

Du Web de services au Web des objets

Michaël Mrissa, Lionel Médini
Université Claude Bernard Lyon 1

Laboratoire d'InfoRmatique en Image et Systèmes d'information

LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/Ecole Centrale de Lyon

<http://liris.cnrs.fr/>



Vue d'ensemble

- Contexte et rappel historique
 - Des SI distribués aux services
- Les services Web
 - Définition
 - Retour sur XML
- Architectures orientées services (SOA)
 - Présentation générale
- Services "classiques"
 - Protocoles et langages (SOAP, WSDL, UDDI)
- Services RESTful
 - L'approche REST et les services



Contexte et rappel historique

- SI répartis sur le réseau (Internet)
 - Technologies client/serveur (parfois P2P)
 - Accès à des données distribuées
- Différents moyens d'accès
 - Mobile (Wap - WML)
 - Web (HTTP – HTML)
 - Autres (minitel, news, etc...)
- Plusieurs sources de données
- Plusieurs mises en formes

Contexte et rappel historique

- Notion de B2B
 - Interconnection entre organisations
 - Valeur ajoutée par aggrégation d'applications
- Exemple de l'agence de voyage
 - Transport, logement, activités sur place
- Un produit, plusieurs services
 - Adaptation au client, notion de profil
 - Choix de différents services selon la demande
- Aggrégation et transformation d'information



Contexte et rappel historique

- Notion de service dans les SI distribués
 - Fournisseur de fonctionnalité
- Combinaison entre services
 - Interne: Enterprise Application Integration (EAI)
 - Externe: portails d'entreprises
- Problématiques "réactualisées"
 - Interopérabilité (échange de données)
 - disponibilité, fiabilité, sécurité, QoS, etc.

Contexte et rappel historique

- Tentatives précédentes
 - CORBA => très (trop?) complexe
 - COM, DCOM => dépendant de la plateforme (M\$)
 - RMI => dépendant du langage (Java)
- Problèmes généraux
 - Clients lourds, implémentations complexes
 - Dépendances nombreuses
- Avantages généraux
 - Efficacité de fonctionnement après installation

Contexte et rappel historique

- De quoi a-t-on besoin?
 - Modèle de communication clair
 - Simplicité de mise en œuvre ET d'exploitation
 - Indépendance plateforme ET langage
 - Adoption rapide par les responsables des SI (!)
- La solution? Le Web, i.e. HTTP et XML
 - Plus quelques langages et protocoles (1^{ère} gen)...
 - ...ou pas (2^{de} gen) !



Contexte et rappel historique

- Le Web comme support
 - HTTP est universel (et simple?) + SSL possible
 - Peut être adopté même intra muros
 - Passe les firewalls (port 80)
 - Client standardisé? (cf browser wars)
 - Moins performant qu'un RMI ou CORBA
 - Débit, qualité des communications...
- XML comme langage
 - HTTP transporte (entre autres) de l'hypertexte...
 - Informations structurées en mode texte => XML

Les Services Web

- Objectif des services Web
 - Simplifier la complexité inhérente aux SI
 - Exploiter au mieux les avantages du Web
- Définition du W3C:

„A Web Service is a **software application** identified by a **URI**, whose **interfaces** and **bindings** are capable of being **defined**, **described** and **discovered** by XML artefacts and support **direct interactions** with other software applications using **XML-based messages** via **Internet-based protocols**.



Les Services Web

- Interactions client/serveur
 - Au dessus de TCP/IP
 - Utilisent HTTP (même si tout est permis)
- Passage du Web client au Web machine
 - L'information n'est plus destinée à être affichée
 - Échanges et traitements automatisés
- Passage des techno "ad hoc" au Web
 - CORBA, RMI, DCOM, => HTTP & XML



Retour sur XML

- HTML
 - Sémantique stricte et définie une fois pour toutes
 - Mise en forme de documents Web
- Problèmes
 - Pas extensible
 - Ne décrit pas (ou peu) la sémantique du contenu
 - Mélange mise en forme et contenu (jusqu'à CSS)
- Par rapport à HTML, XML apporte
 - Définition libre des balises => schéma
 - Support de structures complexes
 - Séparation contenu/mise en page



Retour sur XML

- Grands principes de XML
 - Lisible par l'homme et la machine
 - Full-text (non binaire, donc portable)
 - Structure non ambiguë
 - balisage strict parsable + schéma associé
 - Séparation entre document et relations inter-doc
 - Notion d'espace de nommage
 - Séparation structure des données et mise en forme
 - XML => structure, association avec CSS/XSLT



Retour sur XML

- XML Schéma
 - Support de nombreux types de données
 - Possibilité de définir ses propres types de données
 - Ajout de contraintes sur les données
 - Notion d'héritage
 - Support des espaces de nommage
 - Indicateurs d'occurrences des éléments
 - Conception modulaire des schémas
- Défini sur: <http://www.w3.org/XML/Schema>

Architectures orientées services

- Utilisation des services Web
 - XML (description), XML (messages)
 - Web (HTTP) pour le transport
 - Découverte à l'aide d'annuaires (ou registres)
 - Invocation d'une fonction à distance
- Notion de faible couplage
 - Indépendance des plateformes sous-jacentes
 - Indépendance des langages sous-jacents



Architectures orientées services

- Nouveau modèle de développement
 - Service-oriented architectures (SOA)
 - Rationalisation des SI par domaines
 - Décomposition abstraite des fonctionnalités
- Combinaison de services => composition
 - Fonctionnalités avancées
 - Cross-domain apps...
- Interactions dans les SOA
 - 3 acteurs: le fournisseur, le registre, le client
 - Actions: publication, découverte, exécution

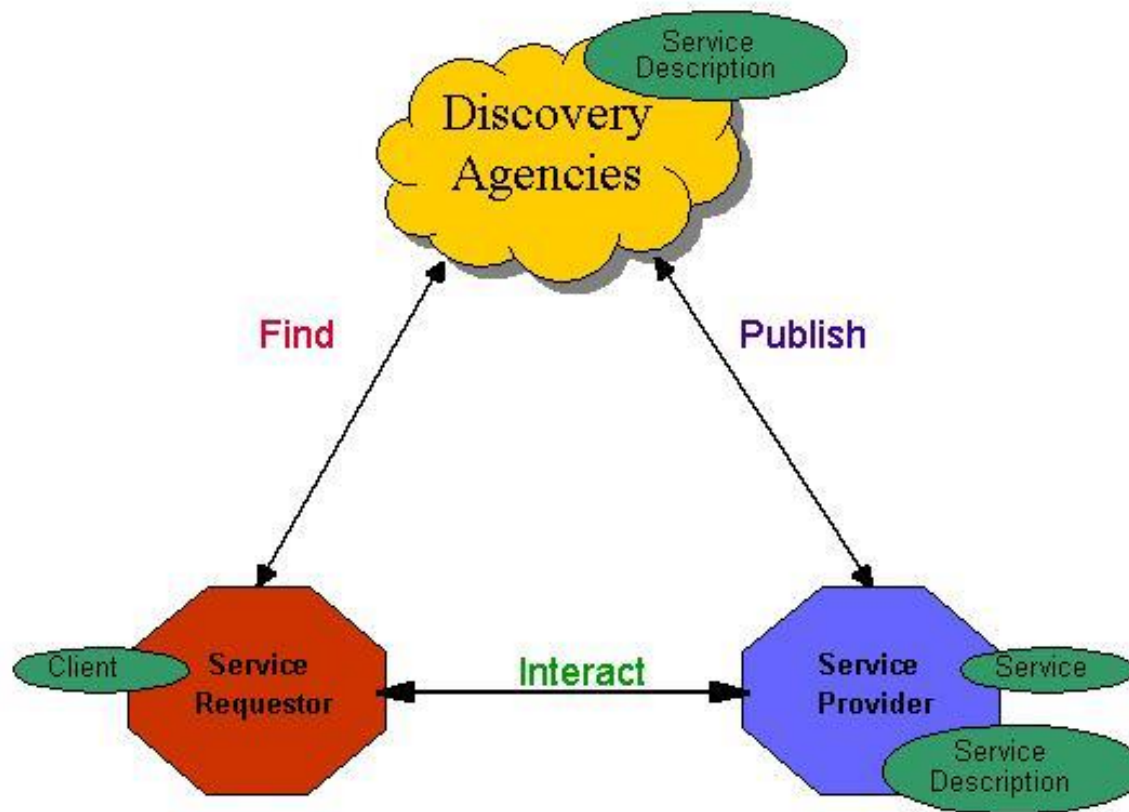
Architectures orientées services

- Le client cherche dans l'annuaire un WS correspondant à ses besoins en termes de
 - Fonctionnalité
 - Qualité de service (QoS): fiabilité, rapidité, etc...
 - Sélection parmi plusieurs WS fournissant la même fonctionnalité
- Plusieurs cas d'utilisation
 - Entre applications distribuées
 - Entre une application et un client Web



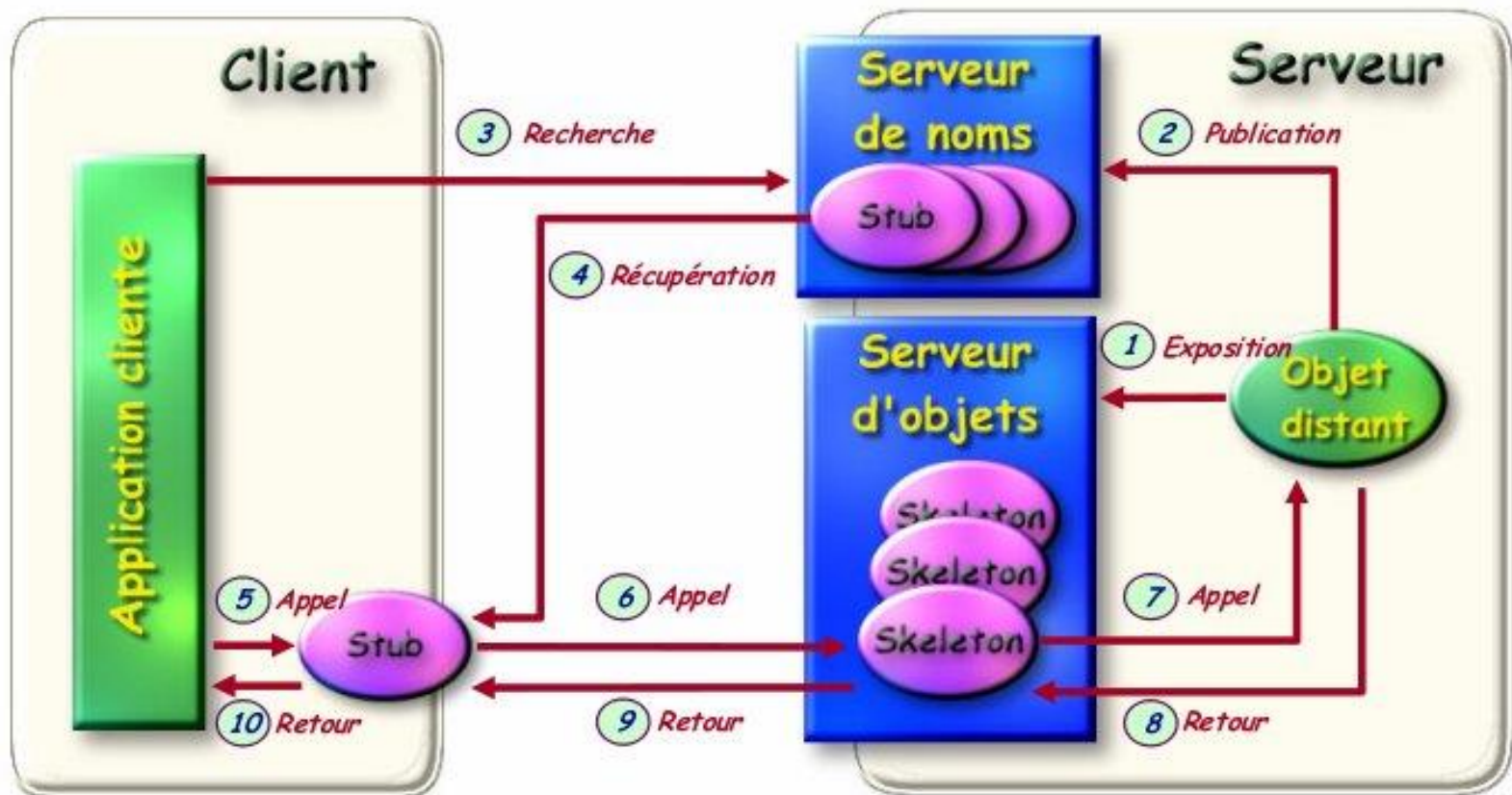
Architectures orientées services

Service Oriented Architecture



Architectures orientées services

□ Quoi de nouveau ?

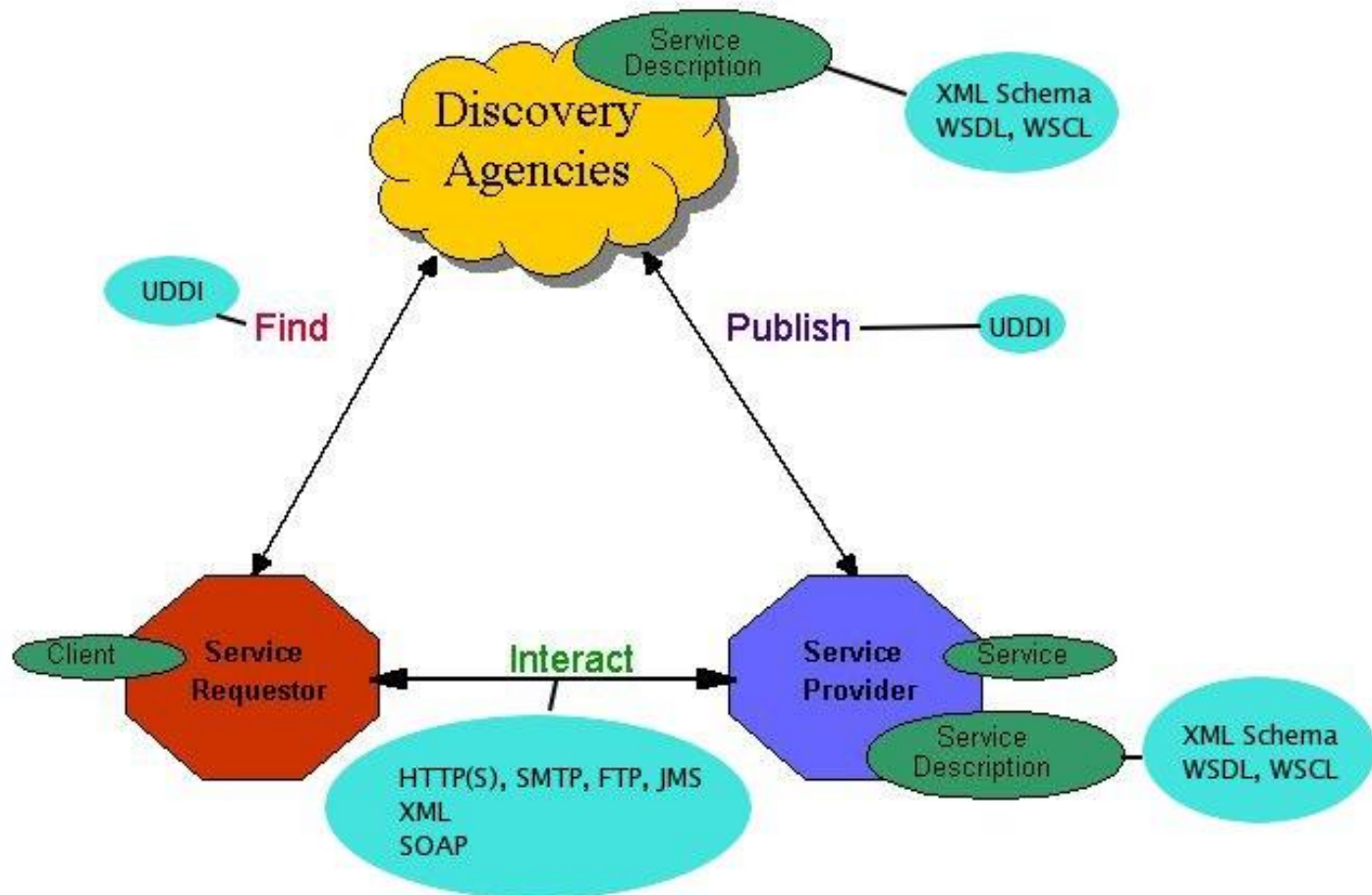


Architectures orientées services

- Alors qu'apportent les services Web?
 - Les technologies utilisées
 - SOAP (Simple Object Access **P**rotocol)
 - Remplace IIOP (CORBA) et RMI-IIOP (RMI)
 - WSDL (Web Service Description **L**anguage)
 - Remplace IDL (CORBA) et interface Java (RMI)
 - UDDI (Universal Description, Discovery and Integration)
 - Remplace CosNaming (CORBA) et JNDI (RMI)
 - L'indépendance des langages et plateformes
 - Interopérabilité, flexibilité, la notion de composant
 - La rationalisation des SI
 - Le soutien des grands acteurs du Web...

Architectures orientées services

Service Oriented Architecture



Architectures orientées services

- Le protocole SOAP
 - Assure les appels de procédure
 - Au dessus d'un protocole de transport (svt HTTP)
 - Simple enveloppe autour des données à transmettre via HTTP

Architectures orientées services

- SOAP Côté client
 - Ouverture d'une connexion HTTP
 - Requête SOAP: document XML décrivant
 - la méthode à invoquer sur la machine distante
 - les paramètres de la méthode
- SOAP Côté Serveur
 - Récupère la requête
 - Exécution de la méthode avec les paramètres
 - Renvoie une réponse SOAP (document XML) au client

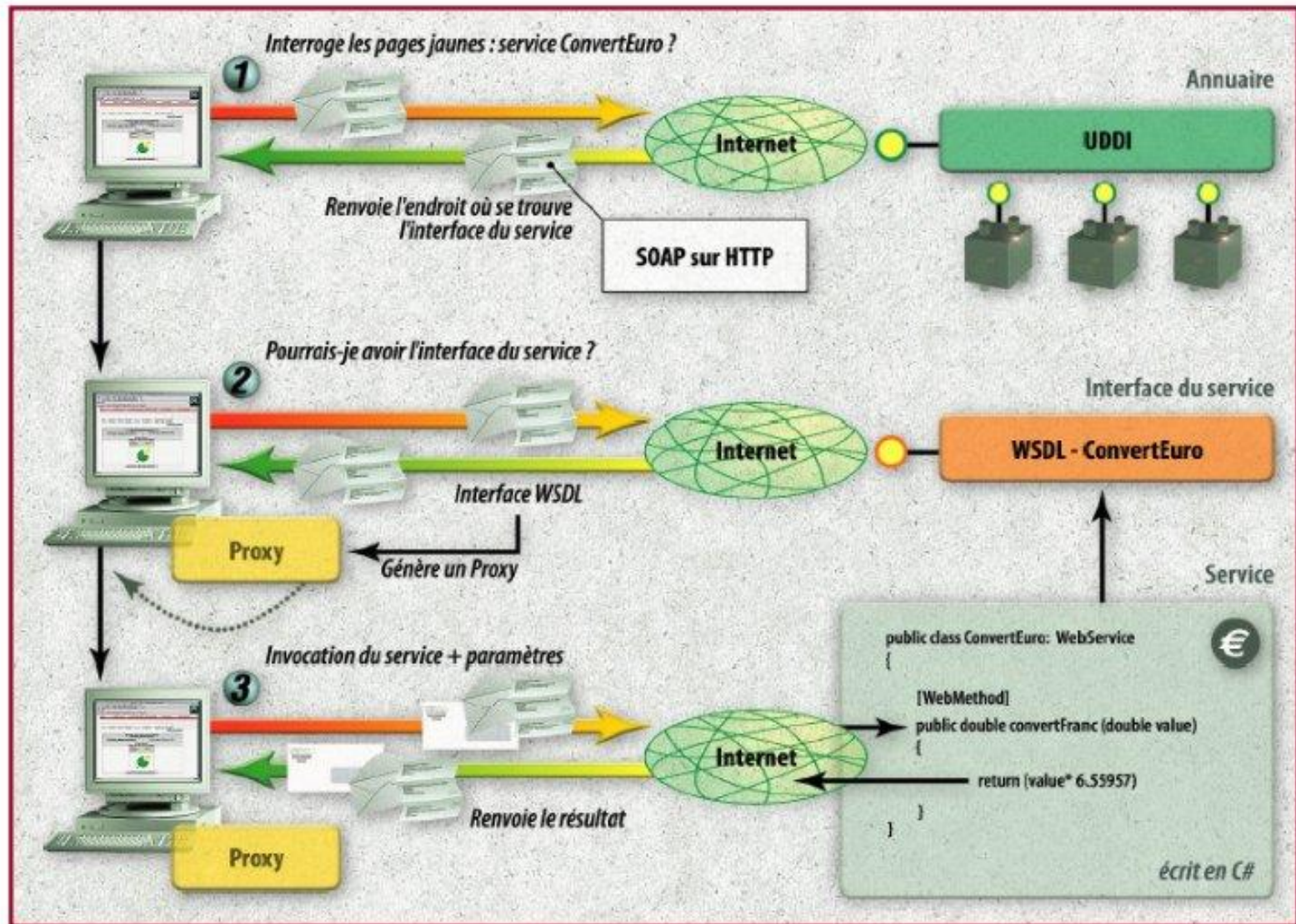


Architectures orientées services

- Le langage WSDL
 - Souvent généré par les outils de développement
 - Décrit l'interface du service au format XML
 - les méthodes, les paramètres et valeurs retournées, les protocoles de transport possibles, et la localisation du service
 - De manière abstraite et concrète
 - Indépendance des 2 parties
- On reste indépendant des plateformes et langages d'implémentation des services
 - => Faible couplage



Architectures orientées services



Le protocole SOAP

- Transparence complète côté serveur
 - BD Oracle ?
 - Autre service Web ?
 - C, C++, Java ?
 - UNIX, Windows ?
 - => boîte noire => faible couplage
- Et de même côté client
- Que voit-on du serveur?
 - On ne voit que les ports ouverts
 - Nécessite un serveur de type Apache Tomcat



Le protocole SOAP

- Avantages de SOAP
 - Séparation des traitements de données
 - Pas besoin de "stub" et "skeleton" comme avec CORBA et RMI
 - Passage de firewall dans HTTP
- Inconvénients
 - Verbeux (bande passante)
 - Pas si "simple"



Le protocole SOAP

- Histoire jusqu'a SOAP
 - 80's: DCOM et CORBA
 - Couplage fort, très orienté objet
 - 99': XML-RPC
 - Messages XML, envoi de formulaire (HTTP/POST), pas extensible, types de données limités
- SOAP
 - Standardisé par le W3C
 - Protocol-independent
 - Extensible, et basé sur XML



Le protocole SOAP

- 2 styles de communication
 - RPC: Appels de procédures distants
 - Paramètres proches des types des langages de programmation
 - Degré élevé d'automatisation
 - DOC: Echanges de messages conformes à des schémas arbitraires (Ex: Demande d'achat).
 - Plus de flexibilité au niveau des datatypes
 - La fonction manque dans le message SOAP
 - Encouragé par .Net
- Voir: <http://www.eherenow.com/soapfight.htm>



Le protocole SOAP

- 2 types de communication
 - Synchrone (par dessus HTTP)
 - Appels bloquants, pb de timeout...
 - Asynchrones (SMTP, JMS...)
 - Appels non bloquants
 - garantie de service (messages recus une et une seule fois)

Le langage WSDL

- WSDL cache le détail de l'implémentation du service, permet une utilisation indépendante
 - de la plate-forme et du langage utilisé
- Regroupe les informations nécessaires pour interagir avec le service :
 - les méthodes, les paramètres et valeurs retournées, le protocole de transport utilisé, la localisation du service
- Les documents WSDL sont générés par les outils de développement et favorisent une intégration rapide du service



Le langage WSDL

- Un document WSDL contient 6 parties
 - Les quatre premières parties décrivent des informations abstraites indépendantes d'un contexte de mise en œuvre. On y trouve :
 - les types de données envoyées et reçues
 - les opérations utilisables
 - le protocole qui sera utilisé,
 - Les deux dernières parties décrivent des informations liées à un usage contextuel du service. On y trouve :
 - Le point d'accès du service, les protocoles utilisés et le lien entre les 2

L'annuaire UDDI

- UDDI (Universal Description, Discovery and Integration), standard né sous l'initiation de Microsoft, IBM, Sun, Oracle, Compaq, HP, Intel, SAP, etc.
- Standard pour faciliter la collaboration entre partenaires dans le cadre d'échanges commerciaux
- Le cœur de UDDI est un annuaire qui contient des informations techniques et administratives sur les entreprises et les services qu'elles publient



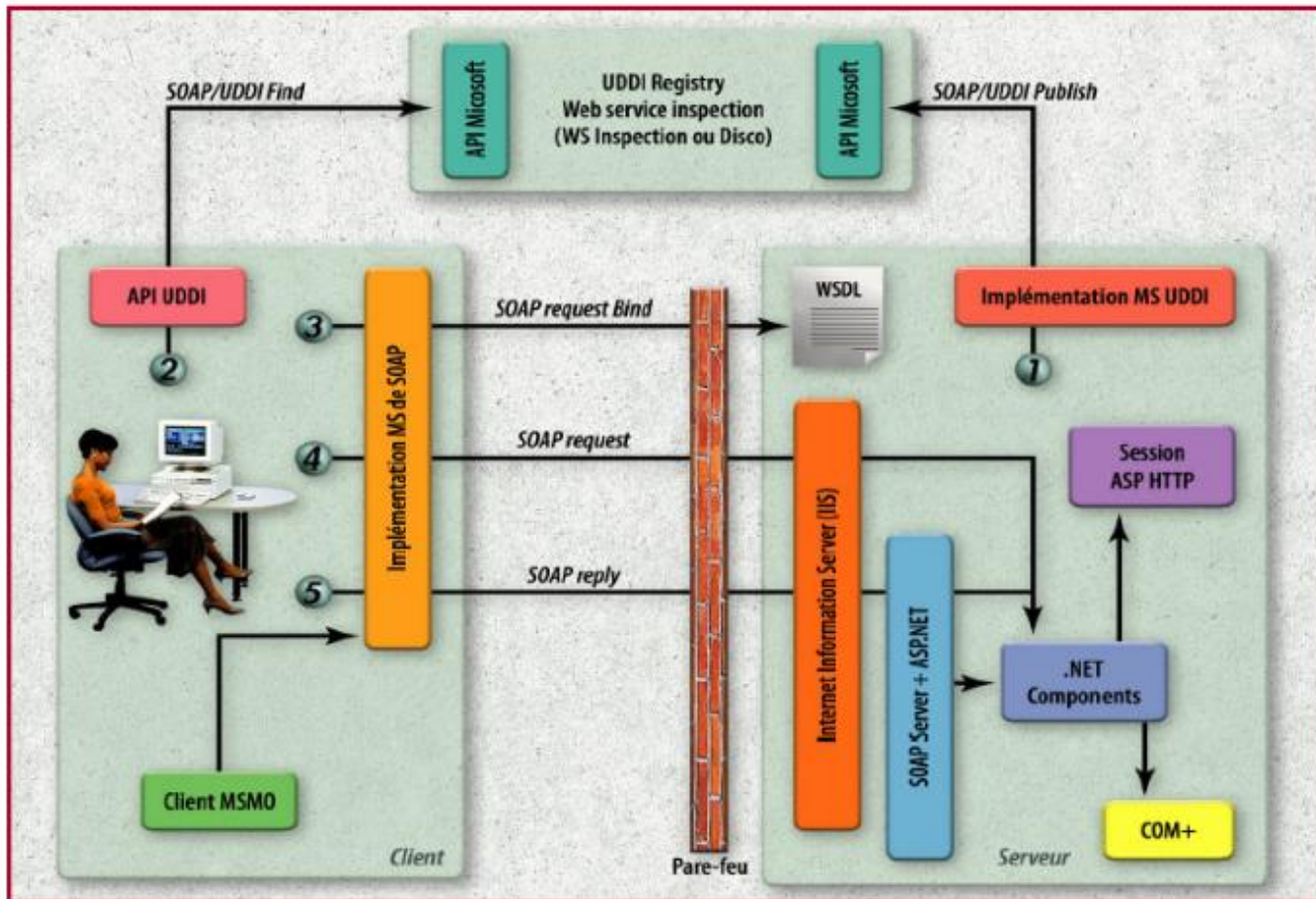
L'annuaire UDDI

- L'annuaire UDDI permet de :
 - Publier, découvrir des informations sur une entreprise et ses services
- L'inscription sur UDDI permet à une entreprise de se présenter ainsi que ses services
- L'adoption de UDDI facilite le développement des échanges de type « B2B »
 - L'enregistrement des services dans un annuaire s'effectue après d'un opérateur (Microsoft ou IBM actuellement) à travers son site mais on peut créer ses propres registres UDDI (UDDI4J, jUDDI)



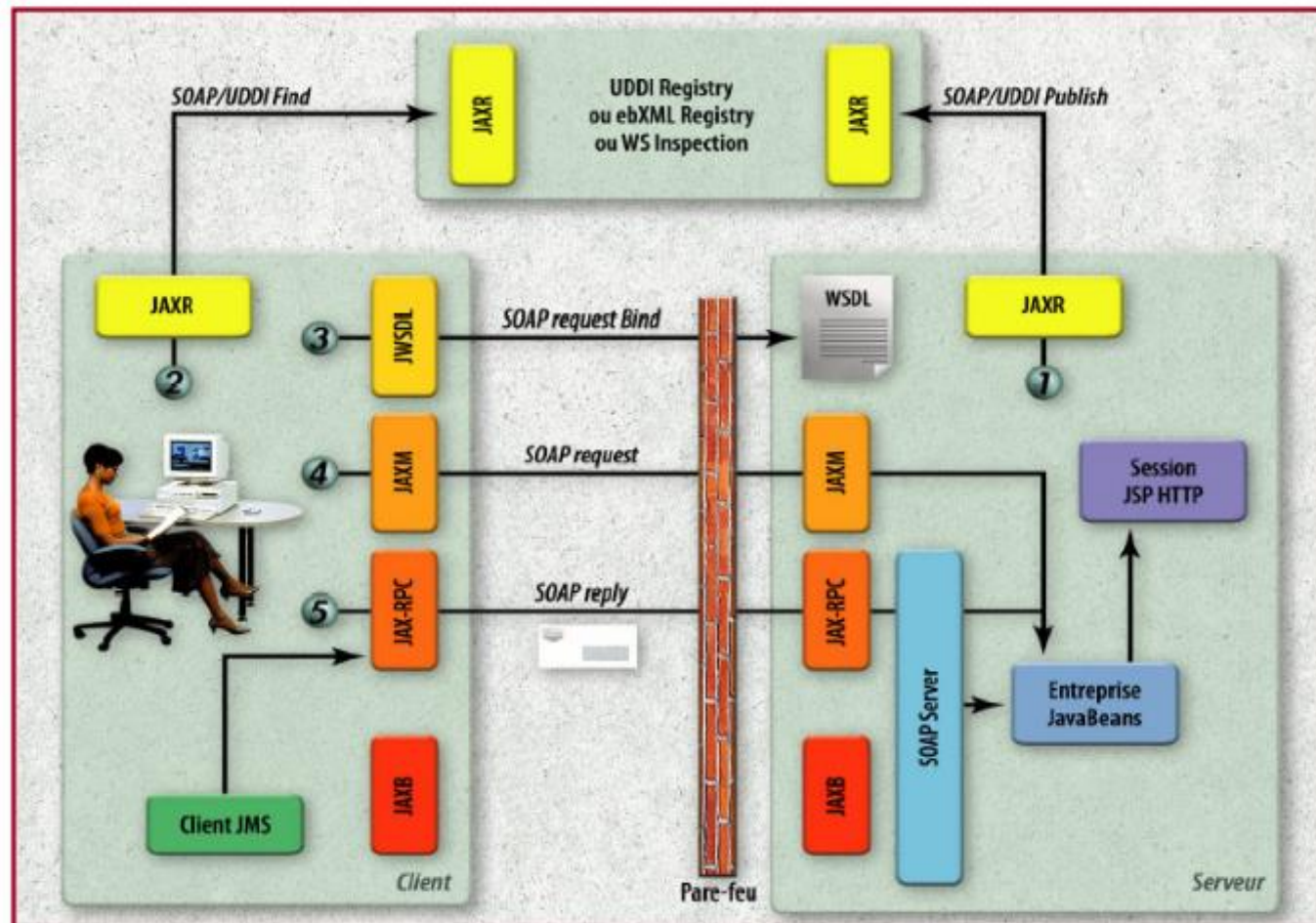
Implémentations

- Architecture .NET



Implémentations

- Architecture J2EE



Conclusion

- Bilan
 - Beaucoup de technologies
 - Poussées par les grosses firmes
 - Trop de standards tuent la standardisation
 - Trop de variantes sur les implémentations
 - Encore des problèmes entre J2EE et .NET
- Questions ouvertes...pour la suite
 - Et les services RESTful?
 - Quid de la sémantique?



REpresentational State Transfer

■ Contexte historique

- REST : Une alternative à SOAP
 - Proposée dans la thèse de Fielding en 2000
 - En 2006, Google change son API pour REST
 - De nombreux fournisseurs de services suivent

■ Aujourd'hui

- Sur programmableweb.com
 - 6279 APIs REST
 - 2052 APIs SOAP

■ Pourquoi ce changement ?



SOAP : inconvénients

≡ Problèmes d'implémentation

- Diverses implémentations incomplètes ou bancales
- Sérialisations des types XML complexes incompatibles

≡ Profusion d'extensions

- La fameuse pile WS-* (policy, security, trust)
- Connue pour sa mauvaise lisibilité

≡ Avantage initial de SOAP

- => indépendance du protocole de transport sous-jacent
- Constat à l'utilisation => HTTP standard de facto
- => redondance, utilisation contre nature du HTTP



REST: définition et objectifs

☰ Définition

- **Style architectural** : ensemble de contraintes pour qu'un système vérifie un certain nombre de propriétés
- L'idée étant de conserver les « bonnes » propriétés du Web et d'en éviter les erreurs

☰ Objectifs

- **Passage à l'échelle (performance, disponibilité)**
 - => fonctionnement décentralisé
- **Interface la plus générique possible**
- **faible couplage (une notion venant des services)**
 - => verbes HTTP

☰ Mise en valeur de HTTP par son utilisation

REST: définition et objectifs

Contraintes techniques à respecter

- Connexions sans état (stateless)
 - L'état n'est plus sauvegardé sur le serveurs
 - => allègement du serveur, passage à l'échelle favorisé
- Communications Client/Serveur
 - Problématique pour les notifications au client
- Interface uniforme
 - WSDL ??
- Principle of partial understanding (Michael Hausenblas)

Cf thèse Roy Fielding...

REST : principes généraux

Ressource

- Une ressource est une entité abstraite.
 - → Ce n'est pas un fichier.
- Une ressource est identifiée par un URI (Uniform Resource Identifier).
 - → Un URI/URL n'identifie pas un fichier !

Représentation

- Une ressource a une ou plusieurs représentation(s).
 - → négociation de contenu
- Ces représentations peuvent varier dans le temps.
- Les ressources sont toujours manipulées via leurs représentations.



Les Services RESTful

☰ Adaptation des services « classiques »

- Censés respecter les contraintes REST
- Lesquelles ne sont souvent pas toutes respectées

☰ Navigation via les liens hypertextes

- L'état d'un échange long est transmis avec la connexion
- Nombreuses implémentations (repose sur HTTP)
 - JAX-RS, PHP, .net, Python...

☰ Conclusion

- Nombreux avantages par rapport à SOAP
- Bonne adoption par la communauté
- Futur prometteur
- Ne résout pas les problèmes « classiques »
 - Interopérabilité, sémantique...



Le Web sémantique

 Tout est ici :

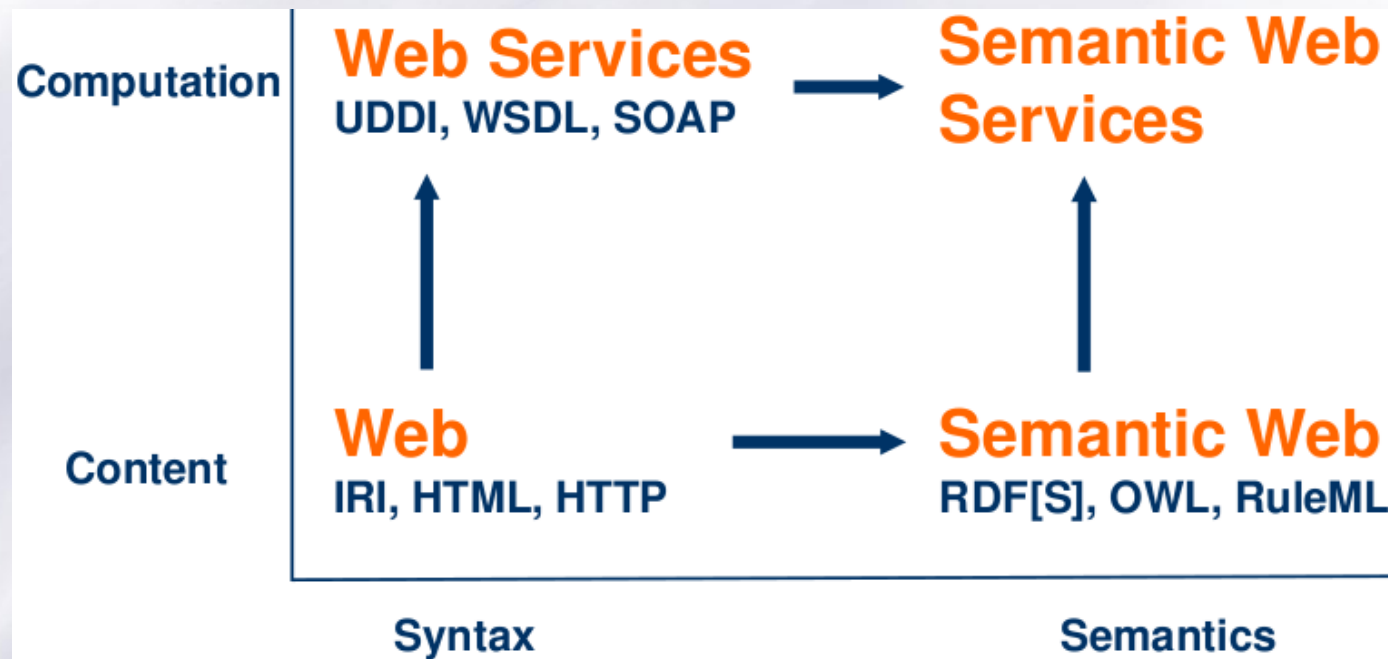
<http://liris.cnrs.fr/~pchampin/2012/semweb/rdf.html>

Services Web sémantiques

- D'un côté le Web sémantique
 - Plusieurs langages (RDF(S) – OWL)
 - Ajout d'une sémantique (machine-)explicite
 - Annotation sur le Web (GRDDL, μ formats, RDFa)
- De l'autre côté les services Web
 - Notion de SOA: passage au Web orienté service
 - Les technologies (SOAP/WSDL/UDDI)
 - Basé sur XML/HTTP
- Combiner les deux ?



Services Web sémantiques



Source: www.icec06.net/WorkshopsAndTutorials/SOATutorial/ICEC06-Tutorial-Semantic-Web-Services.pdf

Services Web sémantiques

- Limitations des services Web
 - Tout est manuel (!)
 - Souvent l'appel aux services est figé dans le code
 - La sémantique est là, mais pas pour les machines
- Objectifs de la sémantique ajoutée:
 - Automatiser et rendre dynamiques les tâches liées aux services Web
 - Découverte, sélection, composition, négociation, invocation, monitoring, reprise sur erreur, etc...



Services Web sémantiques

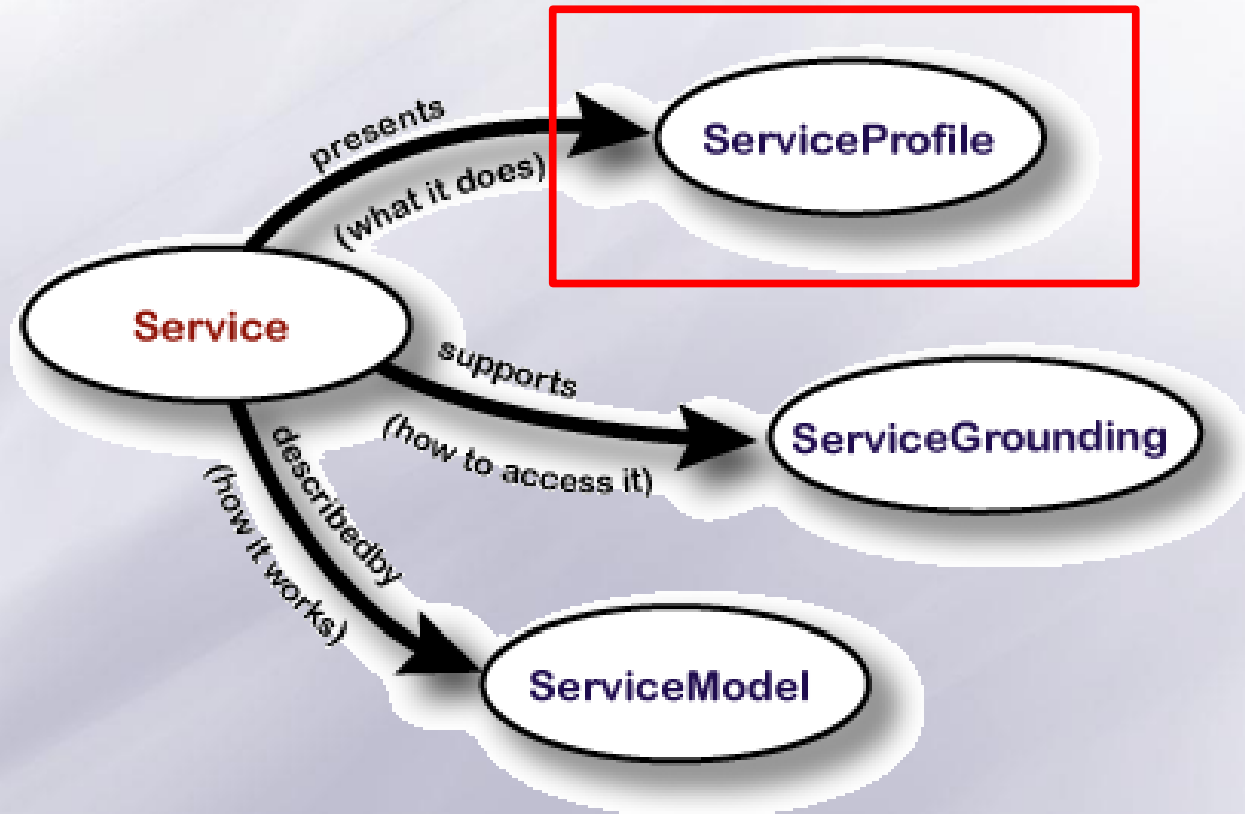
- Annotation des descriptions de services
 - Ajout de sémantique interprétable par les machines
 - Références à des ontologies (OWL)
- 2 approches
 - Bottom-up: Extension de WSDL (SAWSDL, hREST...)
 - Top-down: intégration de WSDL (OWL-S, WSMO...)
 - avantages/inconvénients...
- W3C porteur des projets



Services Web sémantiques

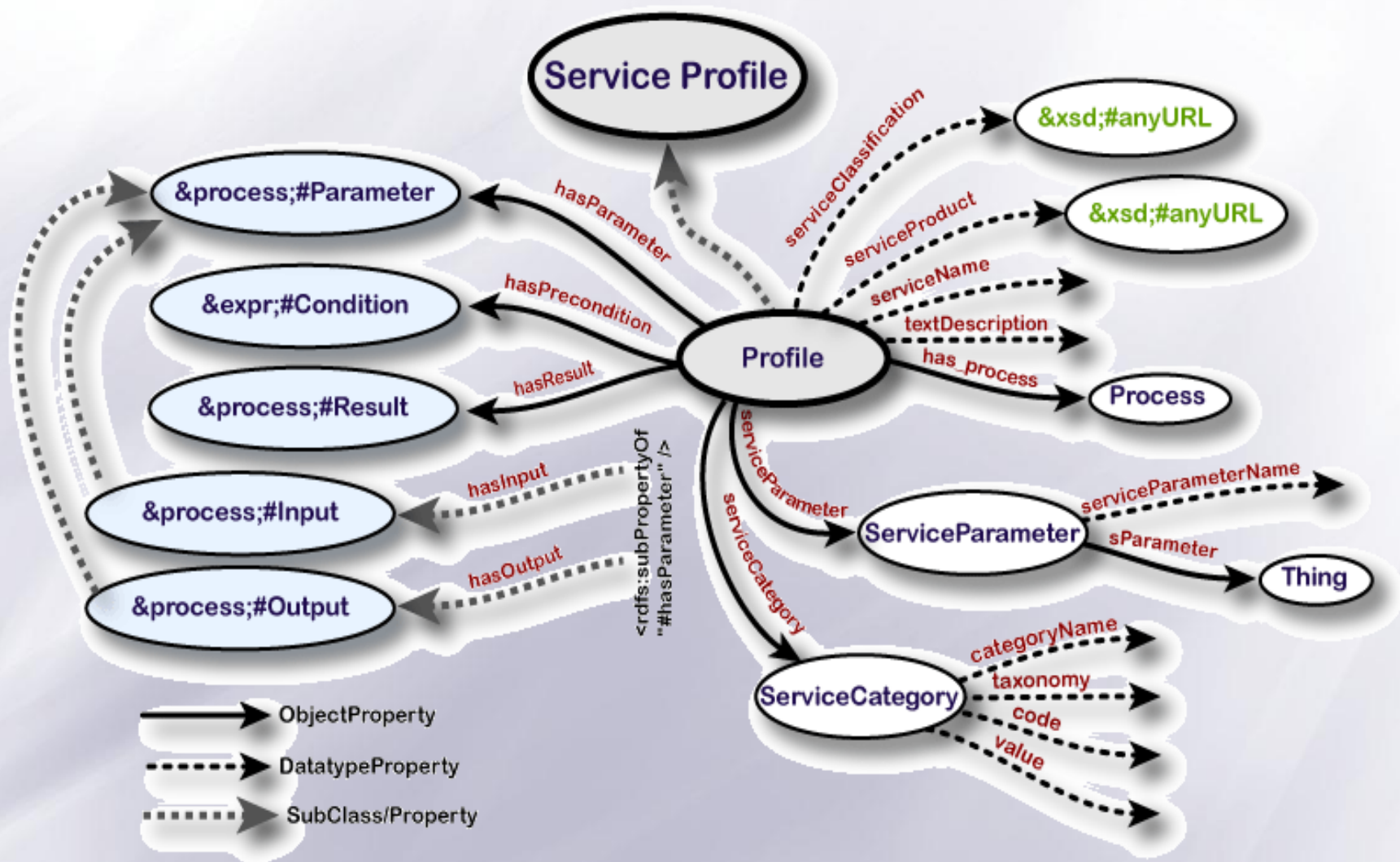
- OWL for Services (OWL-S)
 - Description formelle des services
 - Capacité, interface, etc...
 - Intègre le langage WSDL
- Objectifs
 - Raisonnement automatique sur les descriptions
 - Composition automatique
 - OWL-S est exprimé via OWL
 - Il définit comment décrire un service par un ensemble de structures



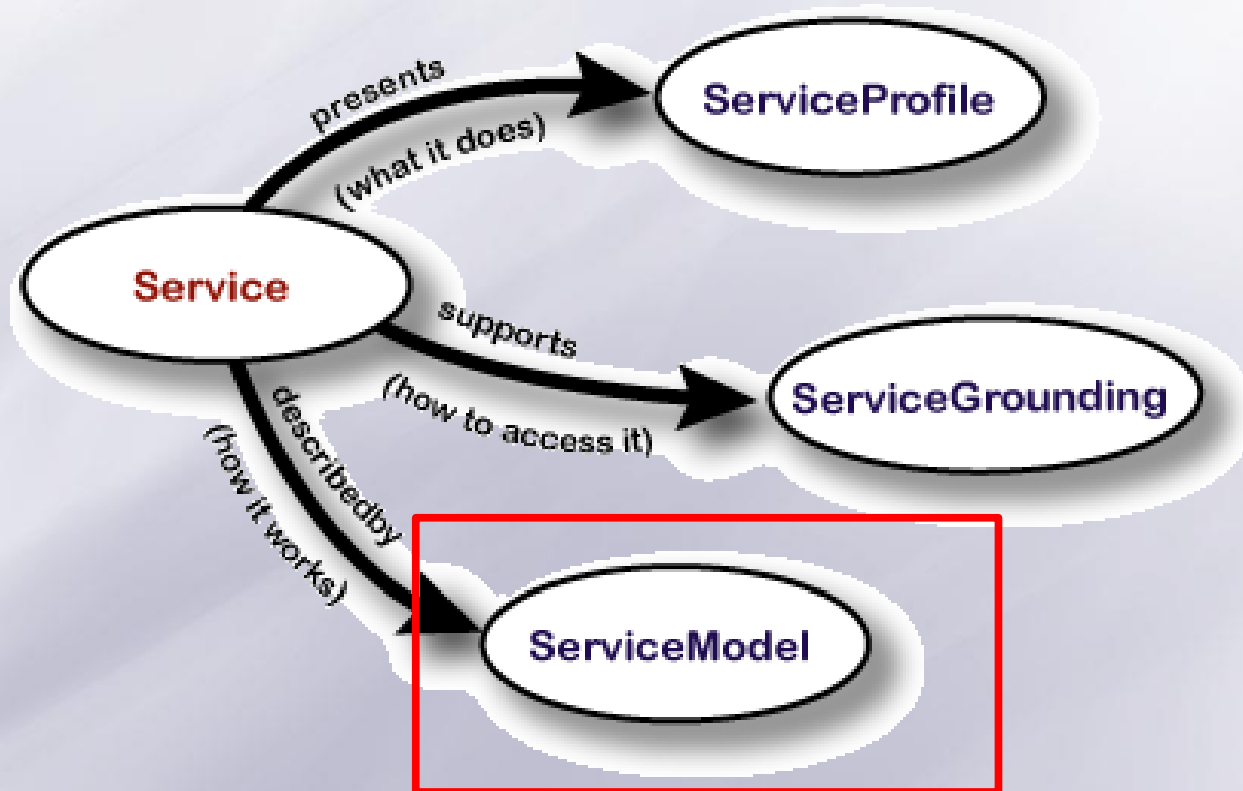


Source: <http://www.w3.org/Submission/OWL-S/>

- Service profile
 - Répond à: "Que fait le service?"
 - Description de haut niveau, utile pour les registres
 - Permet de sélectionner les services selon les fonctionnalités sont définies dans les ontologies
 - Un service peut posséder plusieurs profiles
- Utilisation
 - Sélection: identifier la fonctionnalité via ses attributs
 - Planning: I/O, précondi., effets, notion de process



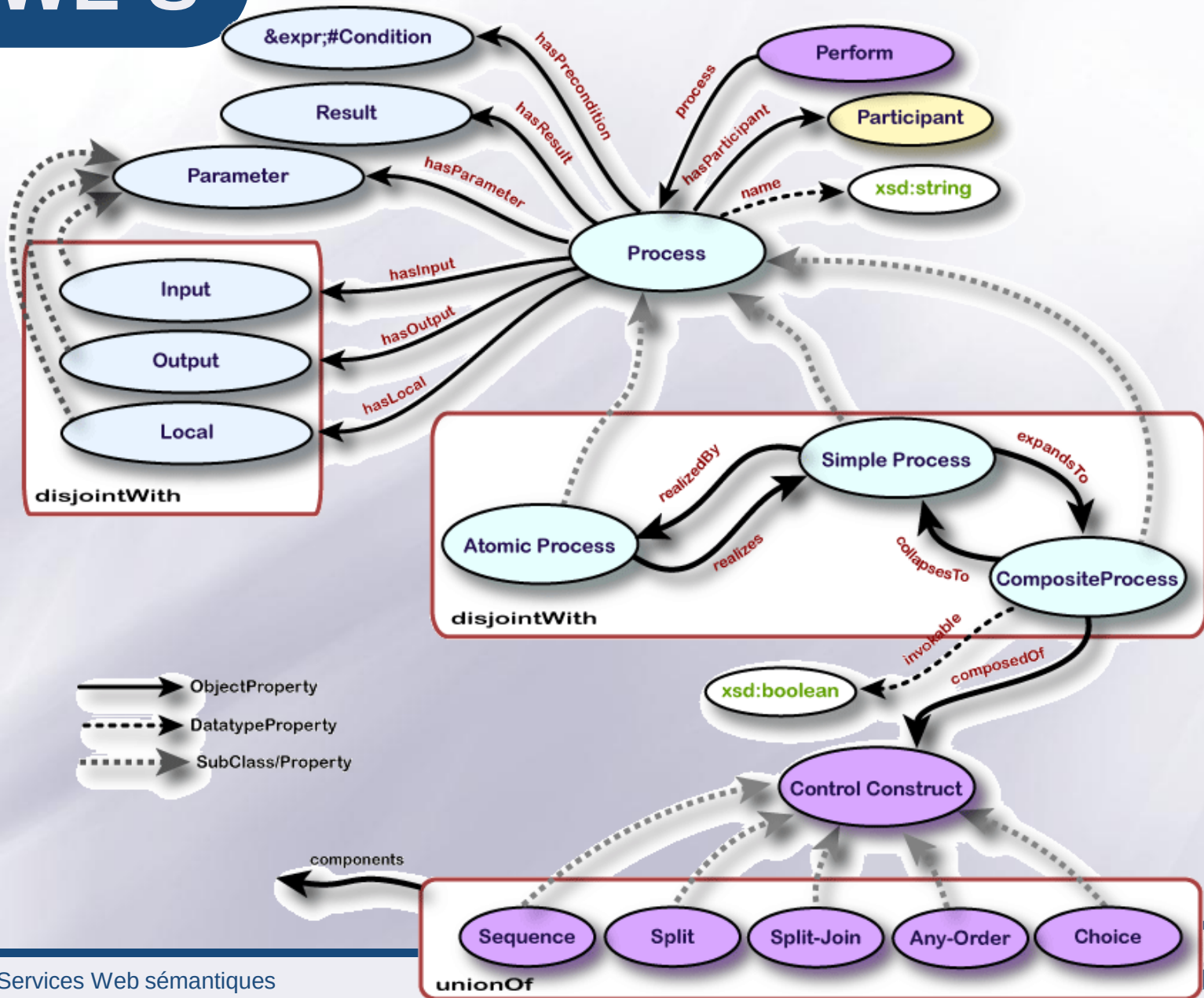
Source: <http://www.w3.org/Submission/OWL-S/>

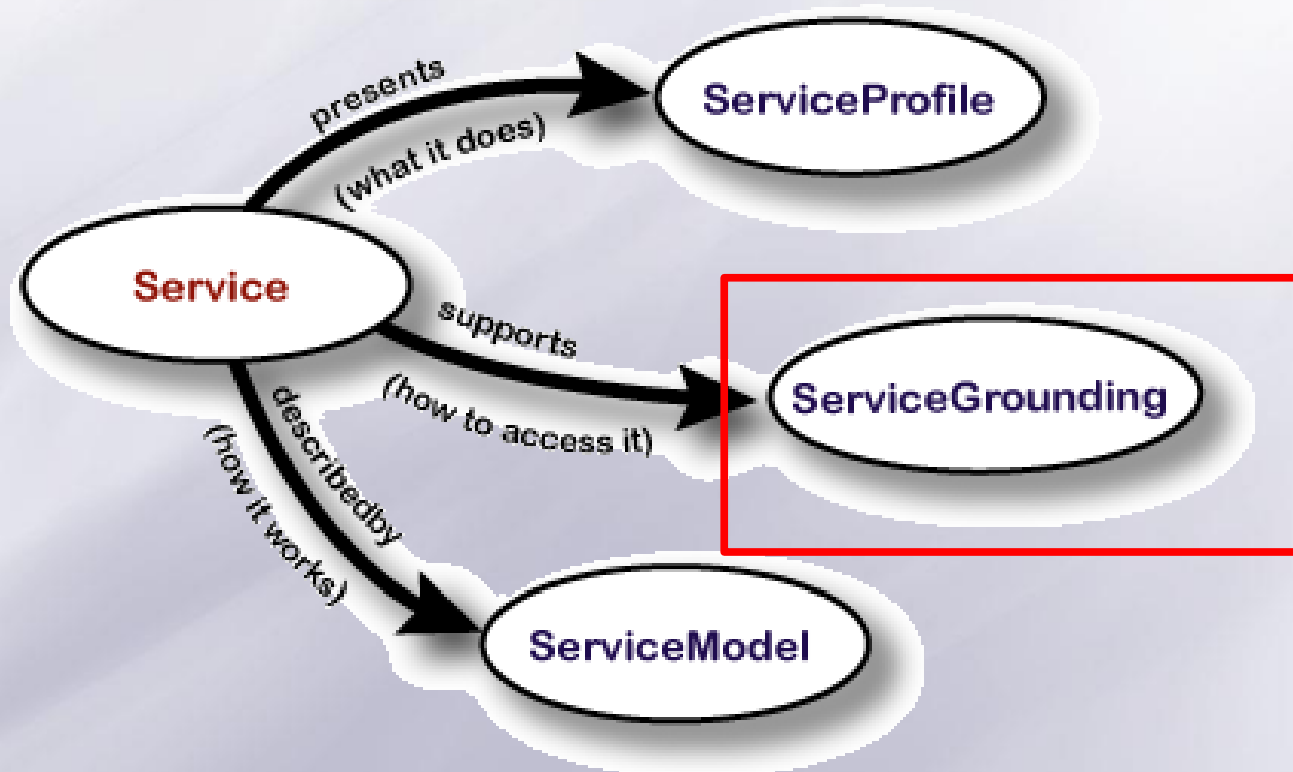


Source: <http://www.w3.org/Submission/OWL-S/>

- Service model
 - Répond à: "Comment fonctionne le service?"
 - Décrit les interactions du service
 - Construit autour du "process" + IOPE
 - Atomique ou composé. Si composé:
 - opérateurs de contrôle et de flux de données
- Utilisation
 - Invocation, planning/composition, interopération, monitoring

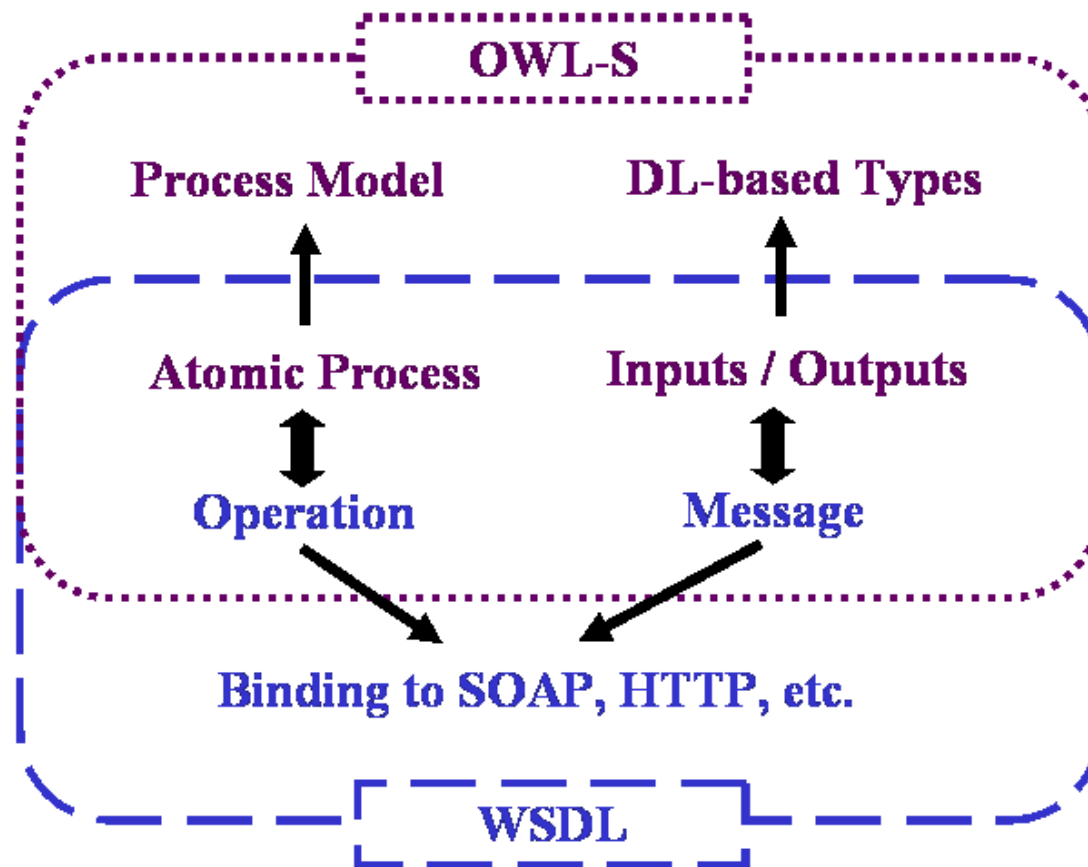
OWL-S





Source: <http://www.w3.org/Submission/OWL-S/>

- Service Grounding
 - Répond à: "Comment accéder au service?"
 - Spécifique à l'implémentation
 - Formats des messages, protocoles de transport, encodage des données
- Utilisation de la description OWL-S
 - Le profile est pour la découverte
 - On peut invoquer un service avec son "Model" et son "Grounding", construits autour de WSDL



Source: <http://www.w3.org/Submission/OWL-S/>

Annotations de services

De nombreuses contributions

- WSDL-S => SA-WSDL, SA-REST
- RDFa, hRESTS (μ -format), microWSMO
- Plus récemment Hydra (<http://www.hydra-cg.com/spec/latest/core>)

Sémantique des services explicite

- Facilite les interactions
- La découverte, les échanges de données
- Ne résout pas certains problèmes
- ➔ Sémantique des données peu détaillée
- Intégration dans le Web sémantique

Les linked services

Linked services

- Des données liées décrivant des services
- Les entrées sorties et fonctionnalités des services
 - décrits en RDF(S)
- Un linked service reçoit et produit du RDF
- Ne pas confondre avec un data service

Avantages

- Meilleure intégration dans le Web sémantique

Inconvénient

- Apprentissage pour l'utilisation de RDF

WoT : Web of Things

≡ Le « Web des choses »

- Souvent appelé le Web des objets

≡ Comment le définir ?

- Une surcouche de l'IoT
- Repose sur les technologies Web pour les interactions
- TCP => HTTP et IP => URI
- Fournit une sous-couche applicative générique
 - Facilite le développement

≡ Intégration « naturelle » avec les services

- Plutôt REST que SOAP
- Un objet = une ou plusieurs ressources
- Accessible par des URIs, propose des services

Domaines de recherche

Le WoT intéresse de nombreux domaines

- Embarqué : intégrer de l'informatique dans des dispositifs souvent limités en ressources
- Web : tirer partie des nouvelles spécifications du W3C (device APIs, WebSockets, Web workers...)
- Services : rationaliser les SI et faciliter les échanges des données/fonctionnalités distantes
- Web sémantique : description explicite et interconnexion des connaissances accessibles par les machines
- SMA : interactions entre des nombreux objets pouvant être vus comme un système multi-agents
- Cloud computing : les objets sont des ressources de calcul, et/ou ont des besoins en ressources