



Les Extensions de SOAP

« Contourner » les limitations

Limitations de SOAP...

- ✓ **Sécurité :**
 - Limité à la sécurisation de HTTP ?

- ✓ **Transfert de données :**
 - Données échangées en XML. Donc
 - Données multi-parties ?
 - Données binaires ?
 - Fournir ou consommer des flux de données ?

- ✓ **Des extensions pour répondre à ces problématiques**

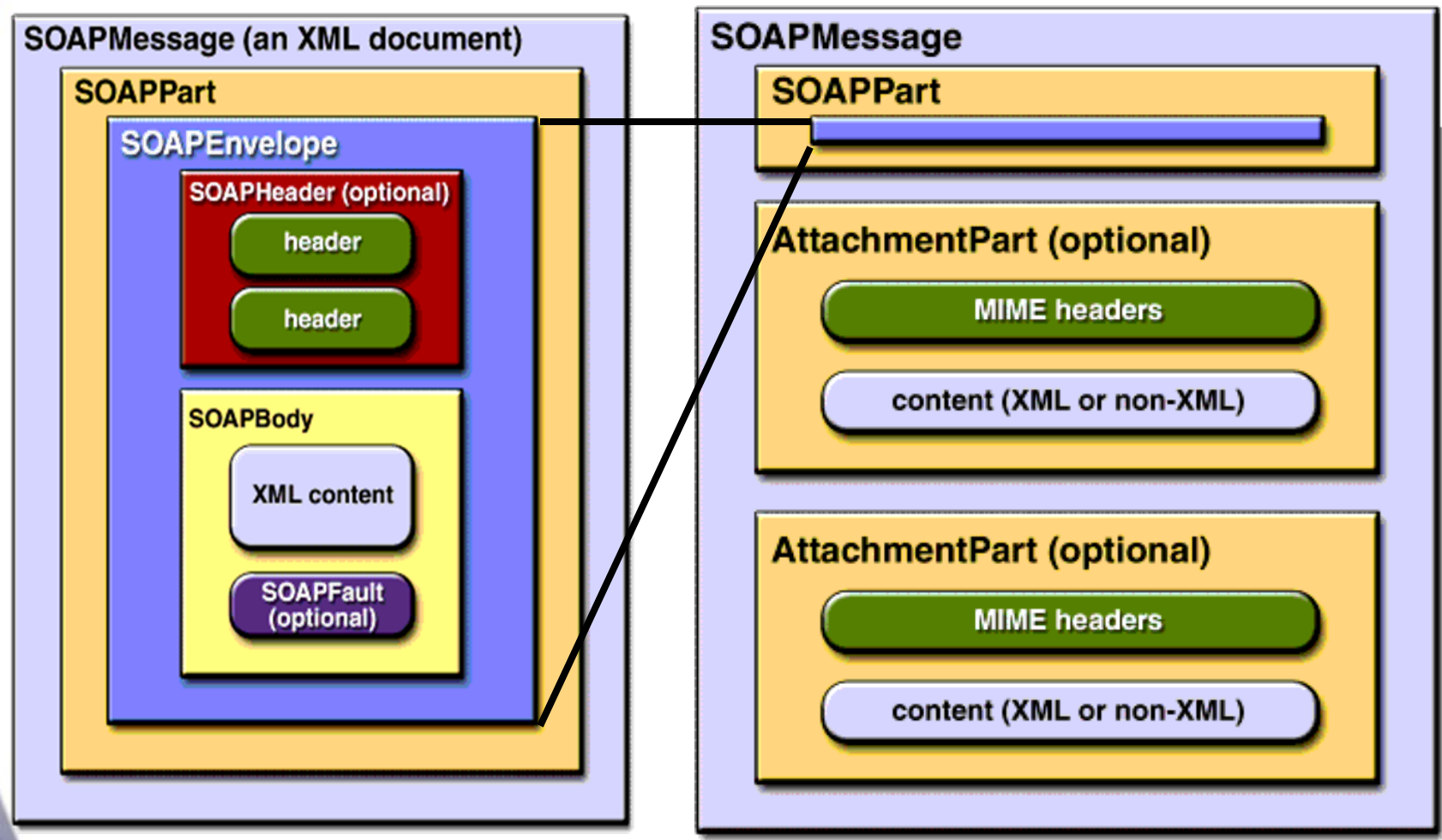
- ✓ **Une sécurité « simple » mais efficace...**
 - Basée sur la sécurité dans HTTP
 - HTTPS: assure une sécurité en point à point
 - S'intègre bien avec la politique des firewall
 - Pas de transfert de code applicatif
 - Uniquement des données
- ✓ **... Mais ne répond pas à tous les problématiques**
 - Repousse le problème des failles
 - Déni de service, failles applicatives, ...
 - Pas un moyen de sécuriser de bout en bout dans des échanges
- ✓ **Donc d'autres standards nécessaires**
 - SAML 2.0: Security Assertion Markup Language (OASIS)
 - XACML 2.0: eXtensible Access Control Markup Language (OASIS)

Attachements avec SOAP

- ✓ **SOAP with Attachments (SwA) ou MIME for Web Services**
 - Utilise MIME

- ✓ **MIME: Multipurpose Internet Mail Extensions**
 - Pour le transport de gros fichiers
 - Placés en dehors de la partie XML
 - Avantages :
 - Simple
 - Inconvénients :
 - Impossible de faire du streaming
 - Le séparateur entre deux « attachements » MIME est une chaîne de caractères
 - Pas standardisé (juste une note du W3C)

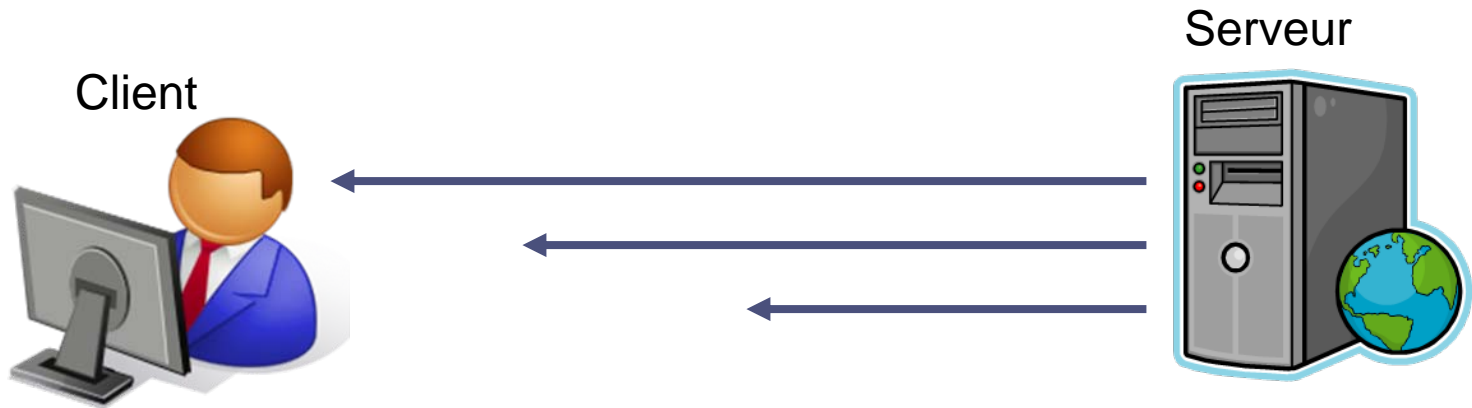
Les Attachements MIME



Les Attachements DIME

- ✓ **DIME: Direct Internet Message Encapsulation**
 - Proposé par Microsoft au début des années 2000
 - Pour le transport de flux continus
 - Placés en dehors de la partie XML
 - A chaque stream correspond un "DIME-record"
 - Un flag avertit le récepteur de la réception du dernier stream
 - Ainsi, il est possible de commencer à envoyer les premiers streams du flux avant même que tous les streams soient créés par la source
- ✓ **Avantages**
 - Simplicité
 - Permet la transmission de flux de données (binaires)
- ✓ **Inconvénients**
 - Impossible à déboguer (données binaires)
 - N'a jamais été standardisé (ni IETF, ni OASIS, ni W3C)

Autres extensions : les attachements DIME



Un DIME-Parser analyse
et reconstitue le flux à
partir des DIME-records
envoyés par le serveur

Un DIME-generator
crée les streams et
constitue les DIME-
records à envoyer
au client

Les attachements MTOM

- ✓ **Message Transmission Optimization Mechanism :**
 - Méthode pour l'envoi efficace de données binaires
 - Recommandation du W3C
 - Supplante SwA et DIME pour les Web Services
- ✓ **Détails:**
 - Abstract SOAP Transmission Optimization Feature
 - Optimized MIME Multipart/Related Serialization of SOAP Messages (XOP)
 - HTTP SOAP Transmission Optimization Feature
- ✓ **Avantages:**
 - Fournit un moyen d'envoyer des données binaires dans leur format d'origine (évitant l'augmentation des infos transmises)
 - La transformation en base64 augmente la taille des données de 33%

Portée et Limitations de SOAP

- ✓ **SOAP est simple et extensible...**
 - **Format XML over HTTP**
 - **Multi-langages**
 - **Multi-plateformes**

- ✓ **... mais il ne couvre pas les fonctions suivantes :**
 - **Distributed garbage collection**
 - **Boxcarring or batching of messages**
 - **Objects-by-reference (qui requière distributed garbage collection)**
 - **Activation (qui requière objects-by-reference)**

Comparaison

	RMI	RPC	DCOM	CORBA	SOAP
Qui	SUN	SUN/OSF	MicroSoft	OMG	W3C
Plate-formes	Multi	Multi	Win32	Multi	Multi
Langages de Programmation	Java	C, C++, ...	C++, VB, VJ, OPascal, ...	Multi	Multi
Langages de Définition de Service	Java	RPCGEN	ODL	IDL	XML
Réseau	TCP, HTTP, IIOP customisable	TCP, UDP	IP/IPX	GIOP, IIOP, Pluggable Transport Layer	RPC, HTTP SNMP
Firewall	Tunneling HTTP			HTTP Tunneling CORBA Firewall	HTTP
Nommage	RMI, JNDI, JINI	IP+Port	IP+Nom	COS Naming COS Trader	IP+Port, URL
Transaction	Non	Non	MTS	OTS, XA	Extension applicative dans le header
Extra	Chargement dynamique des classes			Services Communs Services Sectoriels	

```
- <binding name="TestSoapBinding" type="tns:TestSoapPortType">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
  <operation name="Multiply">
    <soap:operation style="rpc" soapAction="http://soapinterop.org/Multiply" />
    <input>
      <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" use="encoded"
        namespace="http://soapinterop.org" />
    </input>
    <output>
      <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" use="encoded"
        namespace="http://soapinterop.org" />
    </output>
  </operation>
  <operation name="Add">
    <soap:operation style="document" soapAction="http://soapinterop.org/Add" />
    <input>
      <soap:body use="literal" />
    </input>
    <output>
      <soap:body use="literal" />
    </output>
  </operation>
</binding>
<service name="MSInterop1DocAndRPCService">
  <port name="TestSoap" binding="tns:TestSoapBinding">
    <soap:address location="http://localhost/services/msInteropDocAndRpc" />
  </port>
</service>
</definitions>
```

Web Services Description Language

WSDL

✓ **Spécification**

- v1.1 pas « approuvée » par le W3C (note du 15-03-2001)
 - Soutenu par Ariba, IBM, Microsoft
- v1.2 « Working Draft » du W3C (11-06-2003)
- v2.0 recommandation du W3C (27-06-2007)

✓ **Objectif**

- Interface publique d'accès à un Web Service
- Comment communiquer pour utiliser le service (ensemble d'opérations et de messages abstraits reliés (bind) à des protocoles et des serveurs réseaux)

✓ **Grammaire XML (schema XML)**

- Modulaire (import d'autres documents WSDL et XSD)

✓ **Séparation entre la partie abstraite et concrète**

WSDL 1.1

- ✓ **<types>**
 - Contient les définitions des types (utilise un système de typage comme XSD)
- ✓ **<message>**
 - Décrit les noms et types d'un ensemble de champs à transmettre
 - Paramètres d'une invocation, valeur du retour, ...
- ✓ **<portType>**
 - Décrit un ensemble d'opérations et les messages impliqués (0 ou 1 en entrée, 0 ou n en sortie). Partie la plus importante
- ✓ **<binding>**
 - Spécifie une liaison d'un <porttype> à un protocole concret (SOAP1.1, HTTP1.1, MIME, ...). Un portType peut avoir plusieurs liaisons !
- ✓ **<port>**
 - Spécifie un point d'entrée (endpoint) comme la combinaison d'un <binding> et d'une adresse réseau
- ✓ **<service>**
 - Pour agréger un ensemble de ports

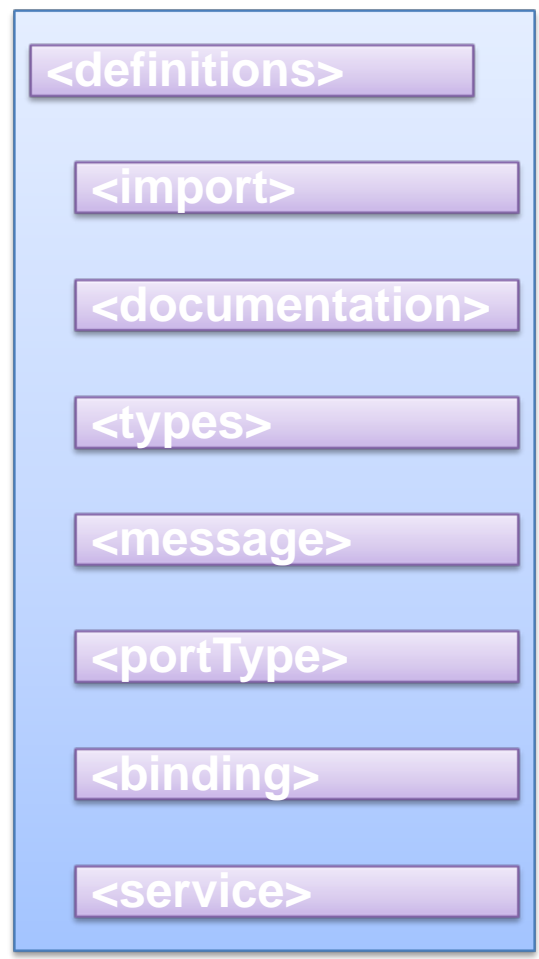
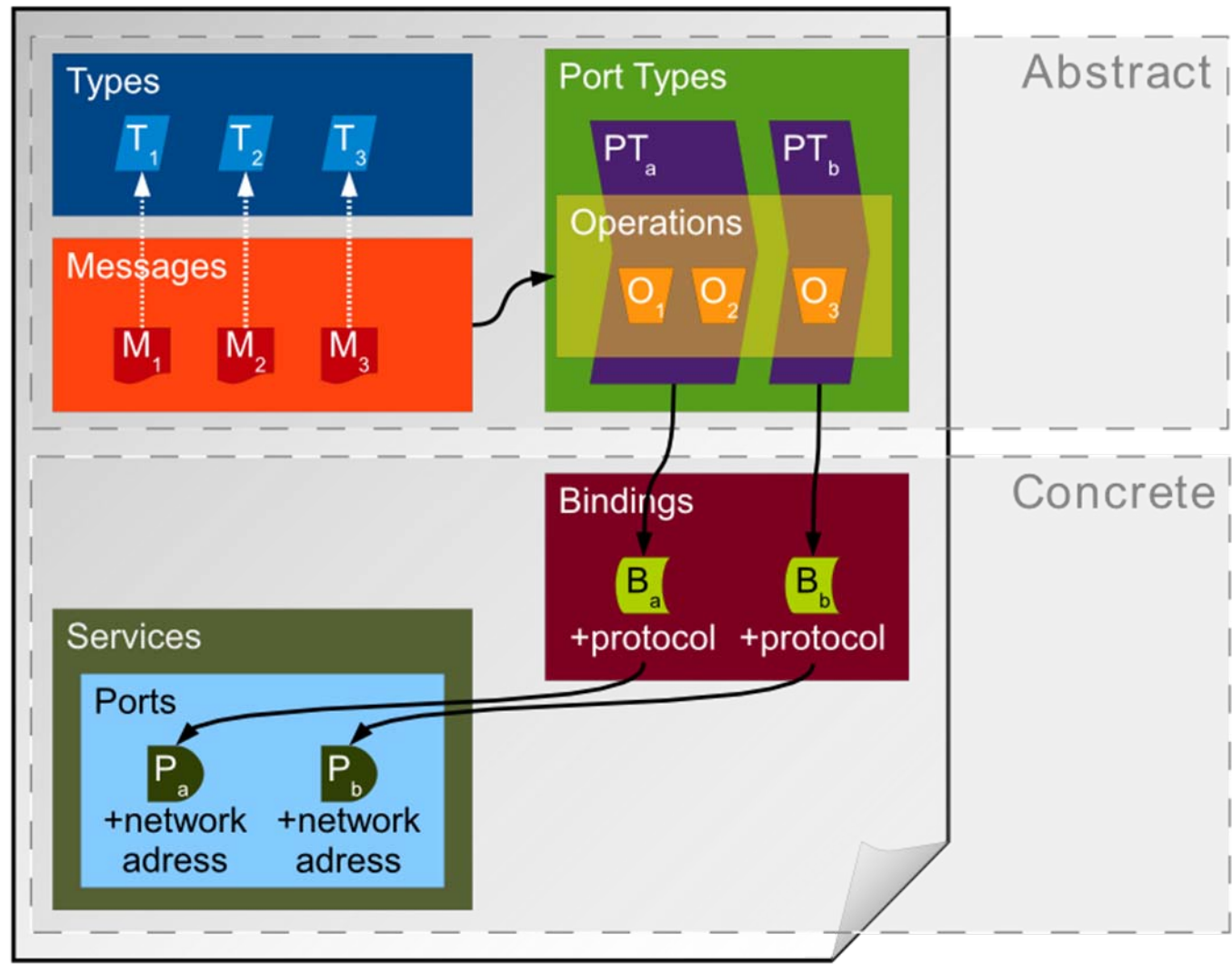


Schéma de WSDL 1.1



Élément <types>

- ✓ Contient les définition de types utilisant un système de typage (comme XSD).
- ✓ Exemple

```
<!-- type defs -->
<types>
  <xsd:schema targetNamespace="urn:xml-soap-address-demo"
    xmlns:xsd="http://www.w3.org/1999/XMLSchema">
    <xsd:complexType name="phone">
      <xsd:element name="areaCode" type="xsd:int"/>
      <xsd:element name="exchange" type="xsd:string"/>
      <xsd:element name="number" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="address">
      <xsd:element name="streetNum" type="xsd:int"/>
      <xsd:element name="streetName" type="xsd:string"/>
      <xsd:element name="city" type="xsd:string"/>
      <xsd:element name="state" type="xsd:string"/>
      <xsd:element name="zip" type="xsd:int"/>
      <xsd:element name="phoneNumber" type="typens:phone"/>
    </xsd:complexType>
  </xsd:schema>
</types>
```

Élément <message>

- ✓ Décrit les noms et types d'un ensemble de champs à transmettre
 - Paramètres d'une invocation, valeur du retour, ...

- ✓ Exemple

```
<!-- message declns -->
```

```
<message name="AddEntryRequest">
```

```
  <part name="name" type="xsd:string"/>
```

```
  <part name="address" type="typens:address"/>
```

```
</message>
```

```
<message name="GetAddressFromNameRequest">
```

```
  <part name="name" type="xsd:string"/>
```

```
</message>
```

```
<message name="GetAddressFromNameResponse">
```

```
  <part name="address" type="typens:address"/>
```

```
</message>
```

Élément <porttype>

- ✓ Décrit un ensemble d'opérations (peut être vu comme une librairie, un module ou une classe)

- ✓ Plusieurs types d'opérations
 - One-way
 - Le point d'entrée reçoit un message (<input>) mais sans réponse
 - Request-response
 - Le point d'entrée reçoit un message (<input>) et retourne un message corrélé (<output>) ou un ou plusieurs messages de faute (<fault>).
 - Solicit-response
 - Le point d'entrée envoie un message (<output>) et reçoit un message corrélé (<input>) ou un ou plusieurs messages de faute (<fault>).
 - Binding HTTP : 2 requêtes HTTP par exemple
 - Notification
 - Le point d'entrée envoie un message de notification (<output>)

- ✓ Paramètres
 - Les champs des messages constituent les paramètres (in,out, inout) des opérations

Exemple <porttype>

✓ Exemple

<!-- port type declarations -->

<portType name="AddressBook">

<!-- One way operation -->

<operation name="addEntry">

<input message="AddEntryRequest"/>

</operation>

<!-- Request-Response operation -->

<operation name="getAddressFromName">

<input message="GetAddressFromNameRequest"/>

<output message="GetAddressFromNameResponse"/>

</operation>

</portType>

Élément <binding>

- ✓ Spécifie une liaison d'un <portType> à un protocole concret (SOAP1.1, HTTP 1.1, MIME, ...)

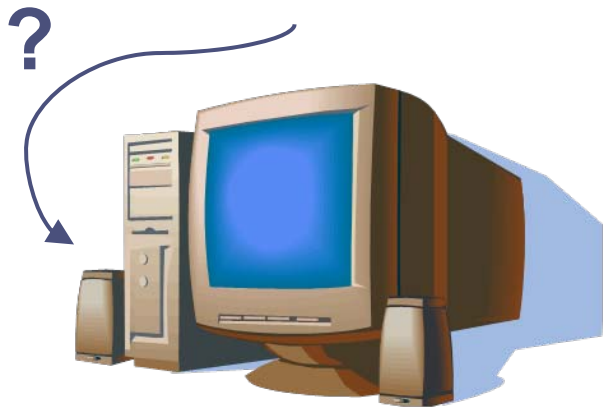
```
<!-- binding decls -->
<binding name="AddressBookSOAPBinding" type="AddressBook">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="addEntry">
    <soap:operation soapAction=""/>
    <input> <soap:body use="encoded" namespace="urn:AddressFetcher2"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/> </input>
    <output> <soap:body use="encoded" namespace="urn:AddressFetcher2"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/> </output>
  </operation>
  <operation name="getAddressFromName">
    <soap:operation soapAction=""/>
    <input> <soap:body use="encoded" namespace="urn:AddressFetcher2"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></input>
    <output> <soap:body use="encoded" namespace="urn:AddressFetcher2"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/></output>
  </operation>
</binding>
```

Élément <service>

- ✓ Une collection de points d'entrée (endpoint) relatifs
- ✓ Exemple

```
<?xml version="1.0" ?>
<definitions name="urn:AddressFetcher"
  targetNamespace="urn:AddressFetcher2"
  xmlns:typens="urn:xml-soap-address-demo"
  xmlns:xsd="http://www.w3.org/1999/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
...
<!-- service decln -->
<service name="AddressBookService">
  <port name="AddressBook" binding="AddressBookSOAPBinding">
    <soap:address location="http://www.mycomp.com/soap/servlet/rpcrouter"/>
  </port>
</service>
</definitions>
```

10101001100
00010011101
11100011100



Environnements de Programmation

Un exemple .NET C#

Des Environnements pour Programmer les Web Services

- ✓ Pour une mise en pratique des Web Services
 - TD 3: ASP .NET
 - <http://dotnet.developpez.com/cours/?page=csharp#servicewebscs>
 - TD 4: Java AXIS
 - <http://javaweb.developpez.com/cours/?page=architectures-web#webservices>
 - TD 5: gSOAP C/C++
 - <http://www.cs.fsu.edu/~engelen/soap.html>

- ✓ Permet de simplifier la mise en œuvre
 - Génère automatiquement le WSDL
 - Crée le binding avec SOAP
 - ... mais comment étendre le comportement par défaut ??

Génération du WSDL par le Framework WS de ASP.NET

✓ Ajout d'un attribut [WebMethod]

[WebMethod]

```
public string DisBonjour(string monsieur) {  
    System.Threading.Thread.Sleep(5000);  
    return "Bonjour, M. " + monsieur + ". Nous sommes " +  
        DateTime.Now.DayOfWeek;  
}
```

✓ Accès dynamique à la description du Web Service

- <http://localhost/BonjourWS.asmx?WSDL>
- Introspecte la classe BonjourWS grâce à ServiceDescriptionReflector

```
[WebMethod]  
public string DisBonjour(string monsieur)  
{  
    System.Threading.Thread.Sleep(5000);  
    return "Bonjour, M. " + monsieur;  
}
```



ServiceDescriptionReflector

Source: Sébastien Bouchet

Attributs de l'Attribut [WebMethod]

- ✓ [WebMethod *attribut=valeur*]
 - (EnableSession=false)
 - Aucune donnée persistante entre le consommateur et le service
 - (BufferResponse=true)
 - Créé une mémoire tampon stockant la réponse
 - (CacheDuration=*nombre de secondes*)
 - Garder les réponse en cache pendant une certaine durée
 - (Description="*une description*")
 - Donne une description à la méthode
 - (MessageName="*alias nom de méthode*")
 - Surcharge de méthode du service
 - (TransactionOption=TransactionOption.[Disabled | Required | Supported | NotSupported | RequiresNew])
 - Gère les transactions

Exemple d'Extension du « Contrat WDSL par défaut »

✓ Exemple:

- Mécanisme de mise en cache
- Cacher les réponses du Web Service (côté consommateur)
 - Eviter au client de refaire une requête si déjà faite dans un laps de temps donnée

✓ But:

- Exemple pédagogique plus que réel
- Illustrer la technique d'extension (d'autres approches sont possibles pour réaliser le cache)

Introspection pour Générer le WSDL

- ✓ **Lors de l'exécution du** `ServiceDescriptionReflector`
 - **Pour chaque attribut** `[WebMethod]` **rencontrée**
 - **Appel** `ReflectMethod` **de tous les** `SoapExtensionReflector`

```
public class MyReflector : SoapExtensionReflector
{
    public override void ReflectMethod()
    {
        ProtocolReflector refl = this.ReflectionContext;
        //Faire quelque chose avec refl, comme insérer
        //des extensions
    }
}
```

- ✓ **Modifier le WSDL généré:**
 - comment représenter une méta-donnée (et accessoirement quel est son contenu)
 - où l'insérer dans le contrat
 - comment définir la nouvelle propriété de la méthode web

Nouvel Attribut à Créer

```
public enum CachingMode
{
    Rotating, Sliding
}

public enum CachingPeriod
{
    ToNextHour, ToNextDay, ToNextWeek, ToNextMonth, ToNextYear
}

[AttributeUsage(AttributeTargets.Method)]
public class
    ClientSideCacheabilityAttribute :
        Attribute
{
    private CachingMode mode =
        CachingMode.Rotating;
    private CachingPeriod period =
        CachingPeriod.ToNextDay;
    private long validity;

    public
        ClientSideCacheabilityAttribute() {}
```

```
public
    ClientSideCacheabilityAttribute(CachingPeriod p)
    {
        mode = CachingMode.Rotating;
        period = p;
    }

    public
        ClientSideCacheabilityAttribute(long validityTimeSpanMs)
    {
        mode = CachingMode.Sliding;
        validity = validityTimeSpanMs;
    }

    public CachingMode Mode {..}

    public CachingPeriod Period {..}

    public long Validity {..}
}
```

Ajout du Nouvel Attribut et Impact sur le WSDL

✓ Ajout du Nouvel Attribut:

```
[WebMethod]
```

```
[ClientSideCacheability(CachingPeriod.ToNextDay)]
```

```
public string DisBonjour(string monsieur)
```

```
{
```

```
    System.Threading.Thread.Sleep(5000);
```

```
    return "Bonjour, M. " + monsieur + ". Nous sommes " +
```

```
    DateTime.Now.DayOfWeek;
```

```
}
```

✓ Ce nouvel attribut doit se retrouver dans le WSDL

- Où ajouter cette information ???

A Ajouter dans le binding WSDL

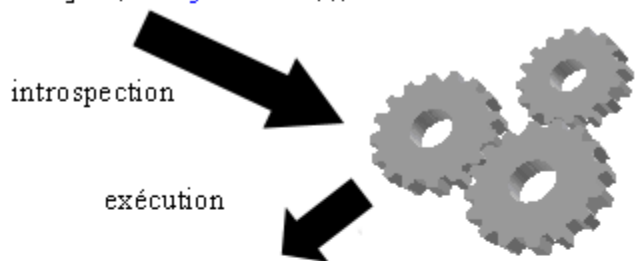
- ✓ La spécification WSDL impose de le faire au niveau du binding

```
<operation name="DisBonjour">  
  <soap:operation soapAction="http://tempuri.org/DisBonjour"  
    style="document" />  
  <dng:cachePolicy >  
    <dng:Mode>Rotating</dng:Mode>  
    <dng:Period>ToNextDay</dng:Period>  
    <dng:Validity>0</dng:Validity>  
  </dng:cachePolicy>  
  <input>  
    <soap:body use="literal" />  
  </input>  
  <output>  
    <soap:body use="literal" />  
  </output>  
</operation>
```

Comment le Générer ?

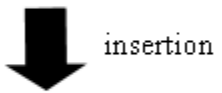
- ✓ Créer une classe **ServiceDescriptionFormatExtension**
 - Pour créer les méta-données
- ✓ La classe **CachingExtension**
 - Possède les mêmes propriétés que l'attribut **ClientSideCacheabilityAttribute**

```
[WebMethod]
[ClientSideCacheability(CachingPeriod.ToNextDay)]
public string DisBonjour(string monsieur){}
```

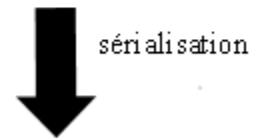


```
public class MyExtension : SoapExtensionReflector {
    public override void ReflectMethod() {
        ProtocolReflector refl = this.ReflectionContext;

        //Insertion d'extension dans le binding
    }
}
```



```
[XmlFormatExtension("myExt", ...)]
public class MyWSDLExt :
    ServiceDescriptionFormatExtension
{
}
```



```
<myExt xmlns="...">
    <myProp>...</myProp>
</myExt>
```

document WSDL

```
[XmlFormatExtension("cachePolicy", "urn:dng-articles.org:seb:cachext", typeof(OperationBinding))]
[XmlFormatExtensionPrefix("dng", "urn:dng-articles.org:seb:cachext")]
public class CachingExtension : ServiceDescriptionFormatExtension {... }
```

Reste à Compléter ReflectMethod

✓ Ecrire SoapExtensionReflector

- qui va insérer cette extension au document WSDL en cours de génération pour chaque méthode cacheable

```
public override void ReflectMethod(){
    ProtocolReflector refl = this.ReflectionContext;
    ClientSideCacheabilityAttribute policy =
        refl.Method.GetCustomAttribute(typeof(ClientSideCacheabilityAttribute))
        as ClientSideCacheabilityAttribute;
    if(policy != null){
        refl.OperationBinding.Extensions.Add(new
        CacheabilityExtension(policy));
    }
}
```

✓ Que reste-t-il à faire pour que cela fonctionne ?

Modifier le Comportement du Générateur de Proxy

✓ Consommation d'un Service Web

– Génération d'un proxy pour le service

- `wSDL http://localhost/Service?WSDL /out:lenomdelaclasseproxy.cs`

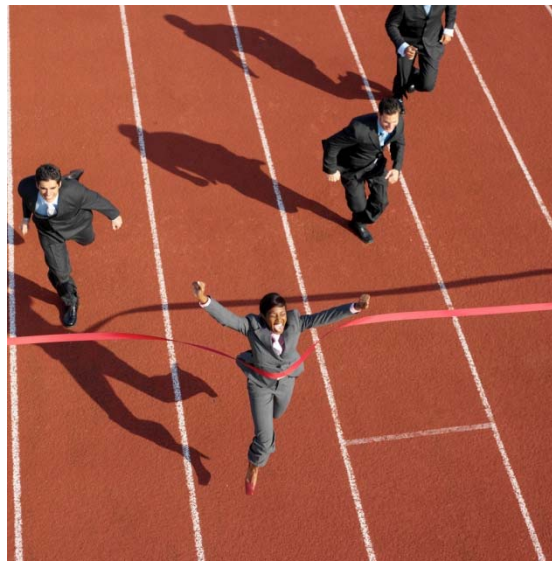
```
C:\>wSDL http://127.0.0.1/PetitExemple/Exemple.asmx?wSDL /out:petitExemple.cs
Microsoft (R) Web Services Description Language Utility
[Microsoft (R) .NET Framework, Version 1.1.4322.5731]
Copyright (C) Microsoft Corporation 1998-2002. All rights reserved.

Writing file 'petitExemple.cs'.

C:\>
```

✓ Voir la suite de l'exemple:

- <http://www.dotnetguru.org/articles/dossiers/extensionsWSDL/ExtensionsWSDL.htm>



Conclusion

Les Services Web

Les Services Web

- ✓ La 3^{ème} génération du Web
- ✓ Technologies Standards du Web
 - SOAP (1.1 puis SOAP 1.2)
 - WSDL (1.1, 1.2 puis 2.0)
- ✓ Technologies non standardisées
 - UDDI, DISCO
 - GXA (Global XML Architecture)
 - WSDD, WSFL, ASMX, ...

Cinématique générale

Service

```
public class MyWebService {  
    public String sendMessage(String name) {  
        return " Hello " + name  
    }  
}
```

(Java, C++, ...)

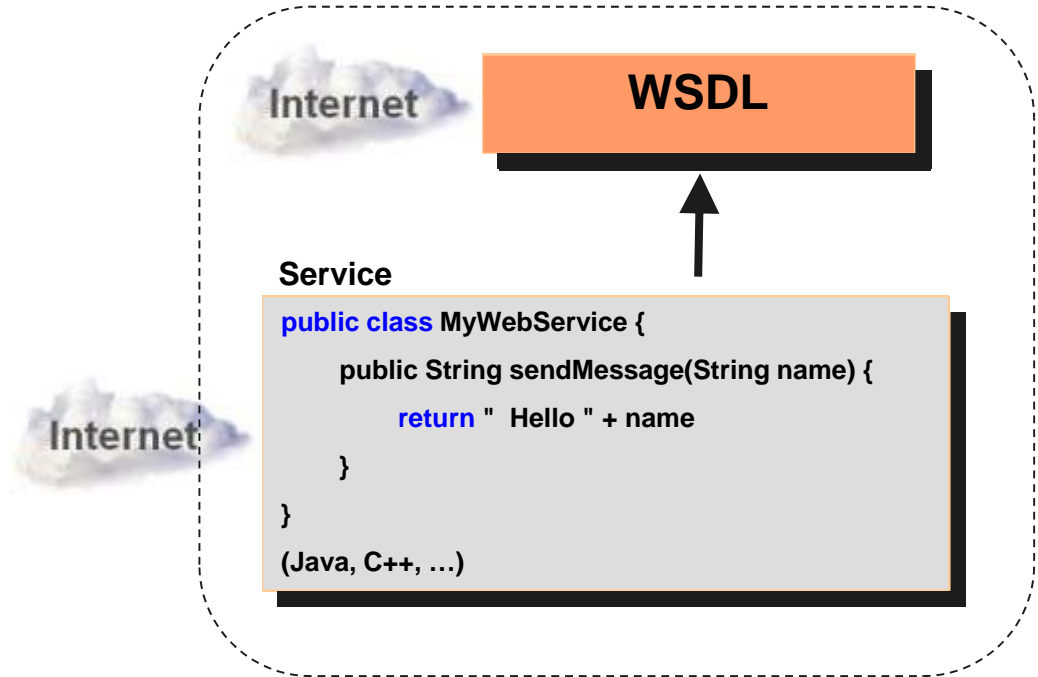
Cinématique générale

Internet

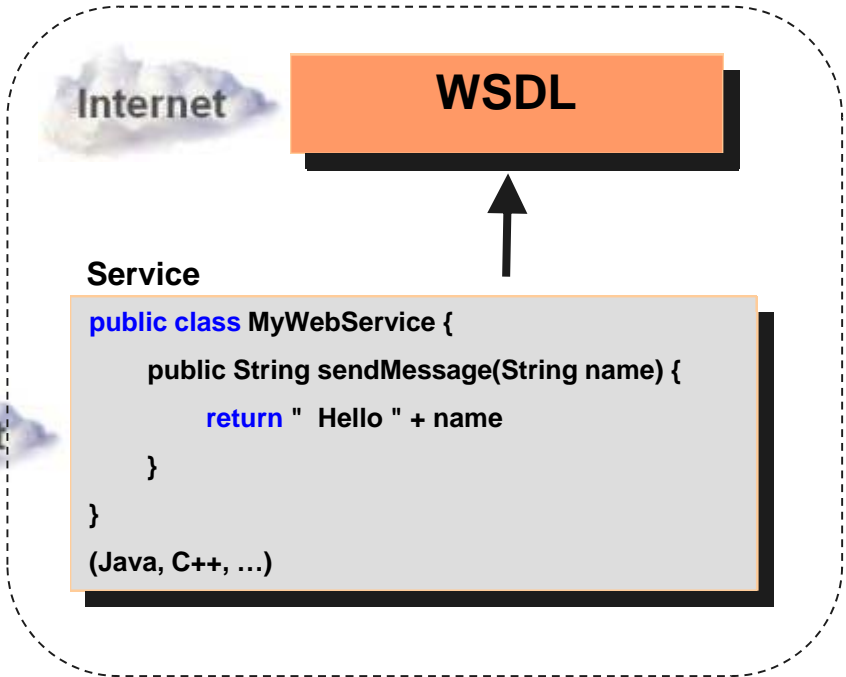
Service

```
public class MyWebService {  
    public String sendMessage(String name) {  
        return " Hello " + name  
    }  
}  
(Java, C++, ...)
```

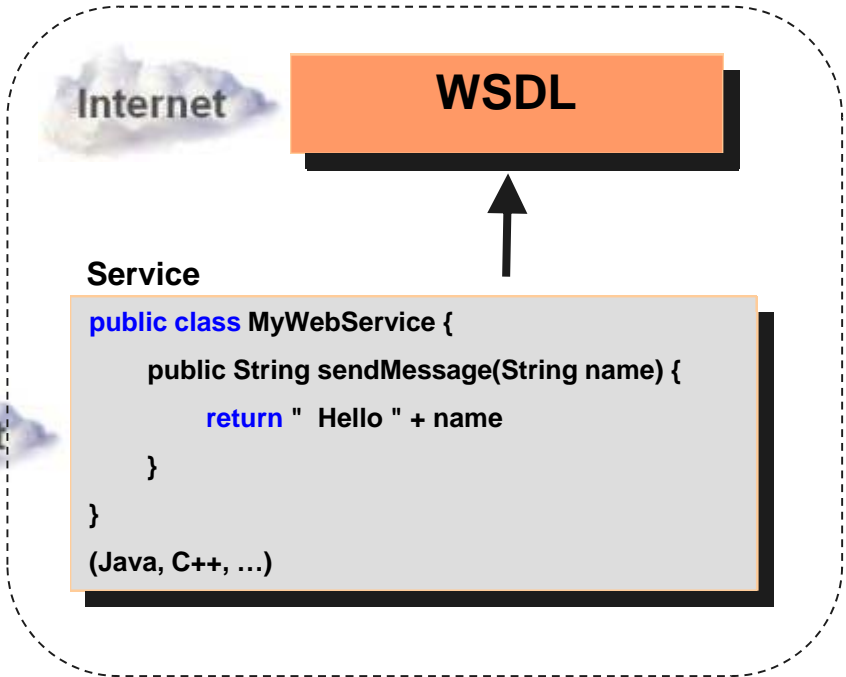
Cinématique générale



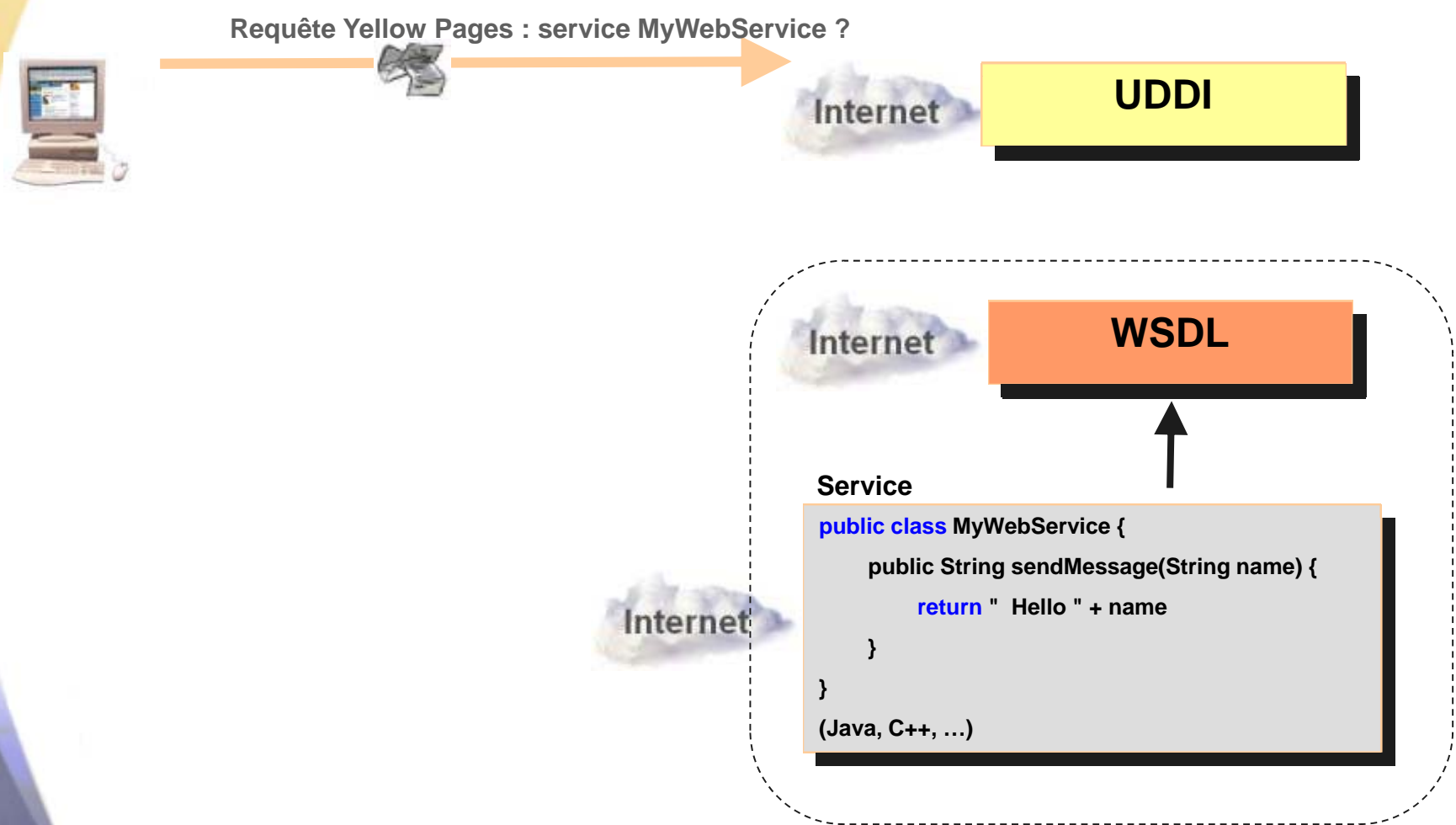
Cinématique générale



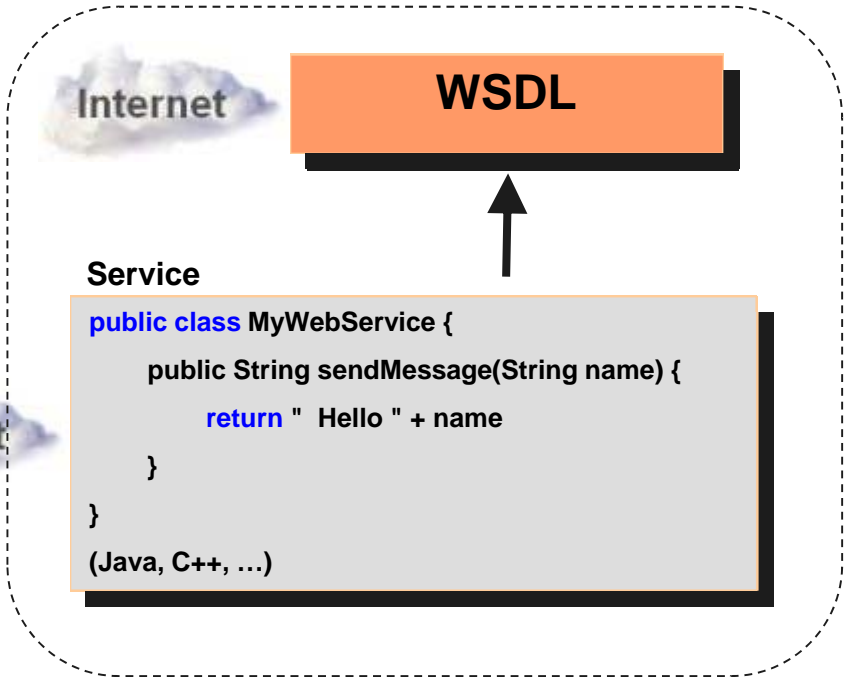
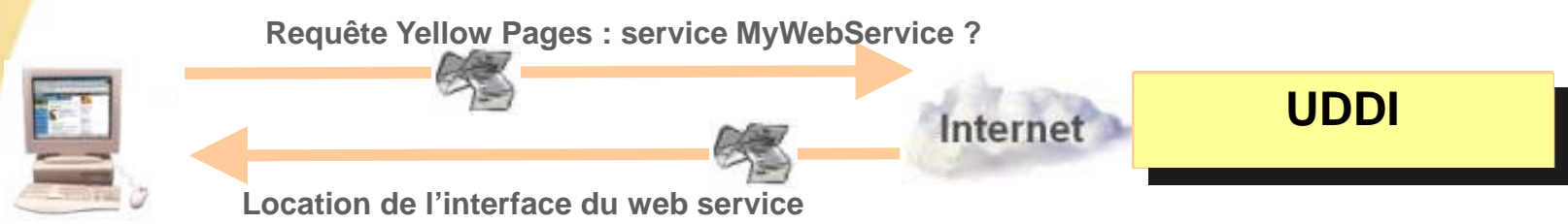
Cinématique générale



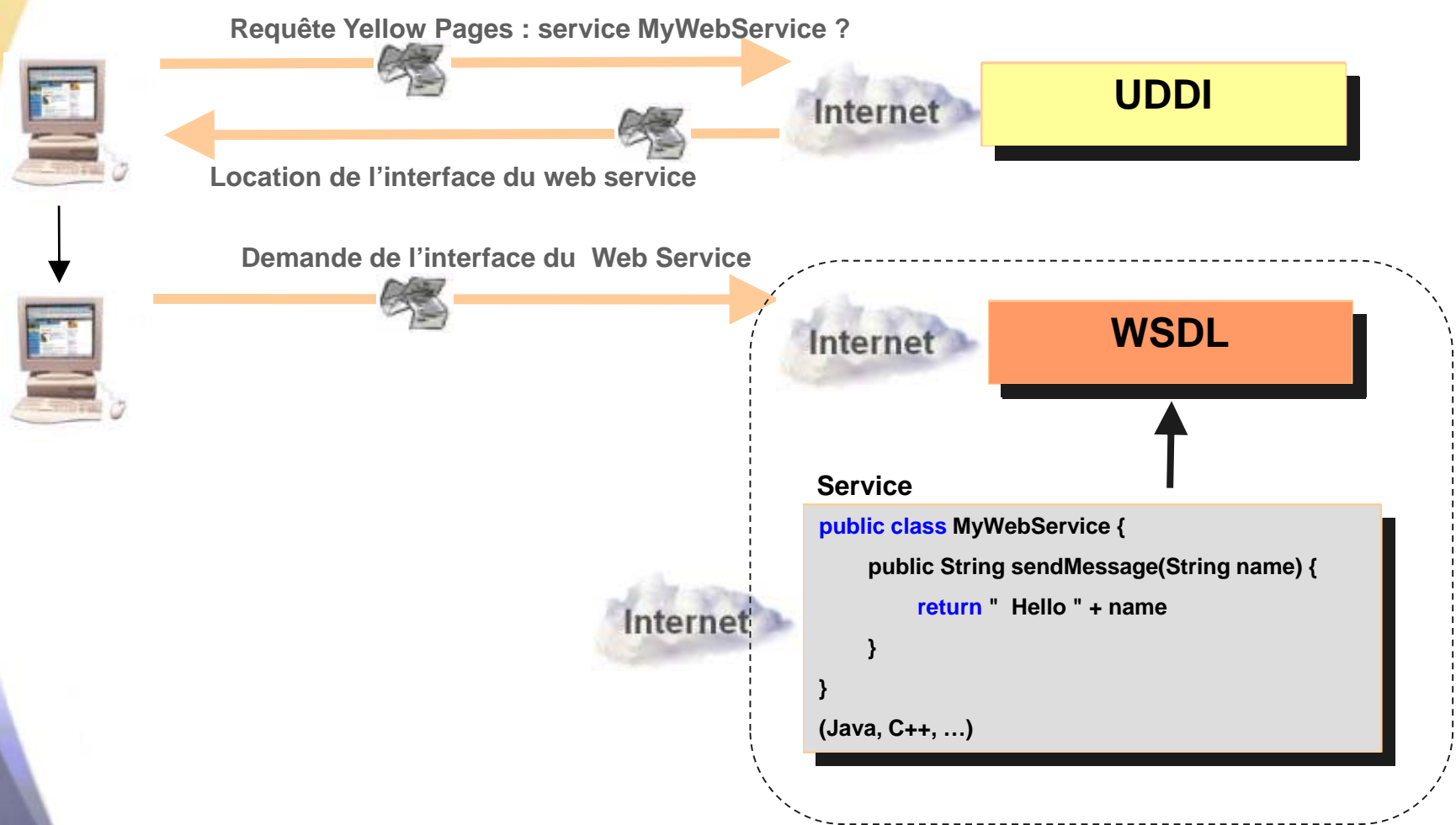
Cinématique générale



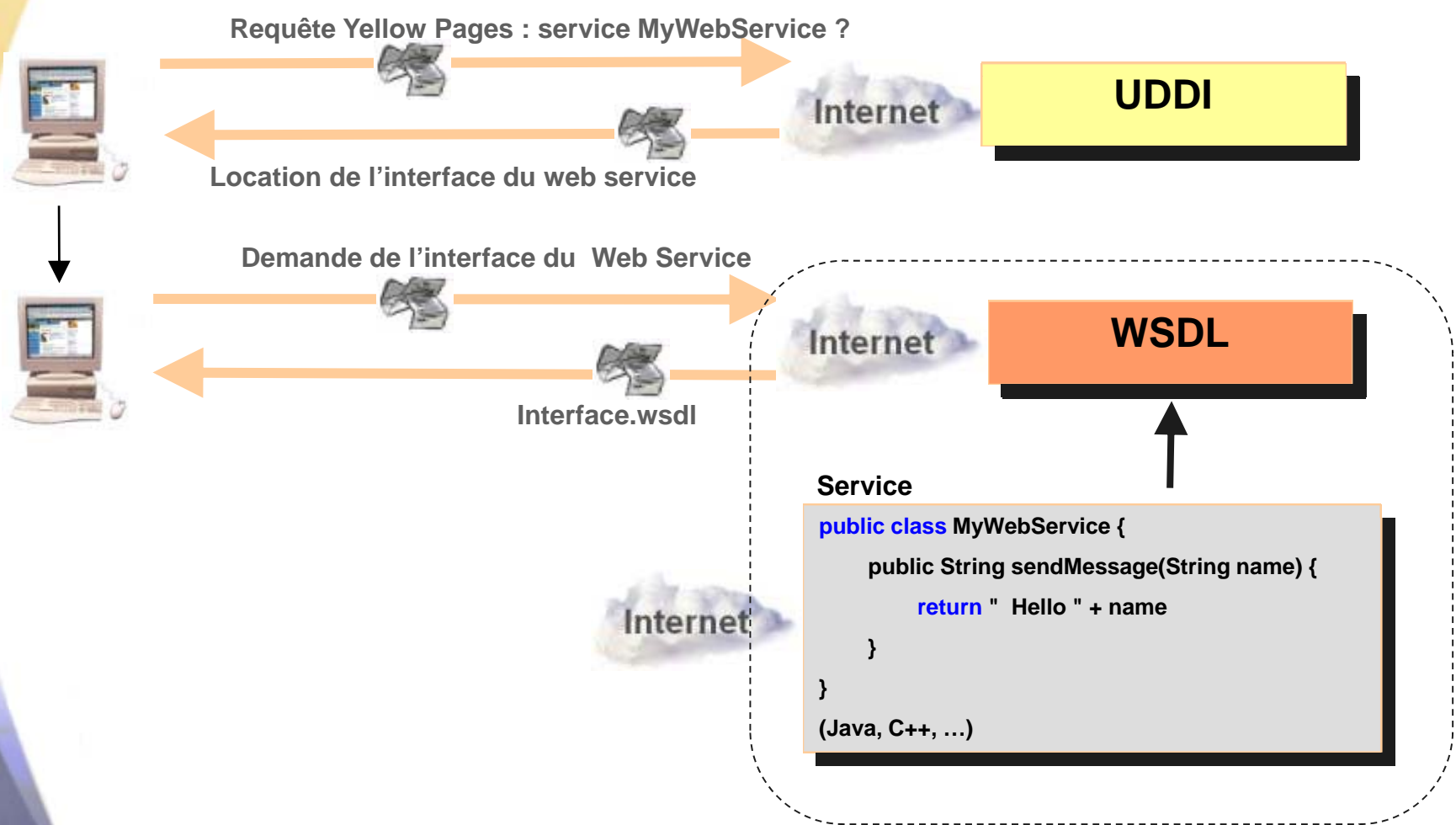
Cinématique générale



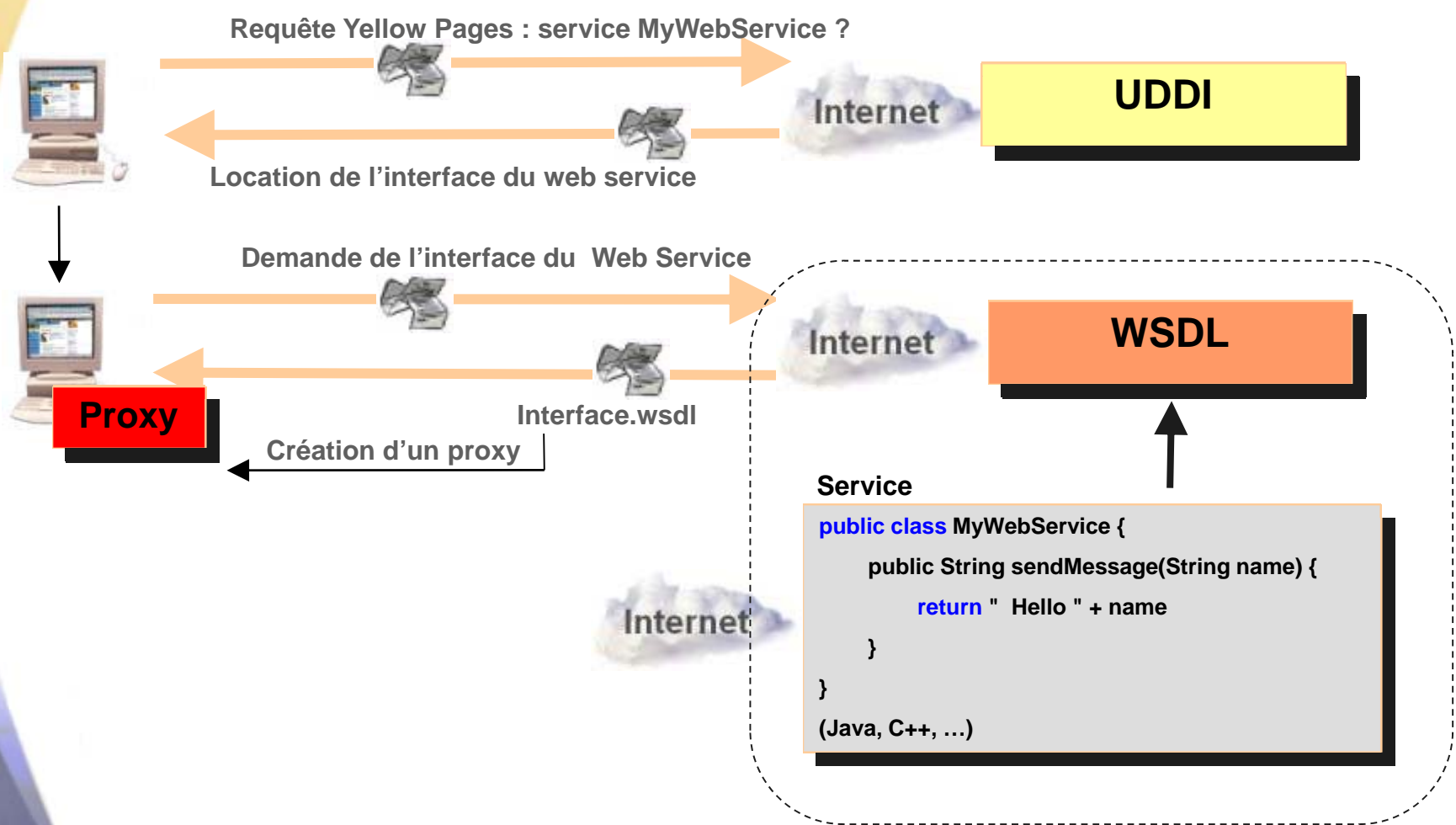
Cinématique générale



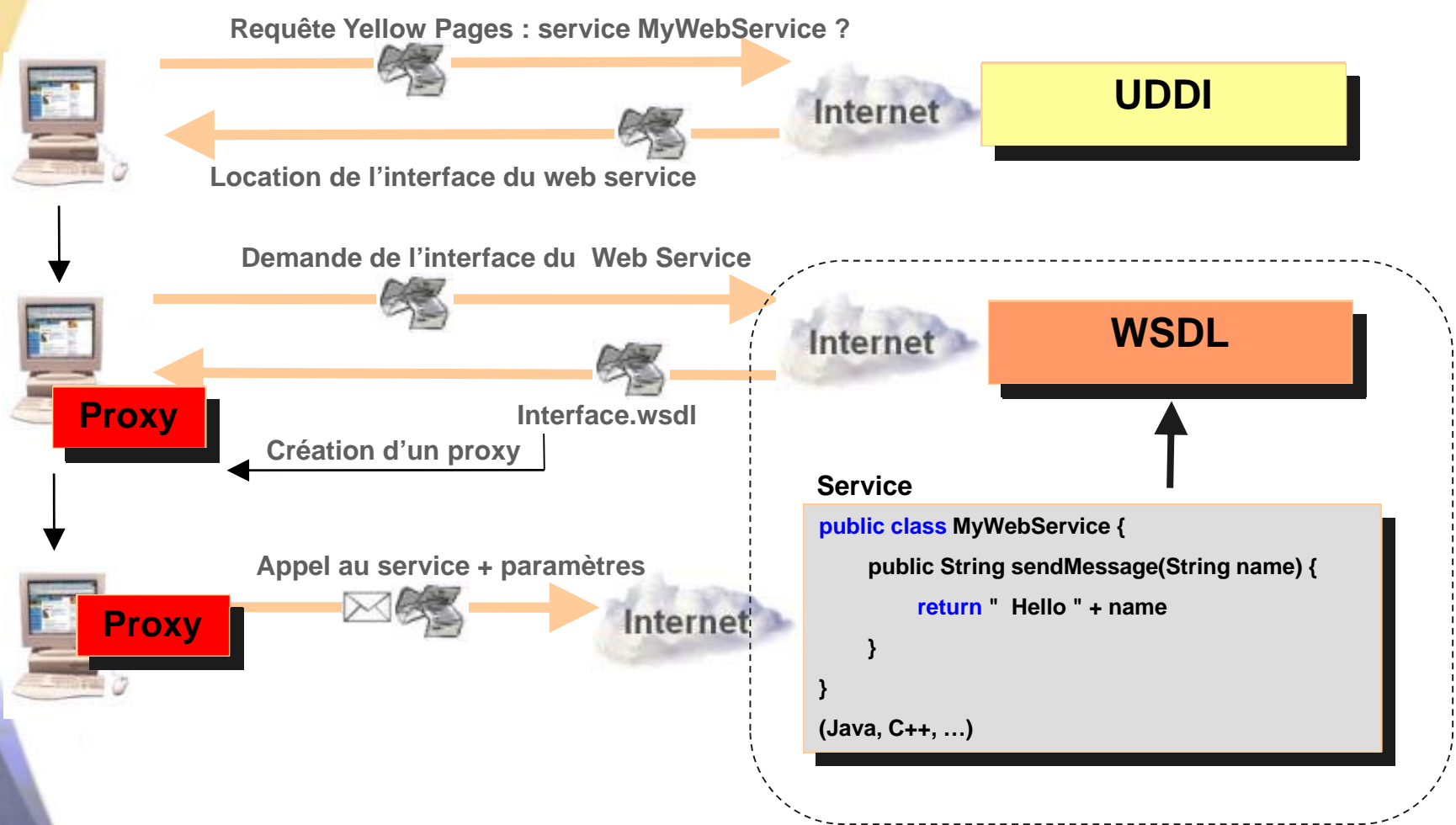
Cinématique générale



Cinématique générale



Cinématique générale



Cinématique générale

