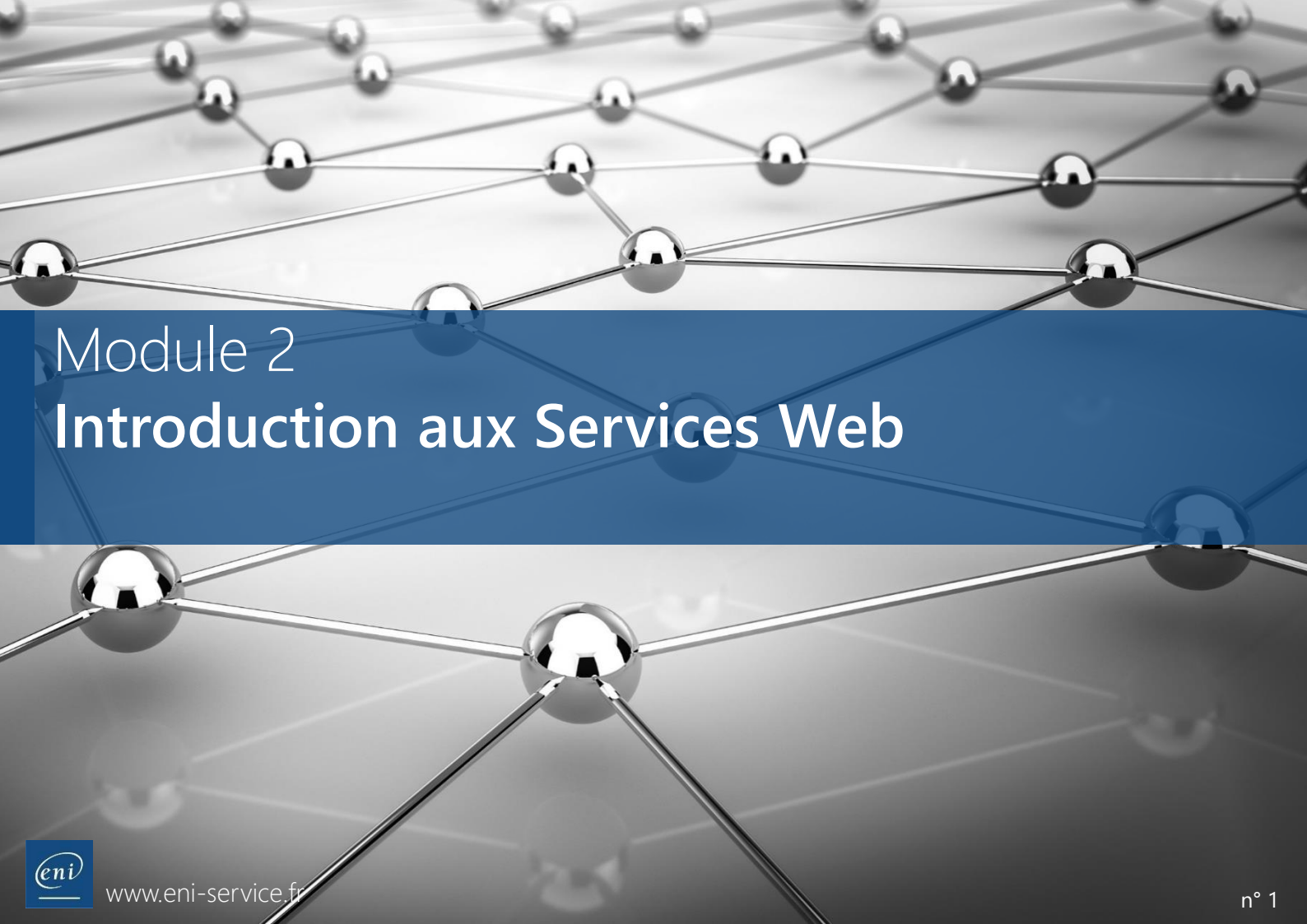


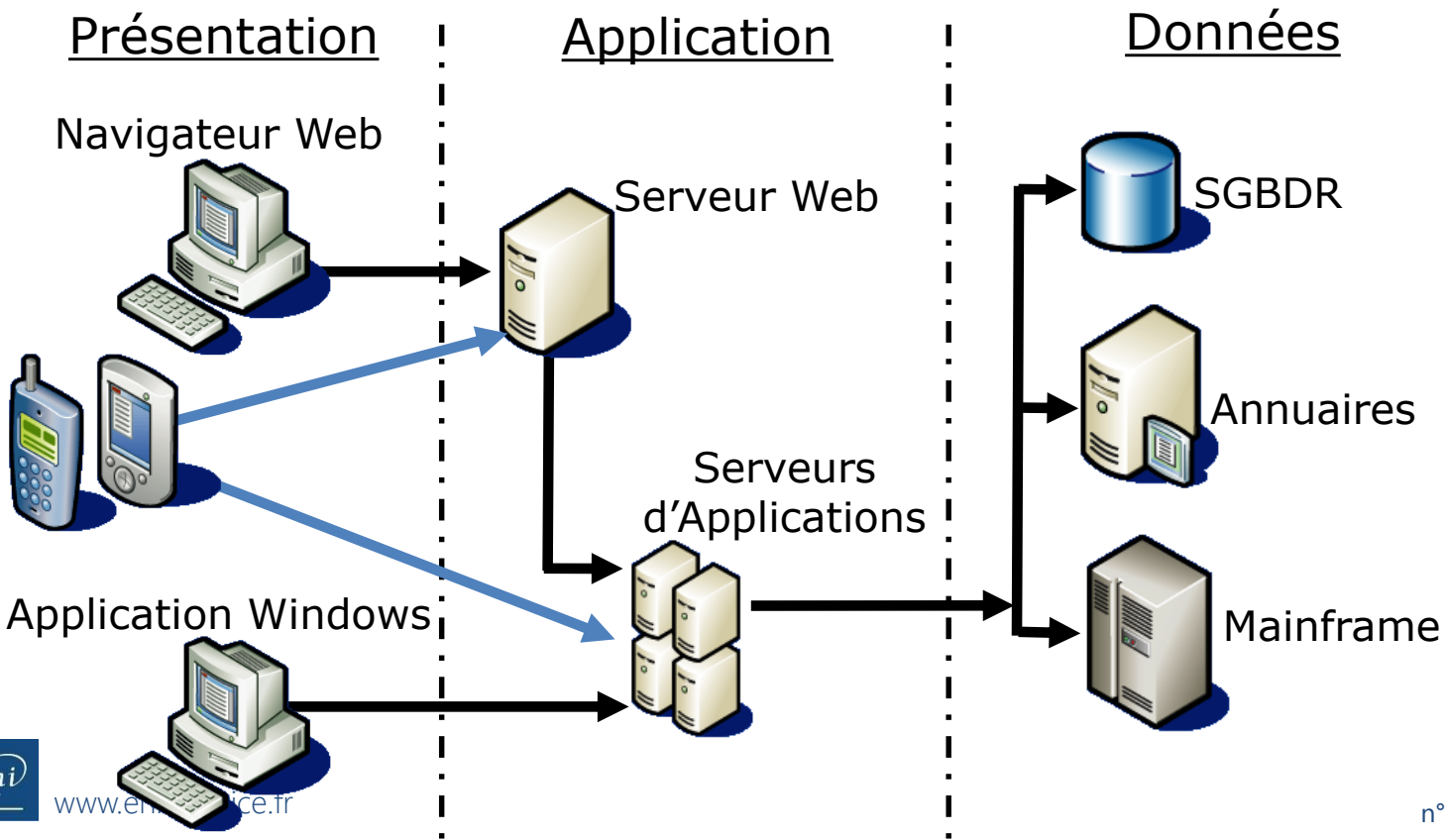
Module 2

Introduction aux Services Web

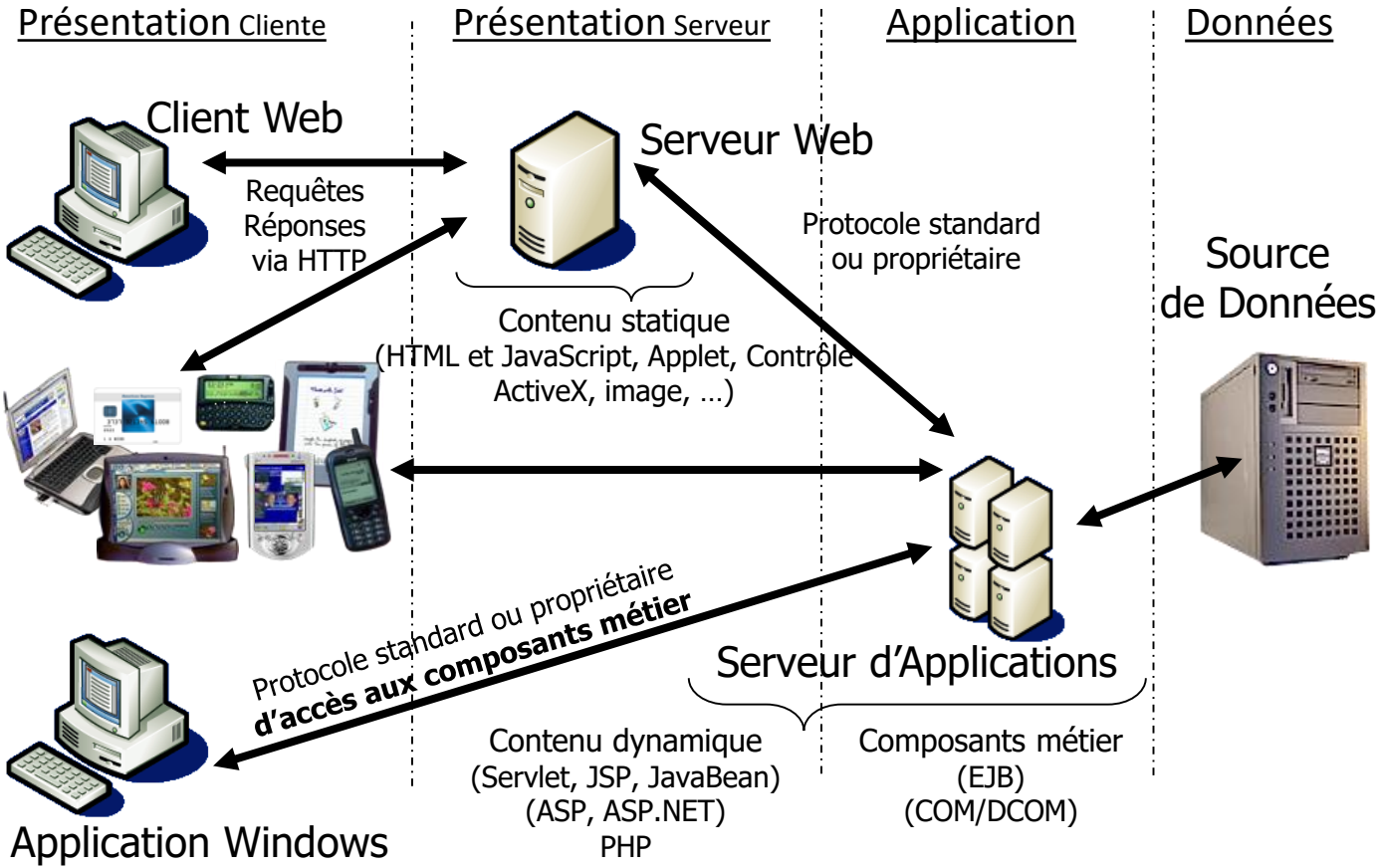
Contenu du module

- Rappels sur les architectures n-tiers et les composants distribués
- Inconvénients et limites du développement à base de composants distribués
- Principes et objectifs des Services Web
  - Avantages et limitations
- Les technologies des Services Web
  - SOAP
  - WSDL
  - UDDI
- Vers une architecture orientée service (SOA)

Architecture de développement 3/3 – Version Simple

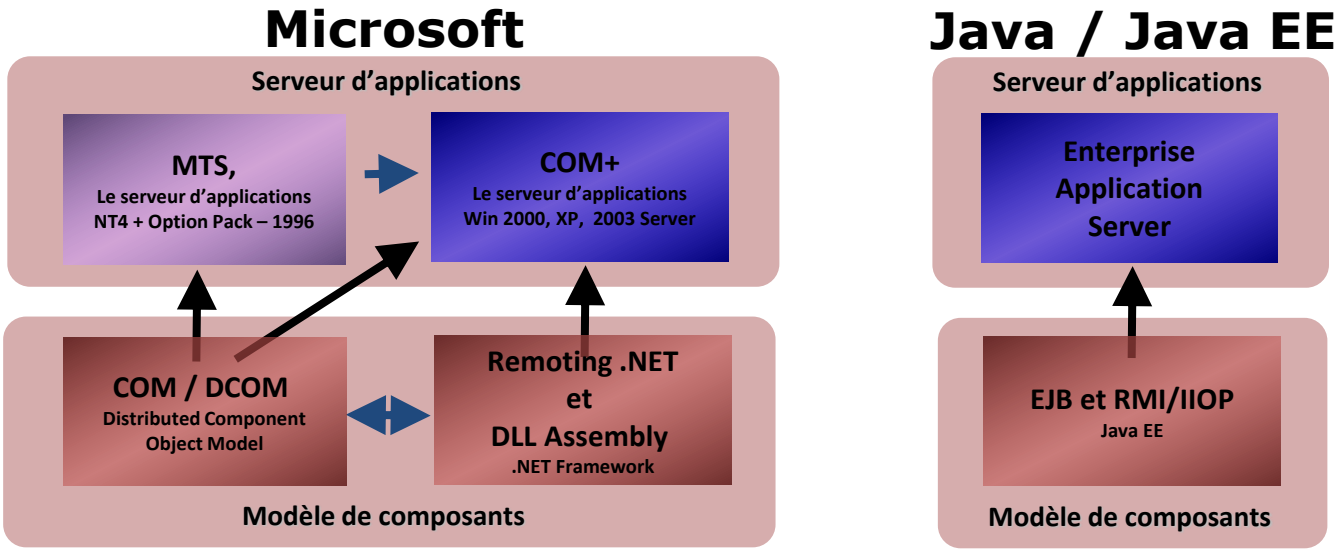


Architecture de développement 3/3 – Version étendue



Les composants distribués

- Les serveurs d'applications hébergent des applications sous la forme de composants distribués sur le réseau



Développement avec composants distribués

- Avantages

- Réutilisation pour des applications clientes différentes
- Maintenance et évolution facilitées des applications
- Allégement de la charge de travail côté client et délégation côté serveur
  - Permet d'avoir des clients légers (interface graphique et gestion des événements)
- Placement des composants au plus près des données qu'ils ont à manipuler

- Inconvénients

- Utilisation limitée aux réseaux locaux car ne passent pas les pares feux
  - Non adapté à une utilisation au travers d'Internet
- Incompatibilité entre les différentes architectures d'implémentation
  - Langages d'implémentation différents
  - Plates-formes sous-jacentes différentes
  - Systèmes d'exploitation différents
  - Quelques passerelles existent, mais elles sont très limitées

Evolution vers les Services Web – Enjeux du S.I.

- Maîtriser la complexité du Système d'Information
  - Homogénéiser l'infrastructure technique du SI
  - Gérer la diversités des technologies

- Ouvrir l'entreprise
  - Intégrer les partenaires, les clients, les employés
  - Automatiser les processus

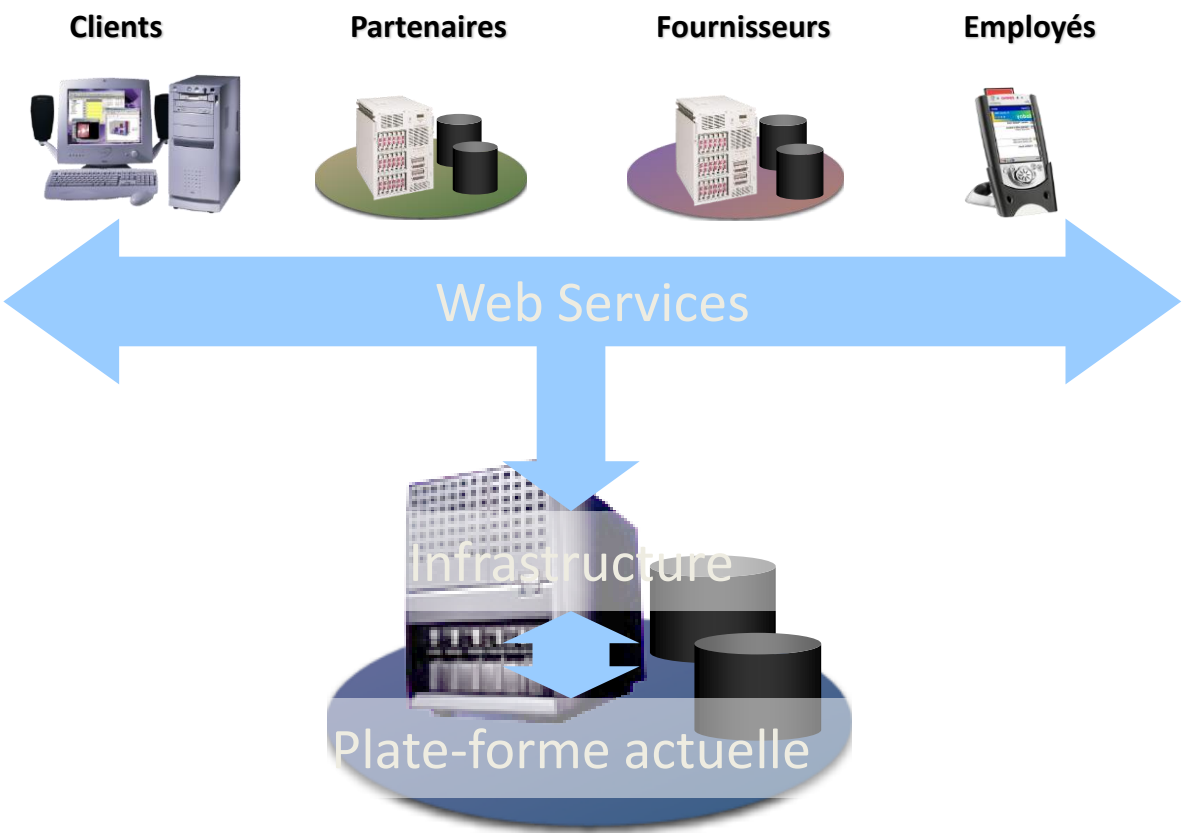
- Assurer un service sans faille
  - Être réactif : 24h/24 et 7j/7

- Accélérer le retour sur investissement
  - Diminuer le coût total des projets
  - Réduire le coût de maintenance des applications
  - Accélérer le « Time to Market »

Evolutions technologiques et limites…

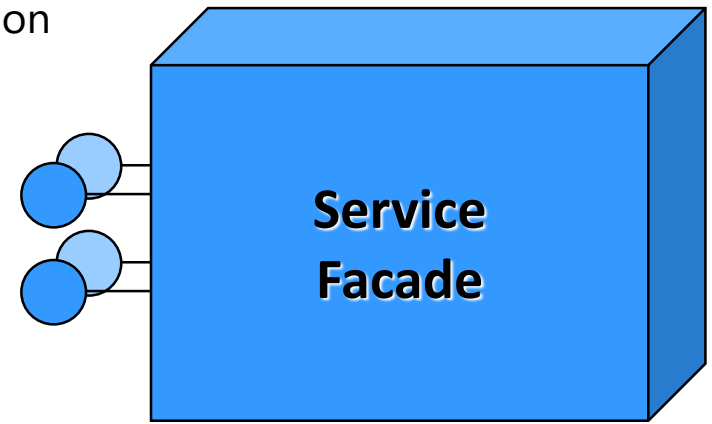
- Modèles de programmation à base d'objets / composants
  - Microsoft® COM / DCOM, CORBA, EJB™
  - Les différentes architectures de développement à base de composants sont difficilement communicantes entre elles.
- Portails offrant davantage de fonctionnalités
  - MSN™, Yahoo, AltaVista, Excite, etc.
  - Les portails web fournissent d'avantages de fonctionnalités, mais cela apporte d'avantage de complexité de conception, ainsi qu'une interdépendance entre les différents sites web.
- Interopérabilité entre les entreprises au niveau de leurs logiques métiers
  - Les entreprises veulent de plus en plus interagir entre elles en partageant leurs logiques métiers.

Interopérabilité entre les entreprises

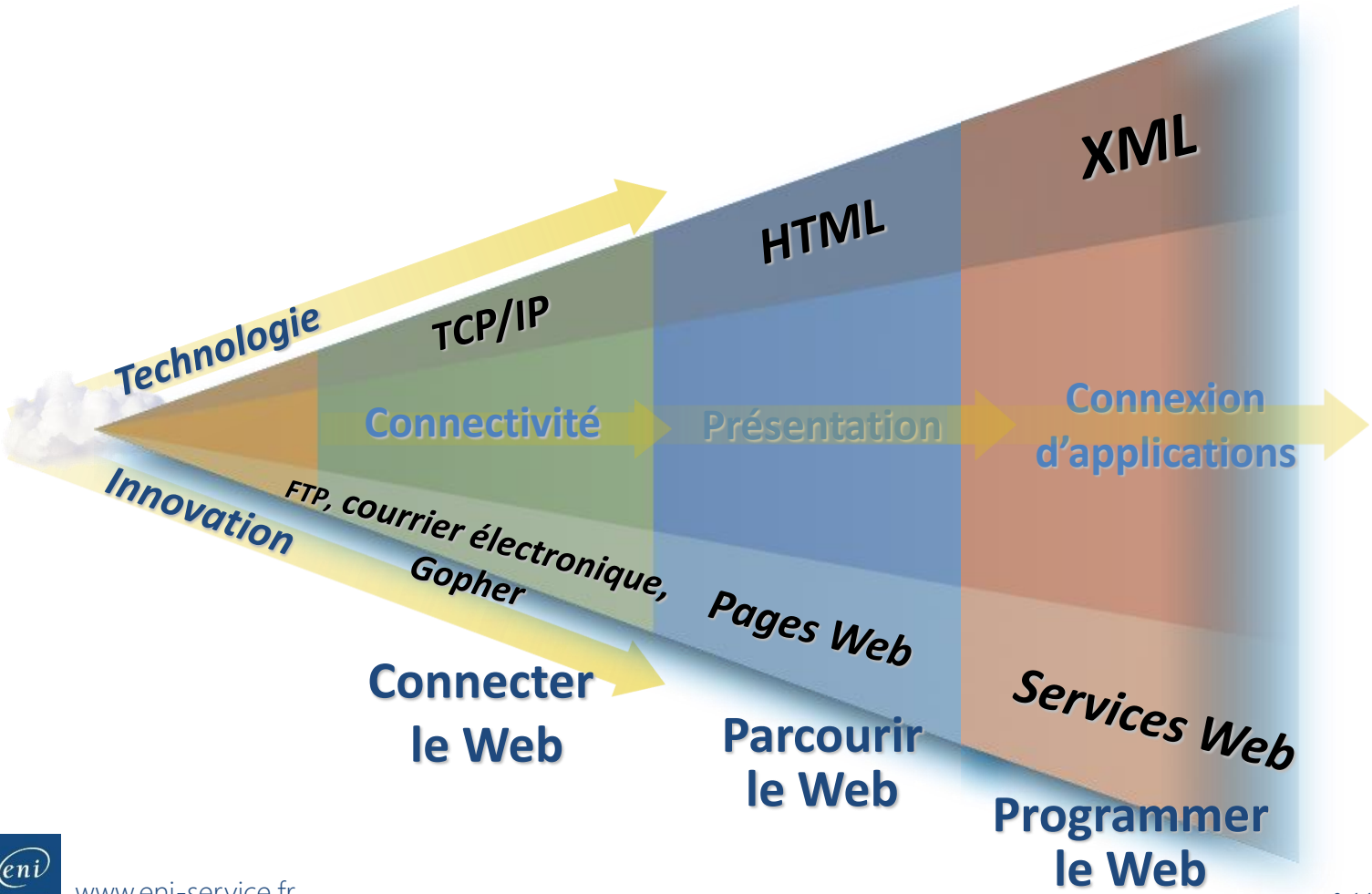


Les Services Web...

- Définition
  - Modèle pour le développement Internet
  - Fourniture d'applications en tant que services
  - Une expérience plus riche et orientée vers la clientèle
  - Accessibles à partir de tout appareil connecté à Internet
  - Fourniture continue de valeur/bits
  - Programmables
- Evolution du modèle de programmation
  - Objet -> Composant -> Service
    - Objets et Méthodes
    - Composants et Interfaces
    - Service Facades et Services



Evolution vers les Services Web



Les Services Web : Une solution pour l'intégration

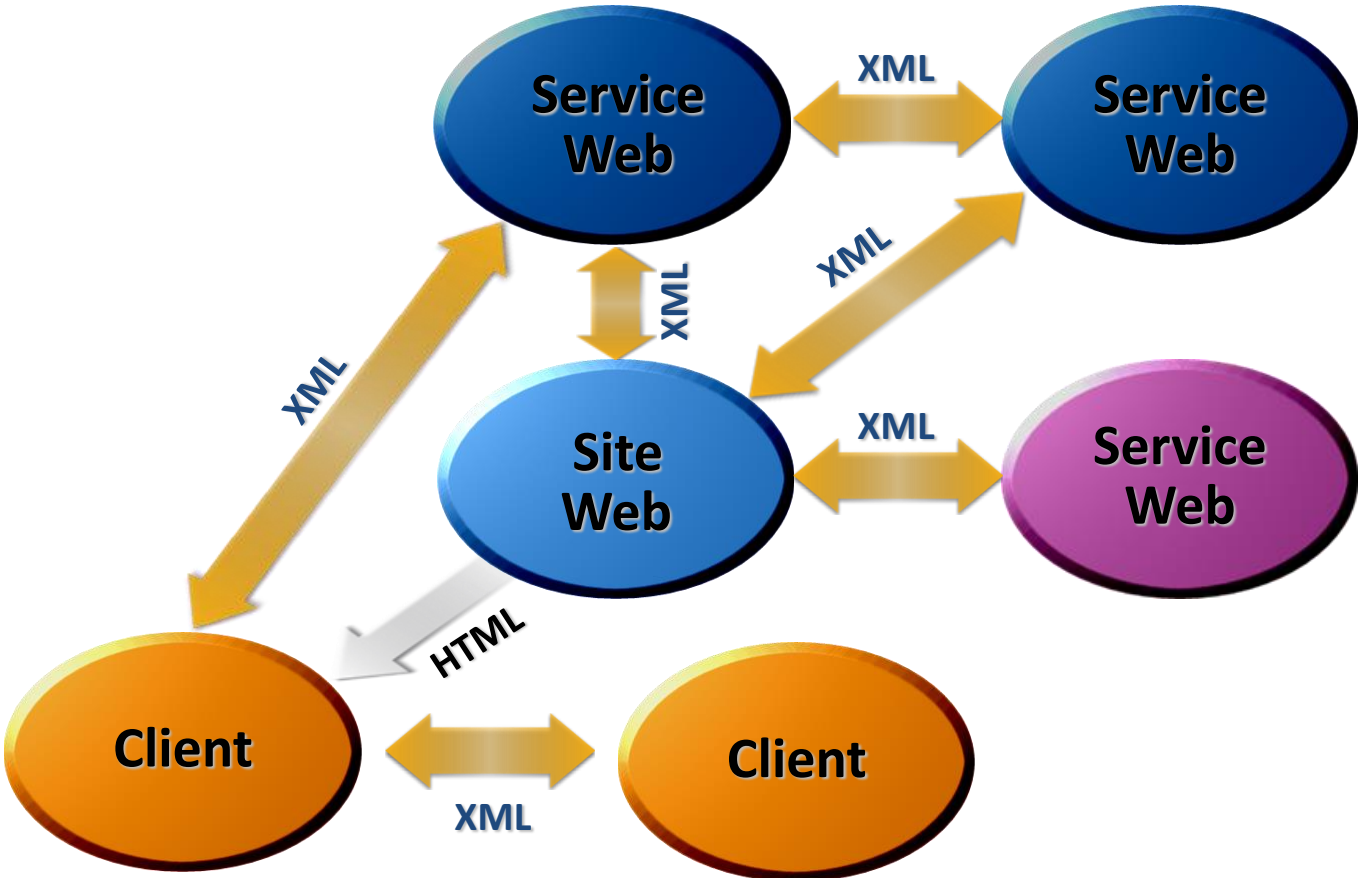
- Les Services Web … Qu'est ce que c'est ?
  - « Librairies » fournissant des données et des services à d'autres applications

- Indépendants
  - du langage d'implémentation (Java, VB, C++,…)
  - de la plate-forme sous-jacente (J2EE, .NET,…)
  - des systèmes d'exploitation (UNIX, Windows,…)

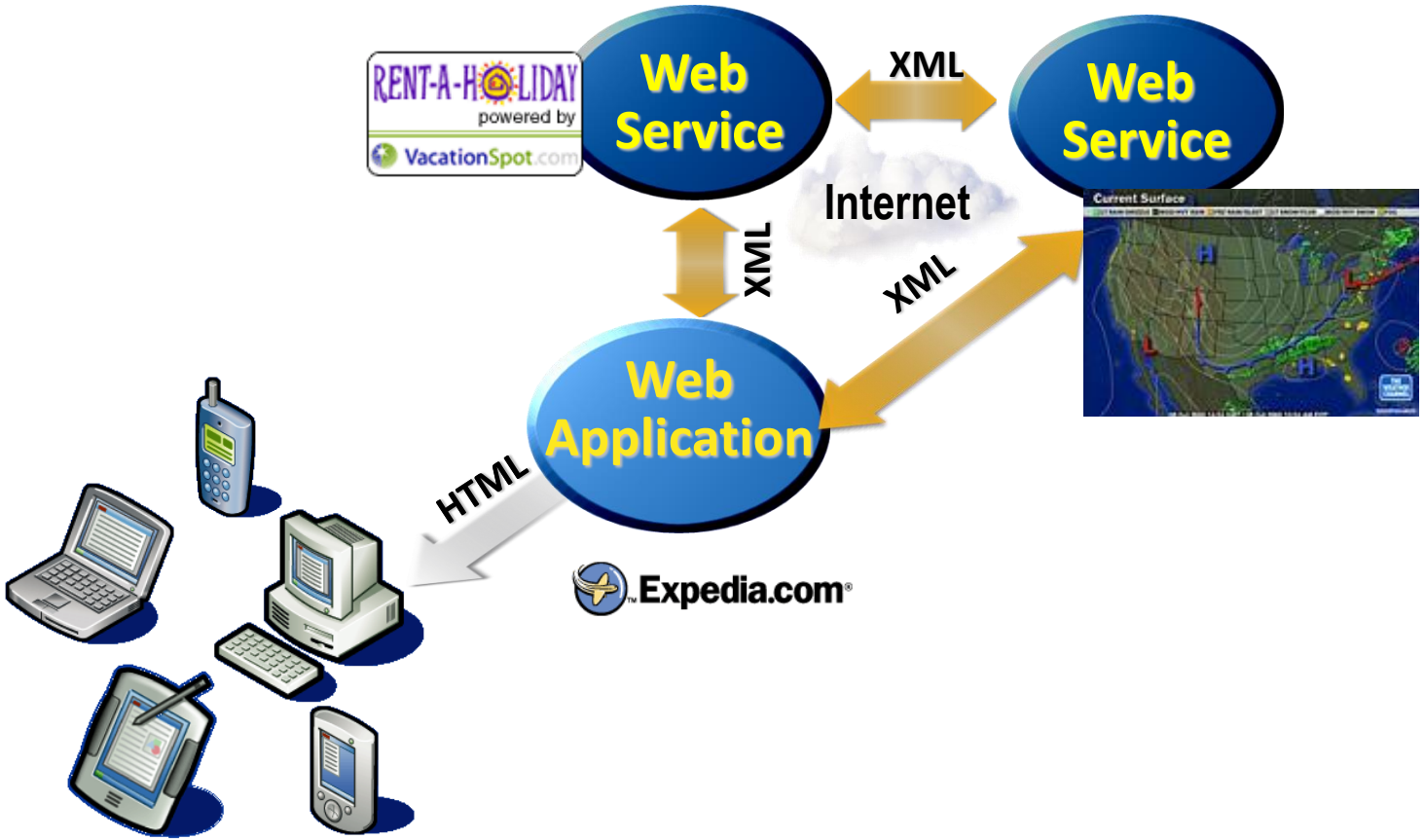
- Non limités aux réseaux locaux, car destinés à être distribués massivement sur Internet

- Les Services Web sont là pour assurer l'intégration et l'interopérabilité
  - EAI : Enterprise Application Integration

Des sites Web au Services Web



Les principe du Service Web dans un site Web



Les composantes d'un Service Web

- Les Services Web SOAP sont basés sur des standards !
  - Eux-mêmes basés sur les standards du web définis par le W3C : HTTP, XML

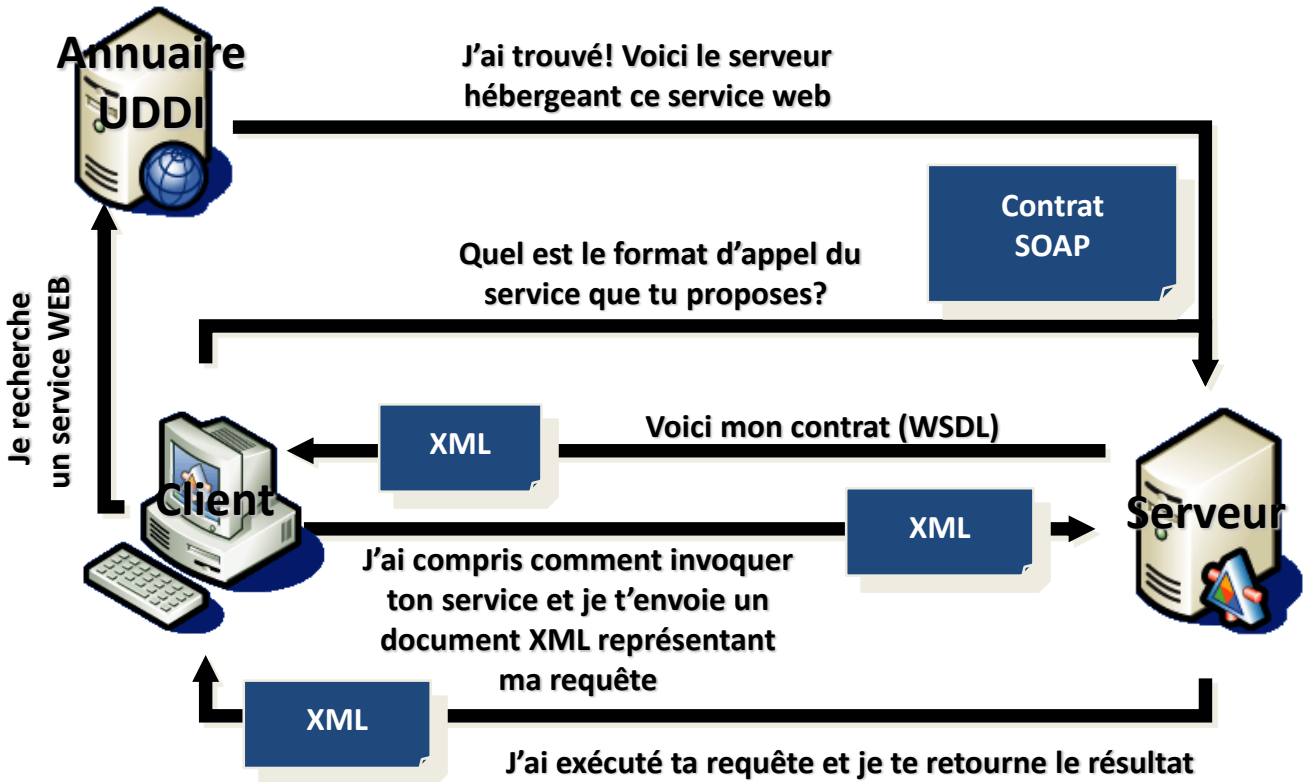
- **WSDL** (*Web Service Description Language*)
  - Langage de description des Services Web au format XML

- **SOAP** (*Simple Object Access Protocol*)
  - Requête/Réponse HTTP au format XML pour invoquer les Services Web

- **UDDI** (*Universal Description Discovery & Integration*)
  - Annuaires de référencement et de recherche des Services Web

Le principe de consommation d'un Service Web

- Service Web statique vs. Service Web dynamique…



L'interface d'un Service Web : WSDL

- Langage de description des Services Web au format XML
  - Décrit les services Web
  - XML
  - Décrit les dépendances
    Par exemple: XSD (Schéma d'objets)
  - Accessible via une simple URL
- Pour pouvoir accéder à un Service Web, un consommateur doit savoir
  - Quelles méthodes sont disponibles
  - Quels paramètres elles acceptent
  - Quels types elles renvoient
- Un Service Web peut être interrogé pour connaître ces informations
  - Il renverra une description de Service Web incluant les messages compris par le Service Web

WSDL : Structure générale

- Un document WSDL est un document XML

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="HelloWorldService"
                  targetNamespace="http://www.formation.fr/webservice/soap">

</wsdl:definitions>
```

- Éléments constitutifs de WSDL
  - Schéma des données
    - Typage de bas niveau pour les paramètres de messages
  - Message
    - Format d'une transmission individuelle
  - PortType
    - Groupe les messages en opérations logiques
  - Liaison
    - Connecte un PortType à une implémentation (habituellement SOAP)
  - Service
    - Définit l'emplacement physique d'un point terminal

WSDL : Le schéma des données

- C'est un schéma XML qui décrit tous les types de données intervenant dans les échanges.
  - On utilise un typage de haut niveau (les types sont encapsulés)

```
<wsdl:types>
    <xs:schema elementFormDefault="unqualified"
               targetNamespace="http://www.formation.fr/webservice/soap"
               version="1.0">
        <xs:element name="sayHello" type="tns:sayHello"/>
        <xs:element name="sayHelloResponse" type="tns:sayHelloResponse"/>
        <xs:complexType name="sayHello">
            <xs:sequence>
                <xs:element minOccurs="0" name="name" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
        <xs:complexType name="sayHelloResponse">
            <xs:sequence>
                <xs:element minOccurs="0" name="return" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:schema>
</wsdl:types>
```

WSDL : Les messages et le « PortType »

- Les messages correspondent aux paramètres et aux retours de méthodes.
- Le PortType contient les opérations qui sont à l'image des méthodes du composant sous-jacent. Il reprend l'interface du composant.

```
<wsdl:message name="sayHelloResponse">
    <wsdl:part element="tns:sayHelloResponse" name="parameters">
    </wsdl:part>
</wsdl:message>
<wsdl:message name="sayHello">
    <wsdl:part element="tns:sayHello" name="parameters">
    </wsdl:part>
</wsdl:message>
<wsdl:portType name="Hello">
    <wsdl:operation name="sayHello">
        <wsdl:input message="tns:sayHello" name="sayHello">
        </wsdl:input>
        <wsdl:output message="tns:sayHelloResponse" name="sayHelloResponse">
        </wsdl:output>
    </wsdl:operation>
</wsdl:portType>
```

WSDL : Liaison et Service

- La liaison (Binding) est l'implémentation du service (décrit par le PortType). Cette implémentation se fait en SOAP.
- Le Service permet essentiellement d'associer le PortType (l'interface) à la liaison (l' implémentation), et à associer le tout à une URL.

```
<wsdl:binding name="HelloWorldServiceSoapBinding" type="tns:Hello">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="sayHello">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="sayHello">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="sayHelloResponse">
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
<wsdl:service name="HelloWorldService">
    <wsdl:port binding="tns:HelloWorldServiceSoapBinding" name="HelloPort">
        <soap:address location="http://localhost:8080/Hello/services/hello"/>
    </wsdl:port>
</wsdl:service>
```

SOAP : L'échange de données

- Requête/Réponse HTTP au format XML pour invoquer les Services Web
- SOAP = XML + HTTP
- Messagerie requête/réponse
  - Utilise HTTP
  - Indépendante de la plate-forme
- Définit une grammaire XML pour
  - Spécifier des noms de méthodes
  - Définir des paramètres et des valeurs de retour
  - Décrire des types

SOAP : Un spécification indépendante d'HTTP

- Un protocole léger pour échanger des informations dans un environnement distribué hétérogène
  - Il améliore la fonctionnalité multi plates-formes
- La spécification SOAP définit
  - Le format de message SOAP
  - Comment envoyer des messages
  - Comment recevoir des réponses
  - Le codage des données

SOAP : Requête SOAP

- Les requête SOAP sont transportées dans des requêtes HTTP POST

```
POST /Hello/services/hello HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
SOAPAction: ""
Content-Length: 316
Host: localhost:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

<soapenv:Envelope xmlns:soap="http://www.formation.fr/webservice/soap"
                  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
</soapenv:Envelope>
```

SOAP : L'enveloppe SOAP

- C'est le format du message !
  - un document XML qui utilise le schéma SOAP
- L'enveloppe est divisée en 2 parties :
  - L'entête (Header) : pour le transport de métadonnées
  - Le corps (Body) : pour le transport des données

```
<soapenv:Envelope xmlns:soap="http://www.formation.fr/webservice/soap"
                  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Header/>
   <soapenv:Body>
      <soap:sayHello>
         <name>Etienne</name>
      </soap:sayHello>
   </soapenv:Body>
</soapenv:Envelope>
```

SOAP : La réponse du service

- C'est une réponse HTTP avec une enveloppe SOAP (enveloppe de réponse)

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/xml;charset=UTF-8
Transfer-Encoding: chunked
Date: Mon, 16 Mar 2015 11:43:40 GMT

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <ns2:sayHelloResponse
            xmlns:ns2="http://www.formation.fr/webservice/soap">

            <return>Bonjour Etienne</return>
        </ns2:sayHelloResponse>
    </soap:Body>
</soap:Envelope>
```

SOAP : Typage des données et expression des érreurs

- SOAP définit aussi l'encodage pour les différents types de données qui est basé sur la technologie schéma XML du W3C.
- Les types simples :
  - un type de base : string, int, float, ...
  - une énumération
  - un tableau d'octets (array of bytes)
- Les types composés :
  - une structure (Struct)
  - un tableau (Array)

- La partie SOAPFault permet d'indiquer qu'une erreur est survenue lors des traitements du service web. Cette partie peut être composée de 4 éléments :
  - faultCode : indique le type de l'erreur (*VersionMismatch* en cas d'incompatibilité avec la version de SOAP utilisée, *MustUnderstand* en cas de problème dans le header du message, *Client* en cas de manque d'informations de la part du client, *Server* en cas de problème d'exécution des traitements par le serveur)
  - faultString : message décrivant l'erreur
  - faultActor : URI de l'élément ayant déclenché l'erreur
  - faultDetail

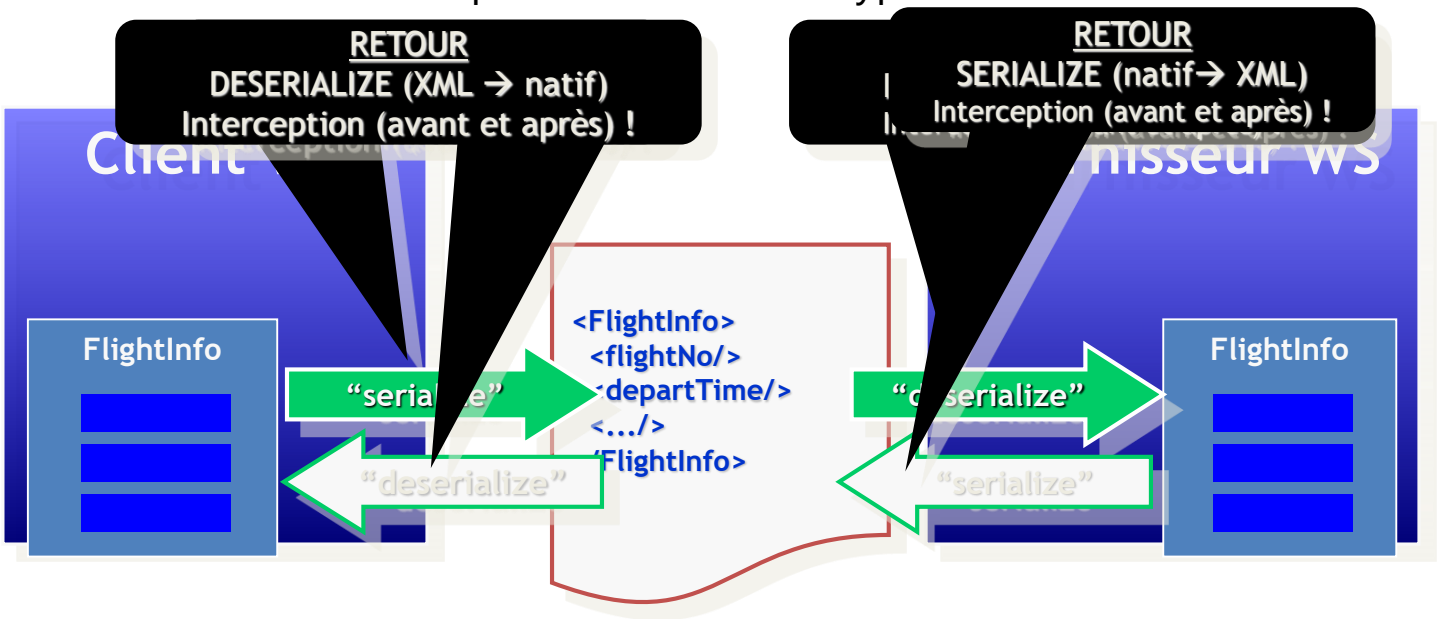
SOAP : Encodage des données complexes

- Le codage des données complexes est réalisé en sérialisant les données en XML

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:authenticateResponse xmlns:ns2="http://www.formation.fr/banque/soap">
         <return>
            <address>40 rue de la Paix</address>
            <zipCode>75000</zipCode>
            <id>0</id>
            <password>password</password>
            <name>DUPONT</name>
            <firstName>Robert</firstName>
            <city>Paris</city>
         </return>
      </ns2:authenticateResponse>
   </soap:Body>
</soap:Envelope>
```

SOAP : Le rôle du moteur SOAP

- Un moteur SOAP est une librairie logicielle responsable des transformations entre un format natif (d'un langage de programmation) et le format XML SOAP.
- Il doit établir une correspondance entre les types de données



SOAP : Style et encodage

- Le protocole SOAP supporte 2 styles de communication :
  - Document
    - (également appelé message-oriented). Ce style de communication fournit niveau d'abstraction très bas et requiert donc plus de travail pour le développement du service. Le paramètre transmis et un document XML et c'est donc le style de communication qui offre la plus grande interopérabilité entre les technologies.
  - RPC (Remote Procedure Call)
    - L'appel de procédure à distance est un appel synchrone d'une méthode qui retourne un résultat. L'utilisation de ce style dépend de la technologie d'implémentation utilisée. Ce mode de communication ne fait pas partie du standard SOAP.
- L'encodage SOAP permet de spécifier au moteur SOAP du serveur qui héberge le service, comment transformer les structures de données d'un langage de programmation (Java par exemple) vers XML, et vice-versa.
  - SOAP Encoding
    - Les types de données sont sérialisés et désérialisés dans les messages SOAP. Ce type d'encodage est défini par le standard SOAP 1.1
  - Literal
    - L'encodage littéral inclus un Schéma XML au message SOAP permettant d'établir la correspondance entre les types.

SOAP : Combinaisons

- La combinaison du style et du type d'encodage peut prendre plusieurs valeurs :
  - RPC/Encoded
  - RPC/Literal
  - Document/Encoded : cette combinaison n'est pas implémentée
  - Document/Literal
- Le style RPC/Encoded a largement été utilisé au début des services web : actuellement ce style est en court d'abandon par l'industrie au profit du style Document/Literal.
- L'appel du service web doit obligatoirement se faire dans le style précisé dans le WSDL puisque celui-ci détermine le format des messages échangés.

SOAP : Spécification de style/encodage

- Le style et le type d'encodage sont précisés dans le WSDL.

```
<wsdl:binding name="HelloWorldServiceSoapBinding" type="tns:Hello">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="sayHello"><soap:operation soapAction="" style="document"/>
        <wsdl:input name="sayHello">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="sayHelloResponse">
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
```

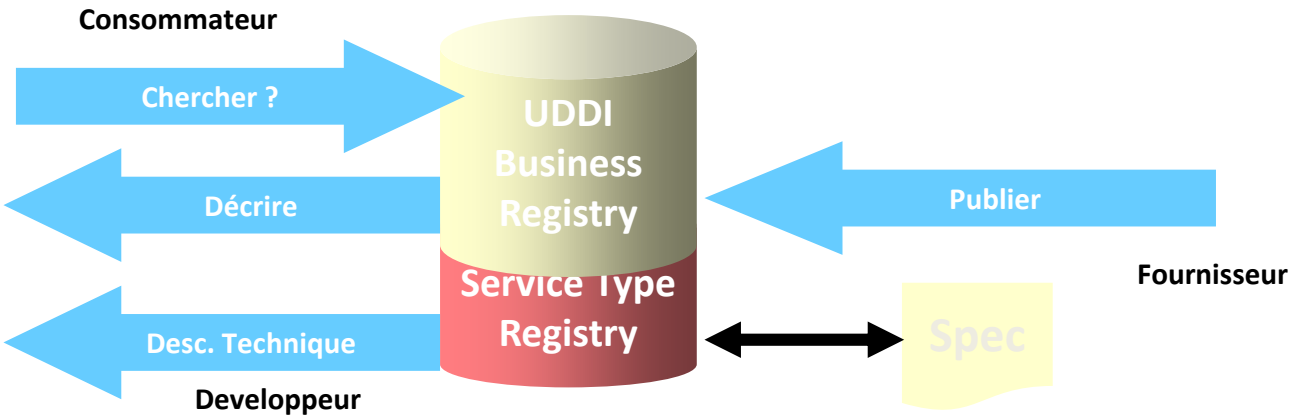
UDDI : Référencement et recherche

- Annuaires de référencement et de recherche des Services Web
- Possibilité de rechercher par activité, service, Service Web (tModel), liaison
- Modèle de requête d'examen en profondeur
  - Utilisation de requêtes find_X pour obtenir un détail de haut niveau – xLists
  - Obtention de détails au moyen d'une requête get_xDetail
- Possibilité de mettre à jour/recenser informations, services et liaisons métier

UDDI : Principes

- Mécanisme de publication pour les fournisseurs de services Web
- Référentiel global hébergé par Microsoft, IBM…
- Administré par uddi.org
  - 260 membres et organisations

- 'Pages blanches'
  - Informations générales sur le fournisseur
- 'Pages jaunes'
  - Classement des produits et services
- 'Pages vertes'
  - Description des service, business process, commercial, etc
- Type de service (tDoc)
  - Pointeur vers la documentation techniques (DEV)



Possibilités et limitations techniques

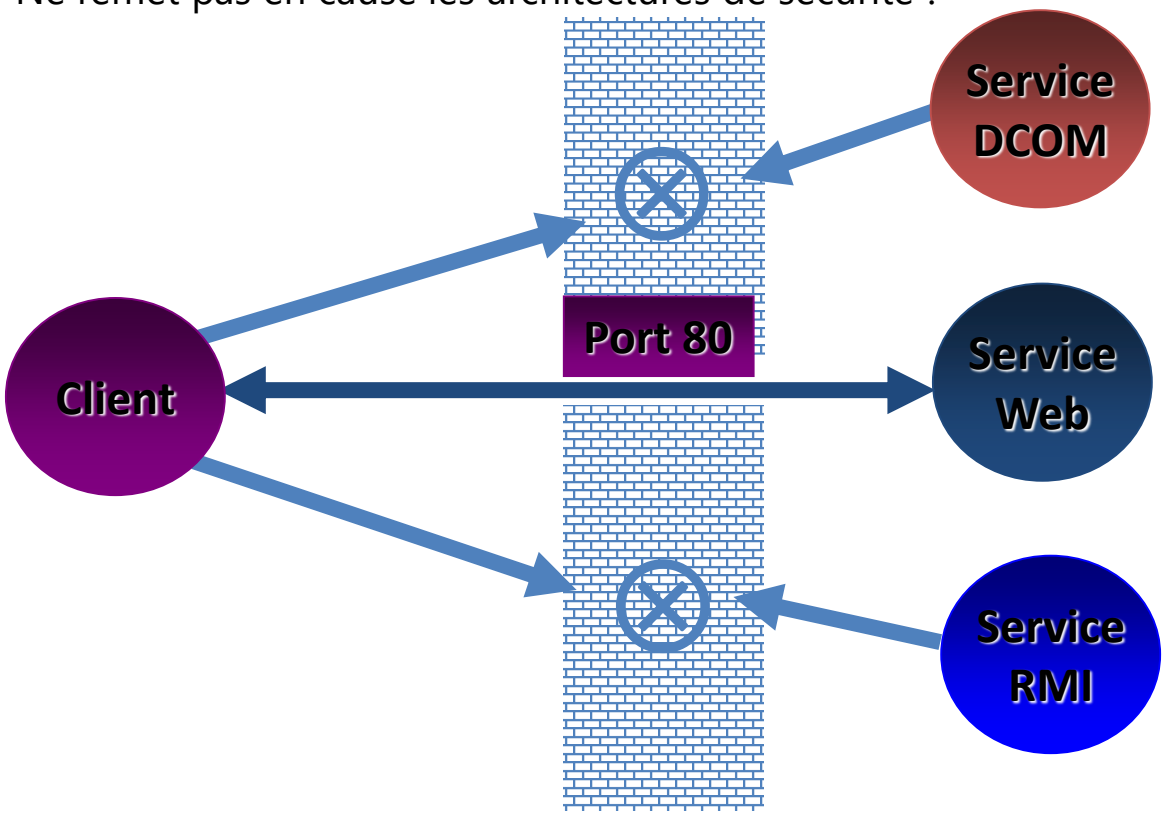
- Invocation au travers d'Internet avec des messages en XML. On peut donc :
  - Invoquer un Service Web de manière synchrone et asynchrone
  - Crypter les messages (HTTPS)
  - Certifier les messages (certificats)
  - Compresser les messages (GZIP)
  - Restreindre les accès par authentification (Basic, Digest, SSL, …)
  - Assurer la scalabilité et la fiabilité des Services Web en les déployant sur une ferme de serveurs
- On s'appuie sur HTTP qui est un protocole déconnecté
  - Pas de transaction
  - Services sans état
    - donc que des méthodes, pas de variables globales dans les composants ou classes sous-jacents

Bonnes pratiques de conception

- Réduire au maximum les appels réseau aux Services Web (très verbeux)
  - Mettre les données en cache
- Ne pas utiliser de sessions (stateless)
- Ne pas utiliser de propriétés : uniquement des méthodes (à cause des sessions)

Sécurité avec les Services Web

- HTTP et Pare-feu : Un protocole firewall friendly !
  - Ne remet pas en cause les architectures de sécurité !



Sécurité avec les Services Web

- Authentification
  - HTTP : Basic, Digest, SSL Client Certificates
    - Impact seulement l'infrastructure
  - Au niveau applicatif
    - Nécessite de ré-authentifier toutes les méthodes du service
    - Un coût d'implémentation !
- Cryptage
  - SSL
    - Impact l'infrastructure seulement
  - XML Encryption
    - Le standard pour SOAP, mais uniquement pour SOAP !
    - Intrusif dans la configuration
- Origine de l'expéditeur
  - Certificats SSL
    - Impact l'infrastructure seulement (sauf côté client)
  - Signature XML
    - Intrusif dans la configuration

Normalisation et interopérabilité

- Le Consortium WS-I

- Initiative de l'industrie pour les services web
  - Ouvert à tous les acteurs des services web.
- Assurer l'interopérabilité des services web
  - Multi-plateforme, applications, et multi-langage
- Faciliter l'adoption et le déploiement
  - Guide d'implémentation & outils de développement et de déploiement.


PIVOTAL
autodesk
COMPAQ
TOSHIBA
IBM
SAP
accenture
McAFEE.COM
UNITED
ORACLE
intel
Peregrine
Microsoft
FUJITSU
KPMG Consulting
Ford Motor Company
DAIMLERCHRYSLER
flamenco networks
GRAND CENTRAL COMMUNICATIONS
jamcracker
LOUDCLOUD
WSI
Web Services Interoperability Organization
hp invent
IONA
POSC
KANA
epicentric
Sun microsystems
CommerceQuest
bea
CAPE CLEAR
epicor
REUTERS
BUSINESS OBJECTS
Rational the software development company
corechange
plumtree
REED ELSEVIER
FrontRange SOLUTIONS
Borland
SYBASE
sas
Sabre
FileNET
macromedia
COMMERCE ONE
ride the light
Qwest
Akamai
webMethods
JDEDWARDS
grooveNETWORKS
DASSAULT SYSTEMES
RealNames
VeriSign
versata
Automating e-Business