Master 1, HMIN202
Web services : développement de services web avec .net

1) Création de service web avec C#

On va ici créer un service web simplissime avec .NET.
Notre service web permettra simplement de récupérer la date du jour (utiliser la classe *System.DateTime*).

- Dans la solution de votre choix, clic droit, ajouter → nouveau site web.
- Vous obtenez (entre autres) d'une part un répertoire pour le code applicatif (*App_Code*) contenant *Service.cs* (un genre de *HelloWorld*, qui contient des services identifiés par *[WebMethod]*), et d'autre part un fichier *.asmx* qui permet le déploiement de ce service, en précisant en quel langage il est écrit (ici *C#*), et dans quelle classe et quelle *assembly* se trouve la méthode à invoquer. Son code ressemble à :

  <%@ WebService Language="C#" CodeBehind="~/App_Code/Service.cs" Class="Service" %>

- Quand on démarre le projet (juste par un *run* du projet), le serveur de développement *ASP.net* se met en route, en local, et un navigateur s'ouvre sur la page *asmx*. Vous obtenez une interface de test de votre service web, vous pouvez aussi visualiser le *wsdl* qui est publié et le *SOAP* qui sera échangé. Jetez-y un coup d'œil.
- Définissez maintenant un service web qui fournit la date du jour.

2) Bibliothèque

Une bibliothèque souhaite informatiser son système d'informations pour permettre aux personnes le souhaitant de consulter à distance son catalogue, et aux abonnés de pouvoir entrer des commentaires sur des ouvrages, qui seront alors consultables également. Dans un premier temps, les réservations et les emprunts d'ouvrage resteront non-informatisés. On ne distinguera donc pas la notion d'ouvrage de la notion d'exemplaire.
Pour la première version, on considèrera un seul type d'ouvrage : le livre, dont les champs à stocker sont : son titre, son auteur, son *ISBN*, son nombre d'exemplaires, son éditeur. On ne réalisera que 2 méthodes de consultation : la recherche d'ouvrage par *ISBN* et par nom d'auteur. Chaque abonné se voit attribuer un numéro d'abonné (unique), et choisit un mot de passe. Ce numéro d'abonné et ce mot de passe lui sera demandé lors de l'ajout d'un commentaire. On ne s'occupera pas du cryptage de ce mot de passe, qui sera stocké et envoyé en clair. On gardera bien à l'esprit que le but n'est pas de créer une application de consultation réaliste, mais de comprendre la mise en œuvre d'applications réparties via les services web. En ce sens, vous ne devez pas vous attarder trop sur les mécanismes de recherche avancés par auteur, par exemple ne prenez pas en compte le fait qu'on peut mal connaitre l'orthographe du nom de l'auteur. On ne réalisera une interface graphique cliente que si l'on a fini le TP.

Question 1. Réfléchissez à la structure de votre application, que vous développerez ensuite en TP. Vous réaliserez au minimum un diagramme de classes. Essayez de faire une conception "propre", en évitant notamment le piège de la grosse classe serveur omnipotente.

Question 2. Développez une première version de votre application sans client ni objet distribué : vous la testerez avec un *Main* simulant le client. Vous veillerez à utiliser à bon escient des propriétés. Vous pouvez utiliser des tables de hachage (classe *Dictionary* de *System.Collections.Generic*). Cela vous permettra d'utiliser les indexeurs pour accéder aux éléments de votre table de hachage, documenté dans *Property→Item*.

Question 3. Mettez en œuvre la distribution en n'utilisant un service web. Cela a entre autres l'avantage de pouvoir développer un client en Java, et de réaliser facilement des clients légers.
Dans votre bibliothèque, proposez certains services comme service web (par exemple : la recherche de livre par *ISBN* et par auteur, et l'ajout de commentaire), et réalisez un client *C#* pour ces services. Pensez bien que les types échanges doivent être simples mais bien structures. Pensez aussi qu'il n'est pas raisonnable de prévoir que pour obtenir une information, on échange de nombreux messages entre le client et le serveur (pensez au temps que cela prendrait).

Question 4. Faites-en sorte que la gestion des abonnés et du catalogue soit une application cliente du serveur de bibliothèque, de manière à ce que plusieurs bibliothécaires puissent administrer la bibliothèque en même temps. Cela a entre autres l'avantage de pouvoir développer un client en Java, et de réaliser facilement des clients légers.
Dans votre bibliothèque, proposez certains services comme service web (par exemple : la recherche de livre par *ISBN* et par auteur, et l'ajout de commentaire), et réalisez un client *C#* pour ces services. Pensez bien que les types échanges doivent être simples mais bien structures. Pensez aussi qu'il n'est pas raisonnable de prévoir que pour obtenir une information, on échange de nombreux messages entre le client et le serveur (pensez au temps que cela prendrait).