Exercice 1 : Services en ligne sur la toile : 'Web Services'

Les services sur la toile ' **WEB Services '** forment une nouvelle approche de communication de programme à programme entre des applications clientes et des services en ligne sur la toile mondiale (WWW ' World Wide Web '). L'objectif technique de base des services sur la toile, est d'améliorer les communications entre des programmes fonctionnant sur des sites clients distribués dans l'Internet mondial et des applications serveuses. Les domaines d'application visés ne sont pas limités a priori mais les concepteurs ont pensé principalement au cas du commerce électronique (vente en ligne de produits et services).

Il s'agit en fait de la définition d'un nouveau type d'intergiciel ('middleware') client serveur qui se place en rival de propositions comme CORBA, DCOM, ou Java RMI. Comme tout intergiciel les services WEB sont définis avec un outil central qui est le protocole de communication et de nombreux outils complémentaires qui traitent des nombreux problèmes annexes (annuaire réparti, sécurité, transactionnel, coordination, …).

Les deux idées essentielles à la base de cette nouvelle approche sont d'une part de faire réaliser les communications entre applications en utilisant le protocole HTTP et d'autre part de spécifier toutes les normes (structures de données, formats de messages) en utilisant le langage XML.

Cette approche est soutenue activement par de très nombreux éditeurs de logiciels avec des acteurs importants comme **Microsoft ou IBM**. Les entreprises intéressées par les services web et leur processus de normalisation sont réunies au sein de consortiums dont le plus important est le **W3C** ('World Wide Web Consortium').

SOAP ('Simple Object Access Protocol') est le protocole de **communication** des services Web. Il définit deux modes de communications entre programmes : un protocole de communication en mode **message** et un protocole de communication en **mode appel de procédure distante** (**RPC**). SOAP utilise **XML** pour représenter les données échangées. Les normes SOAP sont généralement décrites en supposant l'utilisation de **HTTP** comme protocole sous-jacent (pour acheminer les messages SOAP) mais il est aussi indiqué dans les documents SOAP que des protocoles comme **SMTP** ou d'autres protocoles comme des **MOM** peuvent également être utilisés.

1) SOAP a pour objectif de fournir un mode de communications par messages asynchrones et un mode de communication par RPC. Rappelez les grandes lignes de ces catégories de mode de communication ?

2) Quel est l'usage actuel de HTTP. Soap propose d'utiliser HTTP comme un moyen de communications entre programmes. Quels avantages et quels inconvénients voyez vous à la réalisation de communications entre programmes en utilisant le protocole HTTP ?

3) On rappelle que le protocole HTTP est présenté dans ses RFC comme offrant une communication au niveau des objets. HTTP utilise dans son vocabulaire la notion de méthodes. Quelles sont les différentes méthodes offertes en http ?

4) HTTP vous semble t'il correspondre à un protocole d'appel de procédure distante (justifiez votre réponse) ?

5) Les concepteurs de SOAP ont choisi XML comme outil de représentation de la structure des messages. Quel est le principal langage, utilisé avant XML, pour

représenter les différents types de données manipulés par des programmes et échangés par messages ?

6) Quels sont les avantages et quels sont les inconvénients de l'utilisation de XML pour représenter la structure des messages échangés en SOAP ?

Dans le fonctionnement de SOAP en mode RPC, voici un exemple de message d'appel (extrait du document W3C SOAP primer). L'objectif de l'échange est de confirmer une réservation précédemment faite et d'effectuer le paiement.

```
POST /Reservations HTTP/1.1
Host: travelcompany.example.org
Content-Type: application/soap+xml; charset="utf-8"
Content-Length: nnnn

<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
  <env:Header>
  </env:Header>
  <env:Body>
    <m:chargeReservation
      env:encodingStyle="http://www.w3.org/2003/05/soap-encoding"
      xmlns:m="http://travelcompany.example.org/">
      <m:reservation xmlns:m="http://travelcompany.example.org/reservation">
      <m:code>FT35ZBQ</m:code>
      </m:reservation>
      <o:creditCard xmlns:o="http://mycompany.example.com/financial">
      <n:name xmlns:n="http://mycompany.example.com/employees">
        Åke Jógvan Øyvind
      </n:name>
      <o:number>123456789099999</o:number>
      <o:expiration>2005-02</o:expiration>
      </o:creditCard>
    </m:chargeReservation>
  </env:Body>
</env:Envelope>
```

Le message résultat correspondant au message d'entête précédent a la forme suivante (la plupart des détails sont omis pour limiter le volume de l'exemple) :

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset="utf-8"
Content-Length: nnnn

<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope" >
  <env:Header>
   ...
```

```
  ...
 </env:Header>
 <env:Body>
  ...
  ...
 </env:Body>
</env:Envelope>
```

7) A quoi correspondent les cinq premières lignes du message d'appel. Commentez les informations qui se trouvent dans chacune des cinq lignes ?

8) Le document XML qui encode l'appel est composé d'une entête vide et d'un corps qui contient le nom de la méthode à exécuter et les paramètres d'appel. Quel est l'arbre résultant de l'analyse syntaxique du document précédent ?

L'exemple suivant montre un fonctionnement de SOAP avec SMTP.

```
From: a.oyvind@mycompany.example.com
To: reservations@travelcompany.example.org
Subject: Travel to LA
Date: Thu, 29 Nov 2001 13:20:00 EST
Message-Id: <EE492E16A090090276D208424960C0C@mycompany.example.com>
Content-Type: application/soap+xml

<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
 <env:Header>
  ...
 </env:Header>
 <env:Body>
  ...
 </env:Body>
</env:Envelope>
```

9) Expliquez comment fonctionne le protocole soap avec SMTP (est ce que reservations@travelcompany.example.org est le nom du programme qui exécute l'appel) ?

10) Quel(s) avantage(s) peut-on trouver à l'utilisation de SMTP ?

11) WSDL ('Web Service Definition Language') est le langage de définition d'interfaces des services web (IDL 'Interface Definition Language'). Rappelez le rôle d'un tel langage dans le cadre d'un protocole de communication en approche objets répartis (par exemple dans le standard CORBA) ?

Comme tous les outils définis dans le cadre des services sur la toile WSDL est un dialecte XML. La structure d'une déclaration WSDL est décrite dans ses grandes lignes comme suit.

WSDL commence par permette à un utilisateur de décrire de nouveaux types de données dans un langage baptisé xmlschéma (cette possibilité est décrite par les trois lignes suivante dans la descripttion générale de la structure d'un document WSDL)

```
<wsdl:types>
    <xsd:schema> ….</xsd:schema>
</wsdl:types>
```

WSDL propose ensuite de décrire la structure d'un ensemble de messages qui sont en fait des messages de requêtes et de réponse utilisés dans le cadre de RPC.

```
<wsdl:message name="nmtoken">
    <part name="nmtoken" element="qname"? type="qname"?/>
</wsdl:message>
```

Ensuite WSDL définit un élément type de port comme un ensemble d'opérations. Une opération est définie par un message d'appel (input) et un message résultat (output). Le standard prévoit aussi la description d'une structure de données associée aux erreurs.

```
<wsdl:portType name="nmtoken">
  <wsdl:operation name="nmtoken">
    <wsdl:input name="nmtoken"? message="qname">
    </wsdl:input>
    <wsdl:output name="nmtoken"? message="qname">?
    </wsdl:output>
    <wsdl:fault name="nmtoken" message="qname">
    </wsdl:fault>
  </wsdl:operation>
</wsdl:portType>
```

L'élément binding (liaison) définit ensuite concrètement comment accéder à un service en indiquant son type de port, le protocole utilisé pour y accéder (c'est principalement Soap en mode HTTP mais on a vu que d'autres possibilités sont prévues) et la localisation dans la toile de ce service (URI de l'endroit ou se trouve le service web).

```
<wsdl:binding name="nmtoken" type="qname">
  <wsdl:operation name="nmtoken">
    <wsdl:input>
    </wsdl:input>
    <wsdl:output>
    </wsdl:output>
    <wsdl:fault name="nmtoken">
    </wsdl:fault>
  </wsdl:operation>
</wsdl:binding>
```

Finalement le service rendu sur le Web par une entreprise peut être composé de plusieurs ports c'est à dire de plusieurs applications accessibles sur le web (correspondant à des liaisons 'bindings) que nous venons de voir) et donc localisées éventuellement sur des serveurs différents.

```
<wsdl:service name="nmtoken">
  <wsdl:port name="nmtoken" binding="qname">
  </wsdl:port>
</wsdl:service>
```

12) Comparez ce langage de définition d'interface à celui de Corba.
Suggestions pour construire la réponse :
Pourquoi décrire des types de données (à quoi correspond cette possibilité en IDL CORBA) ?

Pourquoi décrire des types de ports qui regroupent des opérations (à quoi correspond cette possibilité en IDL CORBA) ?
Pourquoi décrire des ports qui définissent des points d'accès concrets à des services (à quoi correspond cette possibilité en IDL CORBA) ?