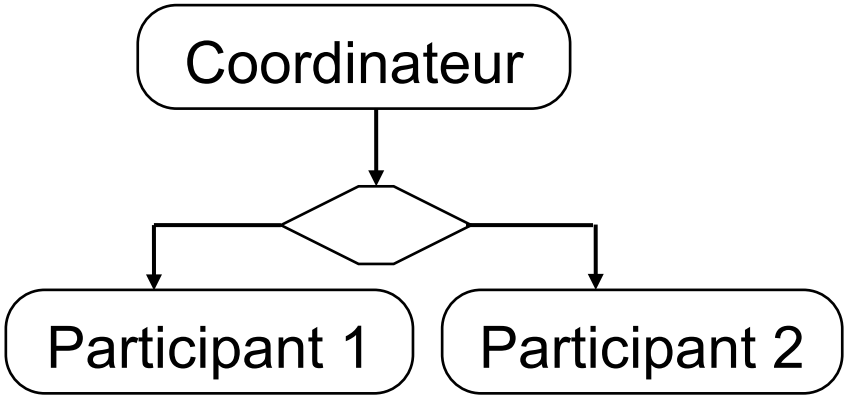
- Une application B2B nécessite d'invoquer un ensemble de services Web dans un ordre précis et selon une logique bien définie.
- SOAP, WSDL et UDDI ne s'intéressent pas à ce problème.

➔ **Solution: Composition de services:** est un concept fondamental dans SOA.

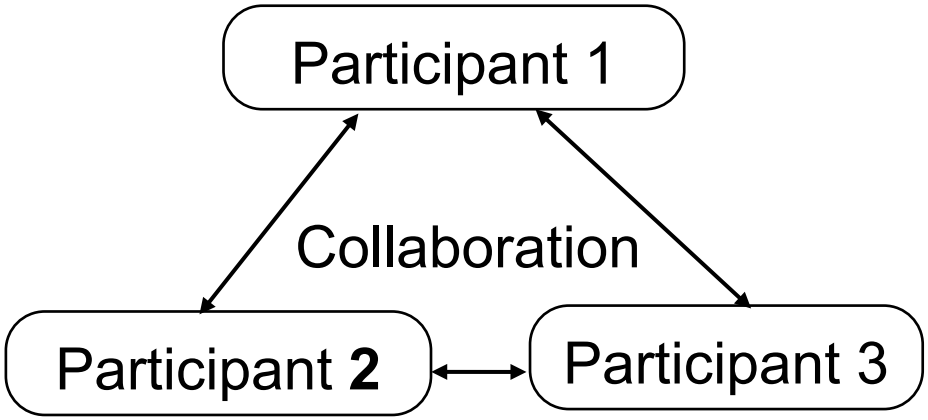
- Un service composite est une agrégation de services composés pour automatiser une tâche ou un processus d'affaires.
- La composition de services Web impliquent une **coordination** ou un **contrôle** acte de rendre les services Web individuels travaillent ensemble pour former un processus global cohérent.

Types de composition de services

- **Orchestration:** par convention elle fait référence à la coordination des services au niveau du **processus d'un seul participant** (qui peut être un autre service)
- **Chorographie:** elle se réfère à la **vue globale**, couvrant **plusieurs participants**



Orchestration

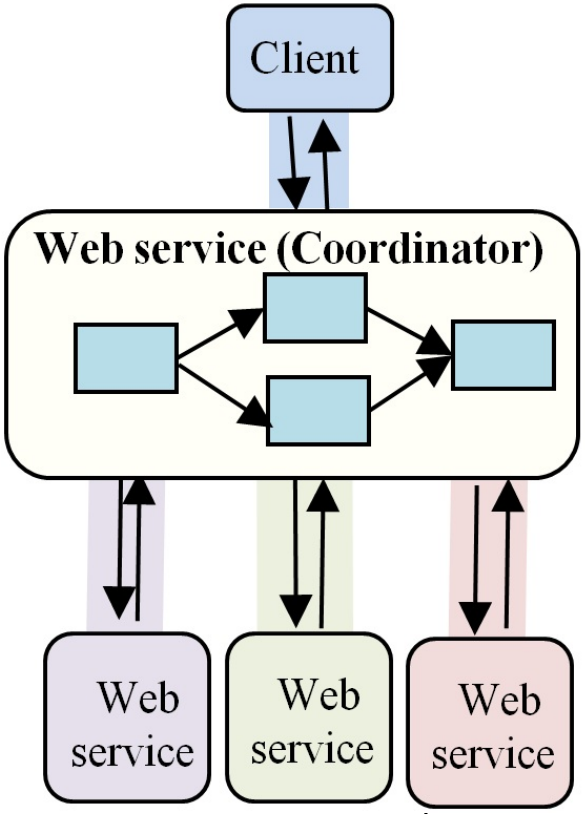


Chorégraphie

Composition de services Web

- **Orchestration:** une orchestration de services Web consiste à invoquer plusieurs services Web en fonction d'un **processus métier** bien défini et à les **exposer** en tant qu'un **service Web unique** (un service Web composite).

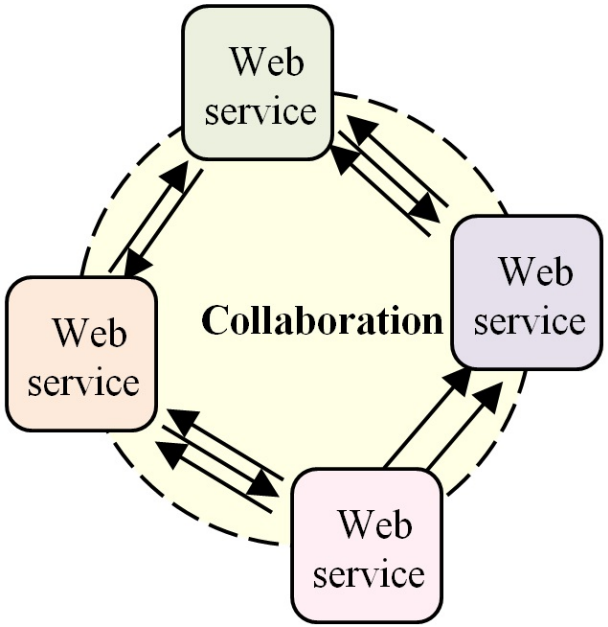
- L'orchestration est donc **centralisée** avec des définitions explicites des opérations et l'**ordre d'invocation** des services Web.
- Les services Web impliqués **ne savent pas** (et n'ont pas besoin de savoir) qu'ils **sont impliqués** dans un processus de composition
- **Le client (dans la figure) peut être un autre service Web** qui invoque les opérations du composite de service Web.



Composition de services Web

- **Chorographie :** La chorégraphie est représentée par un échange de messages entre les services fournis par un ensemble de participants (au moins deux participants), afin d'assurer une certaine interopérabilité.

- Vision collaborative (il n'y a pas de coordinateur central).
- Chaque service Web impliqué sait exactement quand exécuter ses opérations et avec qui il interagit.
- Chaque participant est responsable de la mise en œuvre des décisions de son processus interne
- Chaque service participant pourrait être implémenté en tant que service Web individuel ou orchestration de service Web.



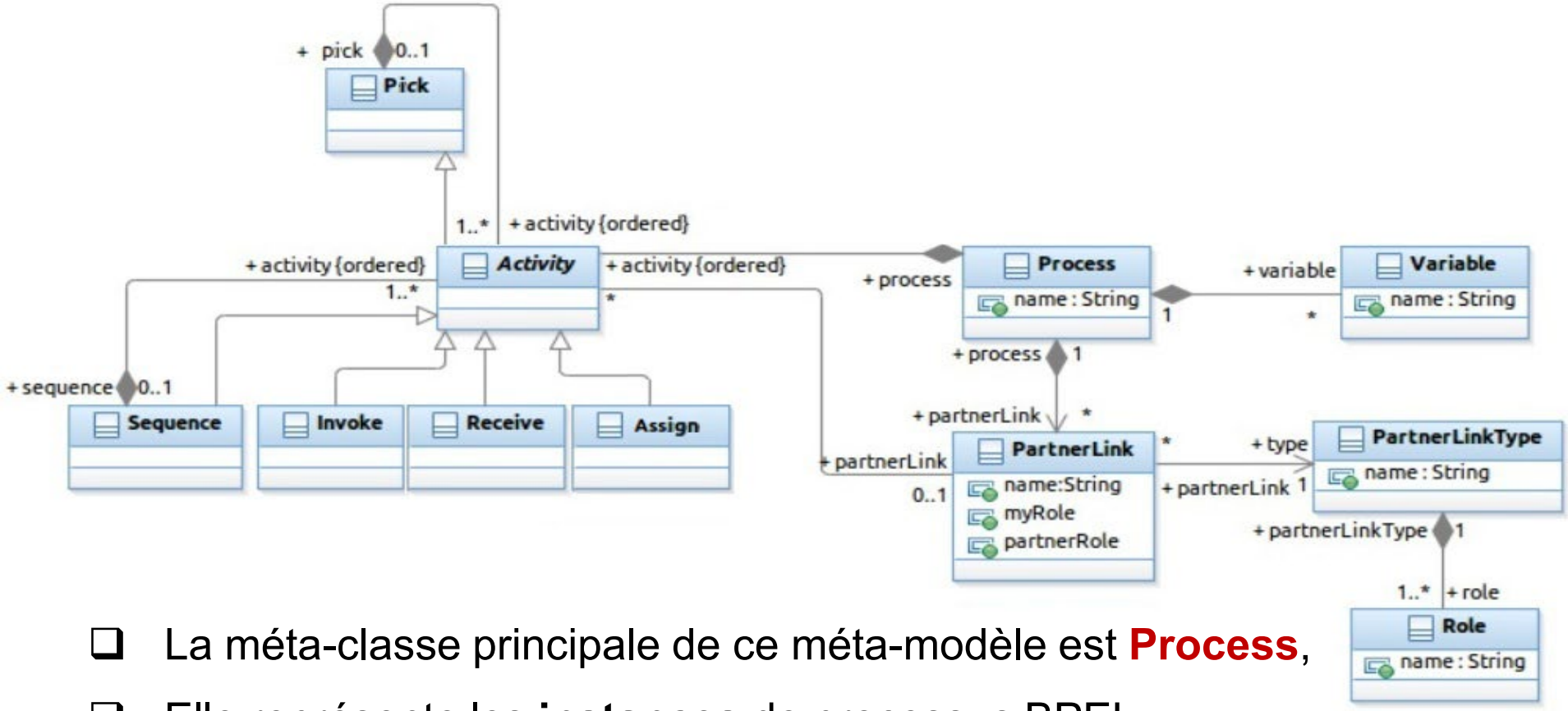
▪ Technologies et Langages de Compositions (Chorographie / Orchestration)

- Business Process Model and Notation **(BPMN)** : est utilisé pour la modélisation de la composition. Il fournit une notation qui soit facilement compréhensible par tous les utilisateurs de l'entreprise.
- Web Services Choreography Description Language **(WSCDL)**
- XML LANGuage (**XLANG)**
- Web Services Flow Language **(WSFL)**
- Business Process Execution Language for Web Service **(BPEL4WS):** prononcé **«** bipeul » **BPEL.**
- Business Process Modeling Language **(BPML)**
- Web Service Choreography Interface **(WSCI)**
- Web Services Conversation Language **(WSCL)**
- ….

▪ Business Process Execution Language for Web services (BPEL)

- **BPEL:** basé sur **XML.** **Origine**: IBM, Microsoft et BEA. **Standardisé** par **OASIS**. BPEL est basé sur **IBM WSFL** et **Microsoft XLANG**.
- Il est considéré comme l'un des **principaux langages** de mise en œuvre des **orchestrations de services Web.**
- Un **processus BPEL** spécifie **l'ordre** dans lequel les services Web concernés doivent être invoqués.
- Permet d'exprimer des comportements **séquentiels, parallèles, conditionnels et en boucle.**
- Par exemple, utiliser un comportement conditionnel si l'invocation d'un service Web dépend de la valeur d'une invocation précédent.
- Il permet également de **déclarer des variables, copier et affecter des valeurs**, définir des gestionnaires **d'erreurs**, etc.
- En **combinant** toutes ces constructions, nous pouvons définir des **processus métier** complexes en tant que spécification d'algorithme.

- **Extrait du Méta-Modèle –BPEL-**



- La méta-classe principale de ce méta-modèle est **Process**,
- Elle représente les **instances** de processus BPEL.
- Un processus BPEL est un ensemble d'étapes, où chaque étape est appelée «**Activity**».
- BPEL prend en charge les activités **primitives** et de **structure**.

Activités primitives de BPEL

- Les activités primitives représentent des constructions de base telles que:
- <**Invoke**>: invocation d'autres services Web.
- <**Receive**>: attendre que le client invoque le processus métier en envoyant un message (requête de réception).
- <**Reply**>: Génération d'une réponse pour les opérations synchrones.
- <**Assign**>: manipuler les variables de données.
- <**Throw**>: Indique les fautes et les exceptions.
- <**Wait**>: Attendre un certain temps.

Activités de Structure de BPEL

• <***Sequence***>: pour définir plusieurs activités qui seront effectuées séquentiellement.

• <***Flow***>*:* pour définir un ensemble d'activités qui seront appelées en parallèle.

• <***Switch***>*:* pour implémenter des branches.

• <***While***>*:* pour définir des boucles.

• <***Pick***>*:* pour sélectionner l'un des plusieurs chemins alternatifs.

- **Autres type d'éléments BPEL: *PartnerLink, Variable, PartnerLinkType***

❑ <***PartnerLink***>*:* les partenaires sont les parties qui interagissent avec le processus BPEL.

- Ils représentent à la fois un consommateur du service fourni par le processus BPEL et un fournisseur de service du processus BPEL.
- Un processus BPEL déclare la liste des ***PartnerLinks*** qu'il prend en charge et, pour chacun, le **rôle** qu'il joue et le **rôle que son partenaire** est censé jouer.

➔Par exemple: **fournisseur d'expédition** ou **fournisseur de planification.**

- Un processus peut avoir un ou plusieurs ***PartnerLinks***.

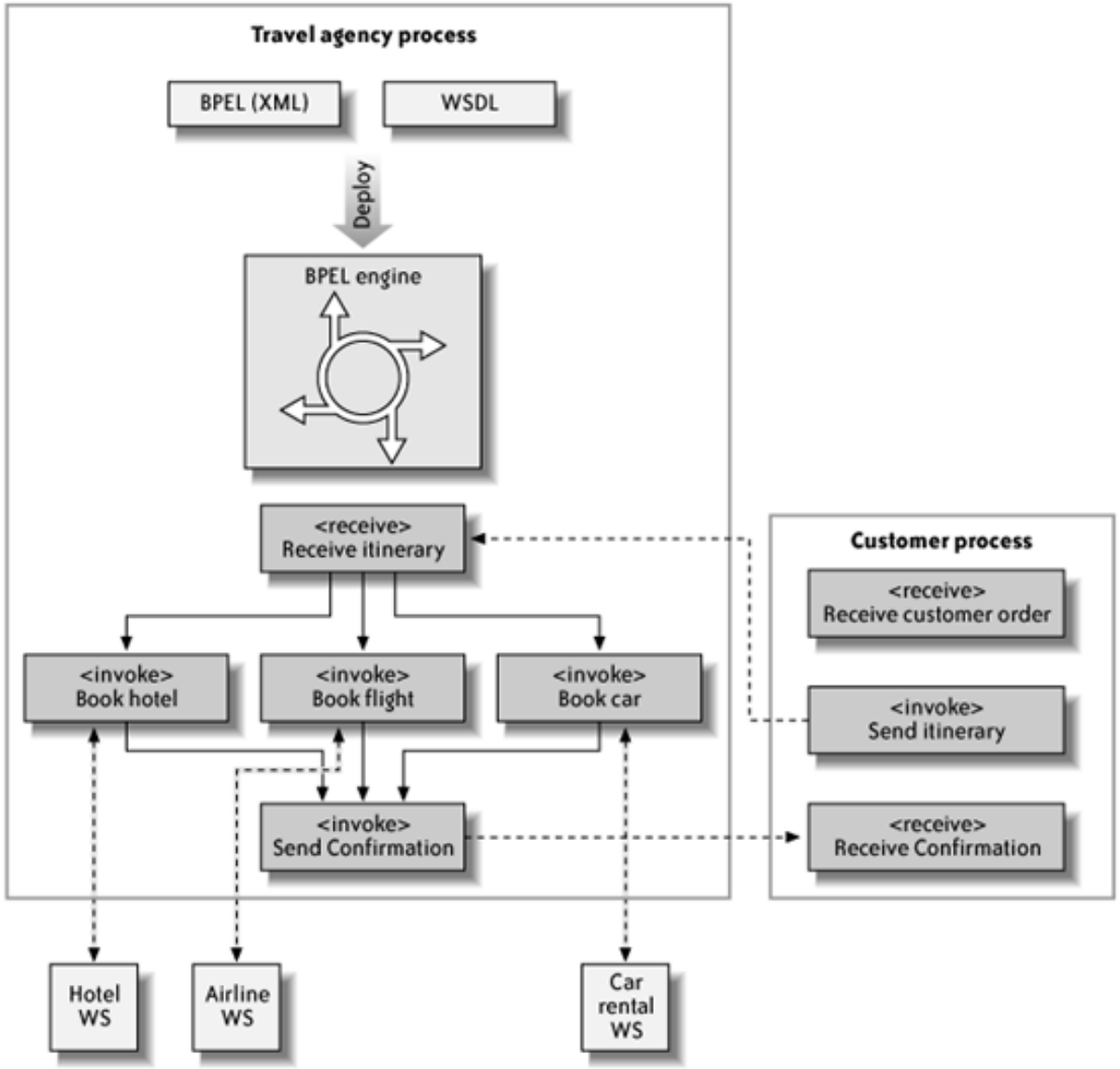
- **Autre type d'éléments BPEL**

❑ <***Variable***> *:* Une variable à utiliser dans un processus, avec un type basé sur un type de message dans un WSDL, ou un type d'élément dans un Schema XSD.

❑ <***PartnerLinkType***>*:* est un mappage des port types de service Web aux rôles partenaires.

➔ Les **PartnerLinkTypes** sont également définis dans les fichiers WSDL des services invoqués via le mécanisme d'extensibilité d'éléments dans WSDL.

- **Exemple Processus d'agence de voyage en BPEL**



- **Structure d'un processus BPEL:** Les caractères ajoutés aux éléments, attributs sont comme suit:
  - "?" (0 ou 1),
  - "*" (0 ou plus),
  - "+" (1 ou plus).

```
<process name="NCName" targetNamespace="anyURI"
  xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable">
<import namespace="anyURI"? location="anyURI"? importType="anyURI" />*
<partnerLinks>?
  <!-- Note: At least one role must be specified. -->
  <partnerLink name="NCName"
    partnerLinkType="QName"
    myRole="NCName"?
    partnerRole="NCName"?
    initializePartnerRole="yes|no"?>+
  </partnerLink>
</partnerLinks>

<messageExchanges>? </messageExchanges>

```

```
16 <variables>?
17    <variable name="BPELVariableName" messageType="QName"? type="QName"?
18     element="QName"?>+ from-spec?
19    </variable>
20 </variables>
21
22 <correlationSets>?
23       <correlationSet name="NCName" properties="QName-list" />+
24 </correlationSets>
25
26 <faultHandlers>?
27      <!-- Note: There must be at least one faultHandler -->
28 </faultHandlers>
29
30 <eventHandlers>?
31     <!-- Note: There must be at least one onEvent or onAlarm. -->
32 </eventHandlers>
33   Activity
34 </process>
```

- **Dans la ligne 33**, nous devons définir une liste ordonnée **d'activités** qui doivent être exécutées par le processus BPEL.
- Un processus BPEL commence à fournir des services à ses partenaires via des activités de message entrant **(<Receive>, <Pick> et <OnEvent>**) et des activités <**Reply**> correspondantes.

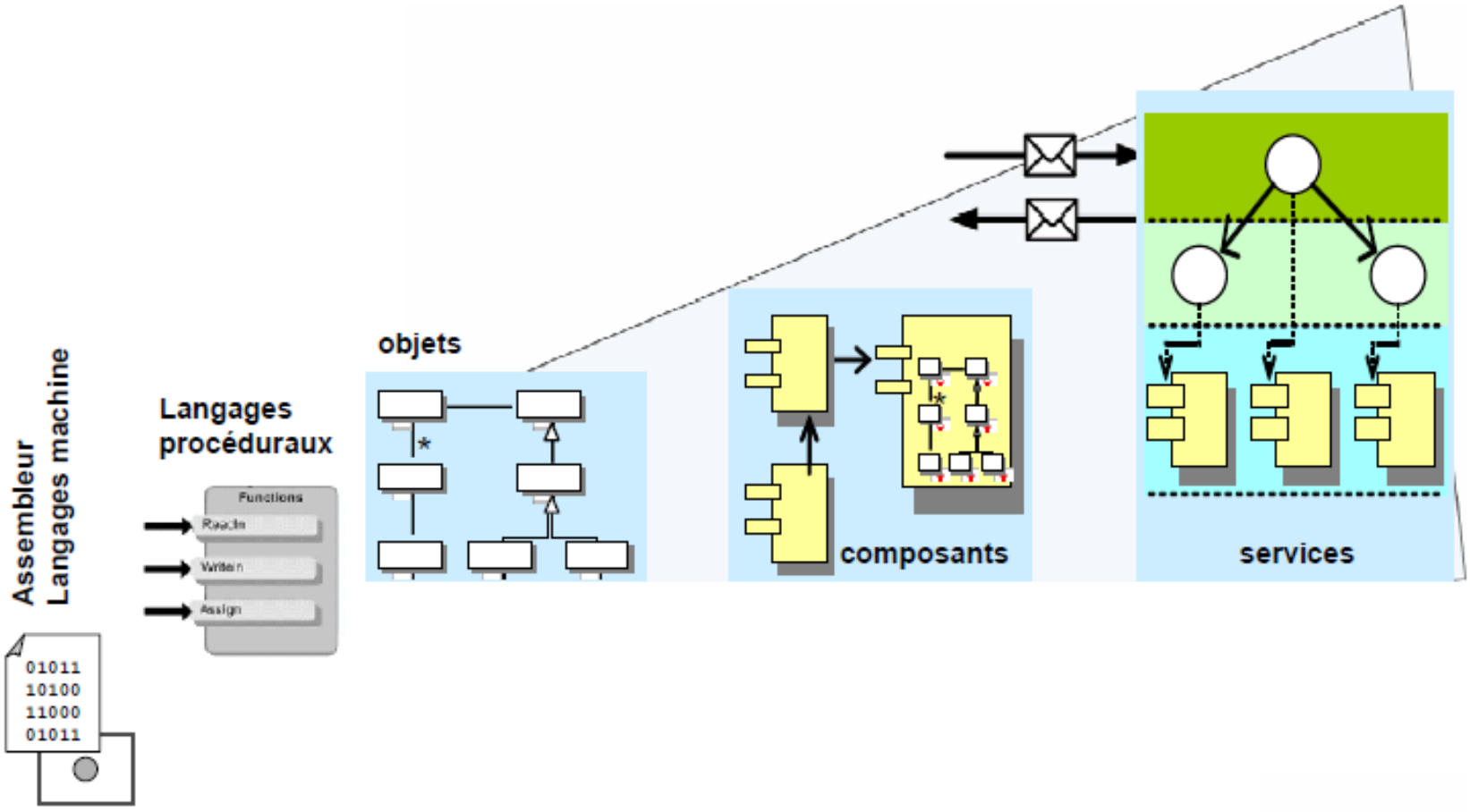
7. Conclusion

SOA est une évolution des plate-forme passées, tout en préservant les caractéristiques réussies des architectures traditionnelles

- **Contractualisation des services**
  - Design by Contract
- **Découplage Interface/Implémentation**, interopérabilité, transparence des communications, …
  - Middlewares à la CORBA
- **Orchestration des services**
  - Travaux autour des workflows, langages de coordination

SOA est une évolution plutôt qu'une révolution

7. Conclusion



Niveaux d'abstraction grandissant

Liens utiles

Voir des liens intéressants :

https://docs.oracle.com/cd/E19182-01/821-0539/cnfg_bpel-create-project_t/index.html

https://docs.oracle.com/cd/E19182-01/821-0539/6nlj8ms62/index.html

Spécification BPEL v2:
http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

Schema UDDI v3:
http://www.uddi.org/schema/uddi_v3.xsd

Questions ??