SOA – Services Web

Introduction SOA

Mickaël BARON – 2010 (Rev. Janvier 2011)
mailto:baron.mickael@gmail.com ou mailto:baron@ensma.fr

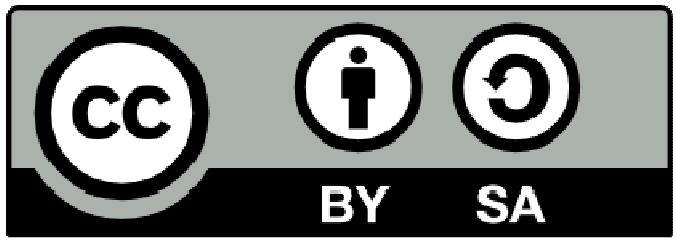
Licence

Creative Commons

Contrat Paternité

Partage des Conditions Initiales à l'Identique

2.0 France



http://creativecommons.org/licenses/by-sa/2.0/fr

Plan du cours

- Vers une architecture SOA
- SOA généralités
- Définition d'un Service
- Services Web étendus et REST
- Plateformes de développement
- Organisation du cours SOA

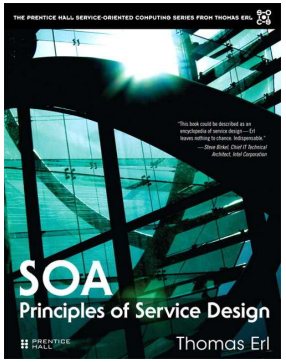
Déroulement du cours : Introduction SOA

- Pédagogie du cours
  - Des bulles d'aide tout au long du cours
  - Comprendre une architecture SOA (problématique, solutions)
  - Pas de technique pour l'instant
- Pré-requis
  - Ingénierie des données
  - Langages de description : XML
  - Architectures multi-couches (Java EE)
- Remerciements
  - Developpez.com [louge]

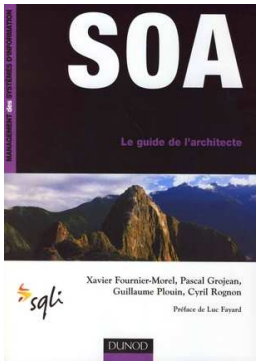
Ressources : Liens sur le Web

- Billets issus de Blog
  - *blog.xebia.fr/category/soa*
  - *jee-bpel-soa.blogspot.com/search/label/web services*
  - *blogs.sun.com/arungupta/*
- Articles
  - *fr.wikipedia.org/wiki/Architecture_orientée_services*
  - *fr.wikipedia.org/wiki/Service_Web*
  - *www.journaldunet.com/solutions/dsi/article/air-france-klm-donne-des-ailes-a-son-systeme-d-information-avec-la-soa.shtml*
  - *www.innoq.com/resources/ws-standards-poster/*
- Cours
  - *www.javapassion.com/soaprogramming/SOAbasics.pdf*
  - *www.javapassion.com/webservices/WebServicesOverview.pdf*

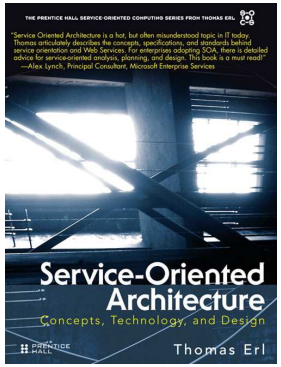
Ressources : Bibliothèque



- SOA Principles of Service Design
  - Auteur : Thomas Erl
  - Éditeur : Prentice Hall Ptr
  - Edition : Juillet 2007 - 608 pages - ISBN : 0132344823



- Le guide de l'architecte du SI
  - Auteur : Xavier Fournier-Morel, Pascal Grosjean, …
  - Éditeur : Dunod
  - Edition : Octobre 2006 - 302 pages - ISBN : 2100499726



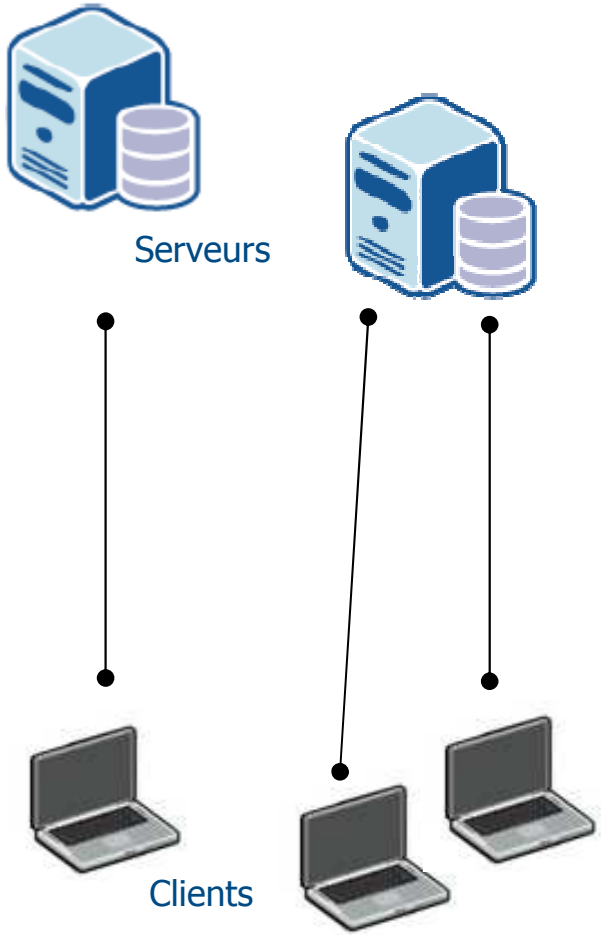
- Service-Oriented Architecture (SOA) : Concepts …
  - Auteur : Thomas Erl
  - Éditeur : Prentice Hall Ptr
  - Edition : Août 2005 - 792 pages - ISBN : 0131858580

Vers une architecture SOA

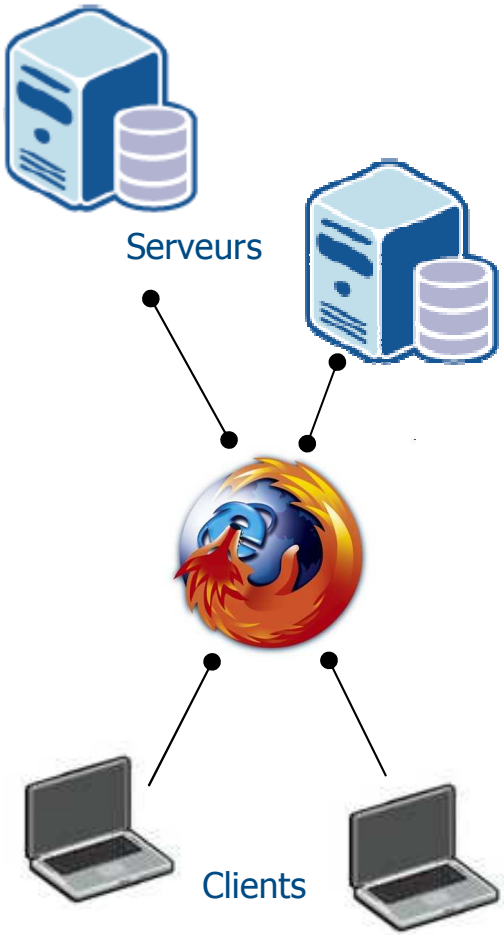
- Une application distribuée est définie par un ensemble de composants
  - Collaborent pour l'exécution de tâches communes
  - Distants géographiquement
  - Interconnectés via un réseau de communication
  - Hétérogènes
- Solutions qui ont fait leur preuve
  - DCOM, CORBA, EJB, RMI, .Net Remoting, …
- Faiblesses de ces solutions
  - Format de représentation données spécifiques
  - Interopérabilité si les composants utilisent la même solution
  - Protocole de transport spécifique nécessite une configuration réseau

Vers une architecture SOA

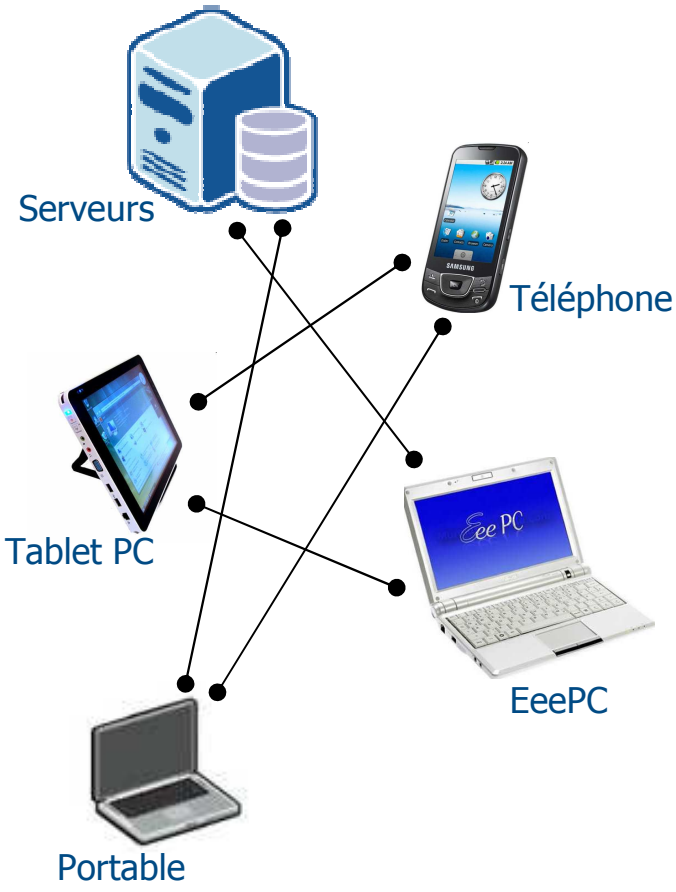
Evolution des applications distribuées



Architecture Client / Serveur



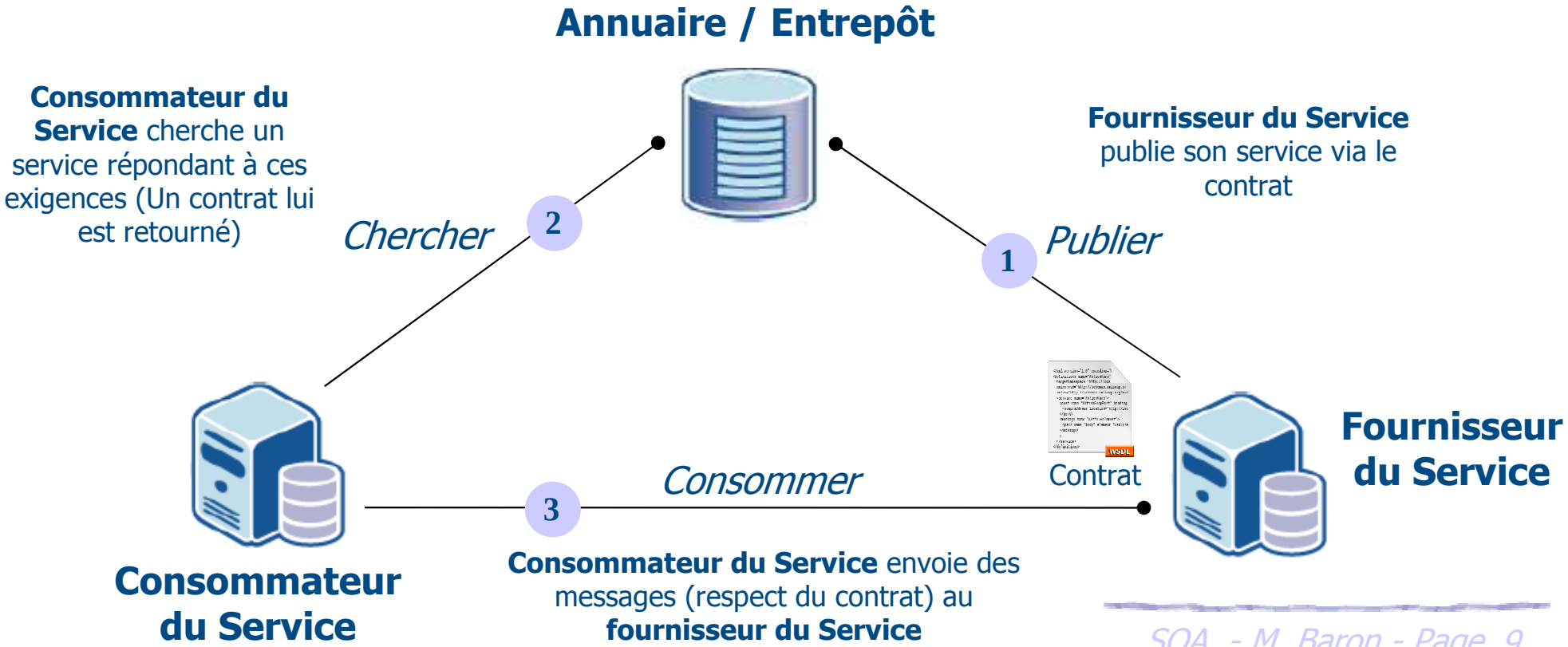
Architecture Fondée sur les Applications Web



Architecture Orientée Service

SOA : Généralités

- SOA est l'acronyme de **S**ervice **O**riented **A**rchitecture qui est traduis comme « Architecture Orientée Service »
- Le Service (ou Composant) désigne le fondement de ce modèle d'interaction entre applications
- Le paradigme SOA : **Chercher**, **Publier** et **Consommer**



SOA : Concepts de Service

- Qu'est-ce qu'un Service ?
  - « Un Service est un composant logiciel distribué, exposant les fonctionnalités à forte valeur ajoutée d'un domaine métier » [XEBIA BLOG : 2009]
- Huit aspects caractérisant un Service
  - Contrat standardisé
  - Couplage lâche
  - Abstraction
  - Réutilisabilité
  - Autonomie
  - Sans état
  - Découvrabilité
  - Composabilité

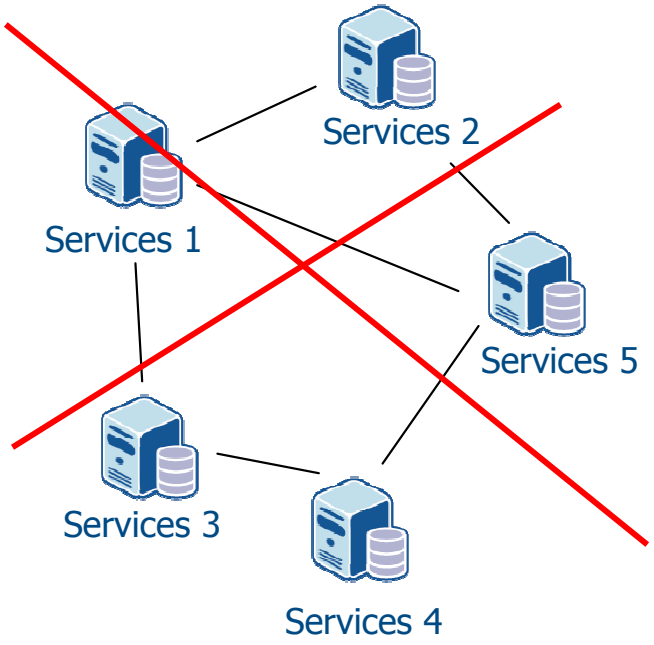
Dans la suite nous détaillons chaque aspect d'un Service

Service : Contrat Standardisé

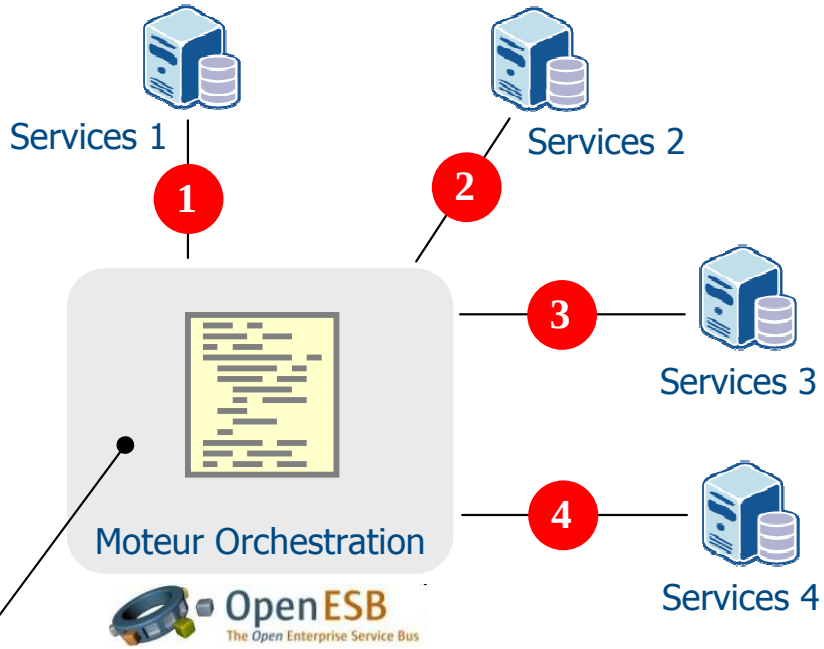
- Contrat entre le fournisseur de service et le consommateur de service
- Trois types de contrat sont à distinguer
  - Lié à la syntaxe du service (opération, messages d'entrée, messages de sortie, …)
  - Lié à la sémantique du service (définition de règles et de contraintes d'usage, …)
  - Lié à la qualité de service (temps de réponse attendu, procédures en cas de panne, temps de reprise après interruption, …)
- S'appuie sur des standards d'interopérabilité pour faciliter le dialogue (exemple : WSDL)

Service : Couplage lâche

- L'échange entre le fournisseur de service et le consommateur doit se faire à travers des messages (couplage lâche vis-à-vis de son environnement)
- L'utilisation d'une orchestration évite que les services aient besoin de connaître les autres services



Couplage fort



Couplage lâche

Etudié dans le dernier cours

Service : Abstraction

- Le contrat du service ne doit contenir que les informations pertinentes à son invocation
- Fonctionnement du service dit en « boîte noire »
  - Seul le contrat exposé au consommateur du service est connue
  - Le fonctionnement interne du service ne doit pas être visible
    - Logique métier
    - Implémentation
- Il est par conséquent important d'assurer la **prédictabilité** d'un service
  - Pas de variation dans le comportement et dans la réponse d'un service lors de la réception d'une requête

Service : Réutilisabilité / Découvrabilité

- Un service doit être accessible depuis un entrepôt ou un annuaire pour faciliter sa découverte
- Le fournisseur de services a la charge de déposer et de mettre à jour ses services depuis l'annuaire
- Le service est enrichi par un ensemble de méta-données pour faciliter la recherche du consommateur de services
- S'appuie sur des standards (UDDI, ebXML)
- D'après la gouvernance SOA
  - Un service est défini avec l'intention d'être réutilisé

Service : Autonomie / Sans état

- Un service doit disposer
  - de l'ensembles des informations nécessaires à son exécution
  - ne doit dépendre d'aucun service externe (couplage lâche)
- Garantir l'autonomie d'un service permet de s'assurer de sa prédictabilité
- Un service doit être sans état de façon à minimiser la consommation de ressources
  - Maintenance : rend complexe la composition de services
  - Performance : gourmand en ressources systèmes

Service : Composabilité

- Un service doit fonctionner de manière modulaire et non pas intégrée
- Assurer la décomposition d'un service complexe en sous services plus simples entre eux (garantie l'autonomie)
- S'inscrire dans une logique de composition de services à travers l'utilisation de l'orchestration (couplage lâche)
- L'orchestration favorise l'indépendance des services et assure que des services n'appellent pas directement d'autres services

Client / Serveur Versus SOA

Architecture Client / Serveur

- Intra-entreprise
- Limitée à un sous ensemble de langages de programmation
- Procédurale
- Protocole de transport propriétaire
- Fortement couplé
- Traitement efficace

Architecture Client / Serveur

VS

- Entre Entreprises
- Indépendance du langage de programmation
- Pilotée par les messages
- Possibilité de choisir le protocole de transport
- Faiblement couplé
- Traitement plus lourd

Architecture Orientée Service

Architecture Orientée Service

Applications « Web » Versus SOA

Architecture Fondée sur les Applications Web

- Interaction Programme / Utilisateur
- Intégration statique des composants
- Service monolithique
- Référencement via des annuaires de sites non standardisés

Architecture Fondée sur les Applications Web

VS

- Interaction Programme / Programme
- Intégration dynamique des services
- Décomposition en sous service avec possibilité de réutilisation
- Annuaires standardisés

Architecture Orientée Service

Architecture Orientée Service

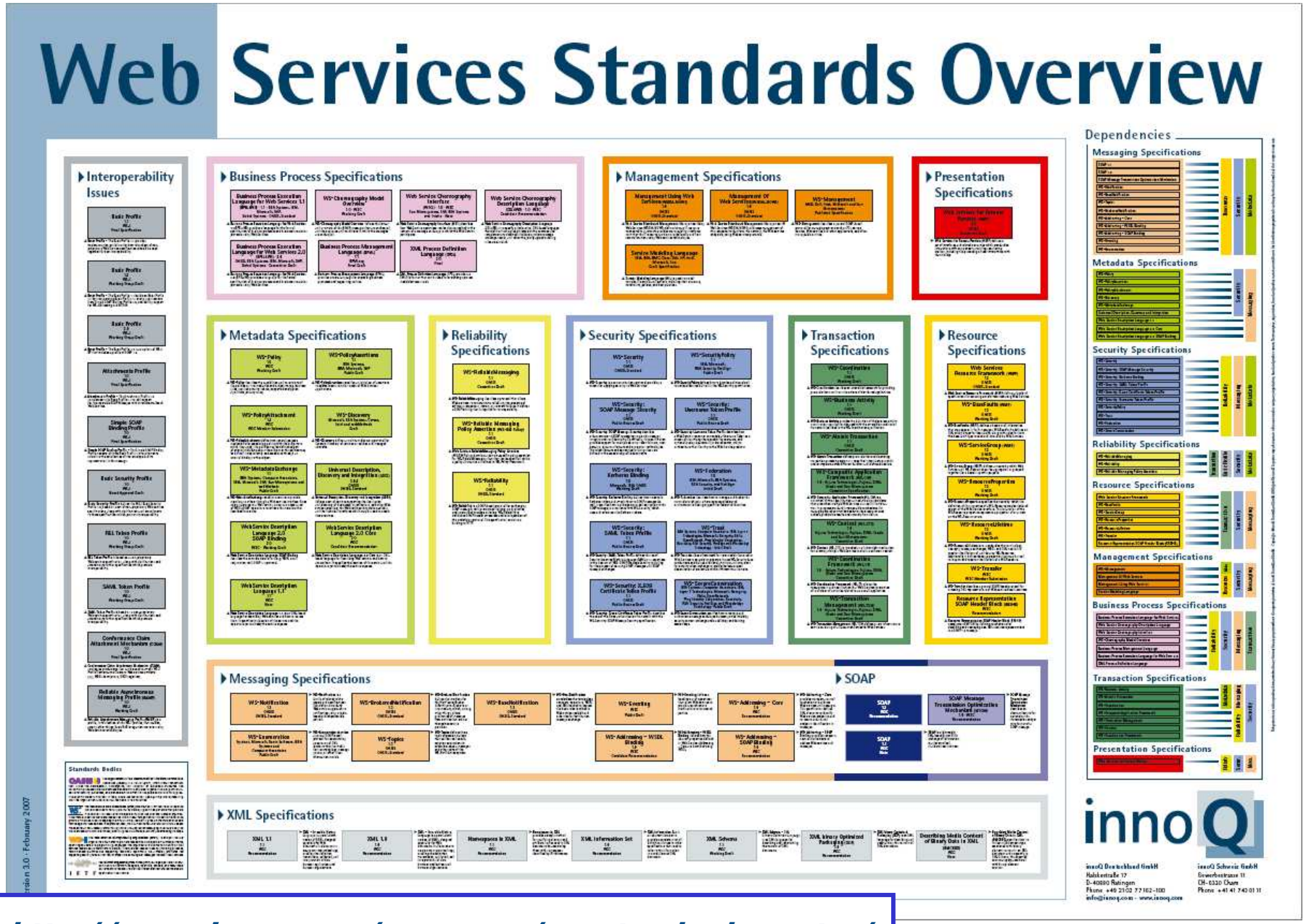
Solutions pour une SOA

- Plusieurs solutions technologiques sont adaptées pour développer une architecture orientée service
  - Services Web (le plus courant)
  - Framework OSGi, …
- Un point sur OSGi (**O**pen **S**ervice **G**ateway **I**nitiative)
  - Spécification définie par l'OSGi Alliance (*http://www.osgi.org*)
  - Longtemps exploité dans le monde de l'embarqué, utilisé dans les serveurs (GlassFish 3, Spring DM) et application (Eclipse)
  - Concepts …
    - **Dynamique** : installé, arrêté, mise à jour, désinstallé
    - **Découvrabilité** : registre des services
    - **Abstraction** : gestion détaillée des classes à exposer
  - Pour aller plus loin : *http://mbaron.developpez.com/eclipse/introplugin*

Services Web : réponses au SOA

- Les Services Web sont basés sur les protocoles et les langages du Web
  - HTTP, XML, TCP/IP pour la couche réseau
  - Ne nécessite pas une configuration réseau particulière
- Les Services Web sont auto-suffisants puisqu'ils contiennent toutes les informations à leurs utilisations
  - Chercher, publier et consommer
  - Annuaire, contrat de fonctionnement et un client pour les consommer
- Les Services Web sont modulaires
  - Une application doit être décomposée en un ensemble de services
  - Utilisation d'une orchestration
- Les Services Web peuvent être définis par des standards
  - OASIS, W3C, WS-I et IETF

Services Web : réponses au SOA



http://www.innoq.com/resources/ws-standards-poster/

Services Web : technologies disponibles

- Deux familles de Services Web se distinguent actuellement
- **Services Web « étendues »**
  - S'appuie sur des standards UDDI / WSDL / SOAP
  - Annuaire de Services Web : UDDI
  - Contrat : WSDL
  - Consommer : SOAP
- **Services Web REST** (Representational State Transfer)
  - Défini par la thèse de Roy Fielding en 2000
  - Utilise directement HTTP au lieu d'utiliser une enveloppe SOAP
  - URI est utilisée pour nommer et identifier une ressource
  - Méthodes HTTP (POST, GET, PUT et DELETE) sont utilisées pour effectuer les opérations de base CRUD

Services Web étendus

Annuaire UDDI

Interrogation de **l'annuaire UDDI** pour rechercher des contrats WSDL suivant des critères spécifiques

Chercher

WSDL

WSDL

Document WSDL est utilisé comme contrat du Service Web

Publier

Messages SOAP est envoyé pour consommer (invoquer) un Service Web

Consommer

Consommateur du Service

Fournisseur du Service

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:hel="http://helloworldwebservice.lisi.ensma.fr/">
        <soapenv:Header/>
        <soapenv:Body>
                <hel:simpleHelloWorld/>
        </soapenv:Body>
</soapenv:Envelope>
```

Services Web étendus

- Pile des standards pour les Services Web étendus

Langages de Processus Métier (BPL) BPEL			Orchestration
Sécurité WS-Security	Fiabilité WS-RM	Transaction WS-Transactions	Qualité de Service
WSDL, UDDI			Découverte & Description
SOAP 1.1 et 1.2			Message
HTTP, SMTP, FTP, BEEP			Transport

Services Web REST

- Exploités pour les Architectures Orientées Données (DOA)
- REST n'est pas un standard, il n'existe pas de spécification W3C définissant une spécification
- REST est un style d'architecture basé sur un mode de compréhension du Web
- REST s'appuie sur des standards du Web :
  - Protocole HTTP
  - URLs
  - Formats de fichiers
  - Sécurisation via SSL

Services Web REST

- Pile des protocoles et langages pour les Services Web REST

Langages de Processus Métier (BPL) BPEL	Orchestration
HTTP Basic, SSL / TLS	Qualité de Service / Sécurité
WADL, ATOM, …	Découverte & Description
MIME Types (Text, JSON, XML, …)	Message
HTTP, FTP, …	Transport

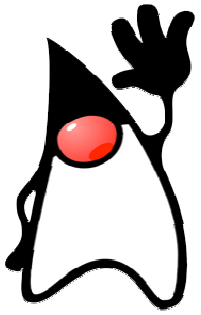
Les fournisseurs de Services Web ?

- Deux types de fournisseurs sont à distinguer
  - Fournisseurs de Services Web « Orientés Web » (public)
  - Fournisseurs de Services Web « Entreprise » (privé)
- Les grands noms du Web sont présents et leurs services sont accessibles
  - Amazon, eBay, Delicious, Facebook, Flickr, Google, Twitter, WheatherBug, Yahoo, Zillow, Zvents

- Oui mais ...
  - Pratiquement tous les fournisseurs de Services Web exploitent l'architecture REST (besoins de performance)
  - Certains (comme Google) ont arrêtés les Services Web étendus
  - eBay propose encore des Services Web étendus

Plateformes de développement

- La grande majorité des plateformes de développements fournissent le support de Services Web (outils et APIs)
  - Plateforme .NET
  - Plateforme Java
  - Plateforme PHP, C++, Python, ...
- Les outils permettent de
  - Manipuler des messages SOAP
  - Manipuler des données au format XML
  - Mapping XML / Classe (*Marshall*, *Unmarshall*)
  - Accéder à la couche HTTP
- Dans ce cours, nous utiliserons la plateforme Java
  - Outillée, gratuite, accessible, légère, respect des standards



Objectifs du cours

- Connaître les normes, les standards et les techniques définissant les Services Web (WSDL, SOAP, UDDI, HTTP)
- Spécifier le contrat d'un Service Web (WSDL, WADL)
- Appeler un Service Web via des messages SOAP, HTTP
- Programmer un Service Web
  - « From Scratch » à partir d'une classe Java
  - A partir de WSDL
- Déployer un Service Web
- Programmer un client d'un Service Web
- Composer des Services Web par orchestration