Plan des cours

- Protocole HTTP et programmation serveur
- Architectures réparties
- Objets distribués
- Introduction aux services web
  - Définitions
  - SOA
  - Contrat de service
  - Conception d'une application orientée services
  - Aperçu des protocoles (WSDL, SOAP, UDDI)
  - Autres technologies
- Projet

Introduction aux Services Web

- Principes
  - Utilisation d'une partie des résultats de l'activité d'une application par une ou plusieurs autres applications, éventuellement réparties sur un réseau
  - On parle aussi de « middleware internet »
  - Ne pas confondre avec les services COS CORBA

Introduction aux Services Web

- Définition
  - Services entre applications
    - Réalisation de traitements par une application pour le compte d'une autre
    - Indépendamment des implémentations des applications
  - Utilisant les technologies du Web
    - Langages « XML-based »
    - Protocoles (en particulier HTTP)

Introduction aux Services Web

- Notions importantes
  - **Relation de service**, ou relation **client-prestataire**
  - Échanges de messages lors d'**actes de communication**
  - Définition du service par un **contrat de service**
  - Les **applications orientées services** permettent de constituer des **architectures orientées services**

Applications Orientées Services

- Peuvent jouer au moins l'un des deux rôles impliqués dans une relation de service
  - Client
    - Peut devoir découvrir le service grâce à un annuaire
    - Doit respecter l'interface du service définie dans le contrat de service
    - Peut devoir payer ou fournir un autre service en retour
  - Prestataire
    - Doit rédiger un contrat de service
    - Peut publier ce contrat dans un annuaire
    - Doit respecter le contrat, en termes d'interface et de résultats
    - Ne s'engage pas sur le modèle d'implémentation

Applications Orientées Services

- Éléments d'une prestation de service web
  - Informations
    - Résultats de l'application prestataire, transmis à l'application cliente sous forme de messages
    - Exemples
      - Résultats d'un calcul complexe (simulation numérique)
      - Interface entre client et données
      - Agrégation de résultats d'autres services…

Applications Orientées Services

- Éléments d'une prestation de service web
  - États
    - L'application prestataire gère les états et les changements d'états des ensembles de données
    - Les états peuvent être
      - Volatiles
      - Persistants
      - Durables
    - Un changement d'état devrait toujours être réversible

Applications Orientées Services

- Éléments d'une prestation de service web
  - Effets de bord
    - Interactions de l'application prestataire avec son environnement (dispositifs d'entrées-sorties)
    - Exemple
      Impression d'une facture
    - Les effets de bord sont toujours irréversibles

Applications Orientées Services

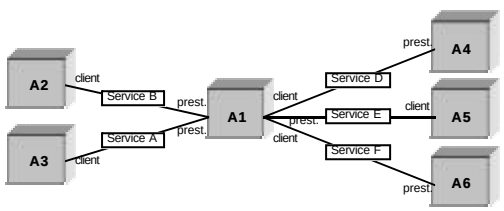
- Éléments d'une prestation de service web

Applications Orientées Services

- Définition
  modèle d'architecture pour l'exécution d'applications logicielles réparties
- Principe
  Architecture d'applications réparties qui participent à un « réseau d'échange de services »
- Remarques
  - Chaque application peut être écrite dans un langage différent
  - Il n'y a pas nécessairement de « démarche de conception » dans une SOA (à l'exemple du Web)

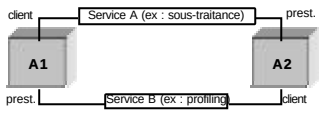
Applications Orientées Services

- Exemples de SOA
  - Fédération de services



Applications Orientées Services

- Exemples de SOA
  - Échange circulaire de services



Contrat de service

- Définition
  - Engagement qui formalise la relation de service
  - Toute application pouvant satisfaire les engagements du prestataire peut interpréter le rôle de prestataire
  - Idem pour le client

⇒ La réalisation de la prestation de service réside dans l'exécution du contrat

Contrat de service

- Rappel
  - Un service web doit être indépendant de la mise en œuvre des applications constituantes de la SOA
    - Langage de programmation
    - Modèle d'implémentation
  - Chaque application doit être capable d'accéder au contrat de service et de l'exploiter

⇒ Le contrat de service doit être rédigé dans un langage neutre (cf. IDL)

⇒ Tant qu'à faire, autant qu'il soit aussi lisible par des agents humains (⇒ XML)

Contrat de service

- Six principales parties
  - Identification des parties
  - Fonctions du service
  - Interface du service
  - Qualité du service
  - Cycle de vie du service et du contrat
  - Description des termes de l'échange

Contrat de service

- Identification des parties
  - Descriptions en termes d'acteurs humains
    - Management (équipe de gestion de projet)
    - Métier (domaine applicatif)
    - Informatique (correspondants fonctionnels, architectes, concepteurs, développeurs, exploitants)
  - Description en termes d'agents logiciels
    - Outils de développement (édition, génération de code, compilation…)
    - Application (client, prestataire…)
    - Outils d'exploitation (pilotage du service)

Contrat de service

- Identification des parties
  - Parties potentielles
    - Prestataires
    - Clients
    - Tiers (pas de relation directe avec le client)
    - Intermédiaires (entre le client et le prestataire)
  - Remarques
    - L'identification de toutes les parties n'est pas obligatoire
    - On parlera indifféremment d'acteurs humains ou logiciels

Contrat de service

- Fonctions du service
  - Les spécifications fonctionnelles sont données par
    - Les objectifs : à quoi il sert
    - Les actions : comment on s'en sert
    - Les informations et les règles de fonctionnement du service : connaissances permettant à l'application prestataire d'accomplir sa tâche

Contrat de service

- Fonctions du service
  - Résumé
    - Le système utilise les *informations* et les *règles* en sa possession pour sélectionner les *actions* qui lui permettent d'accomplir ses *objectifs*
  - Remarque
    - Le modèle fonctionnel n'est pas un modèle d'implémentation : le contrat de service n'indique pas *comment* le service est réalisé

Contrat de service

- Interface du service
  - Interface abstraite
    - Syntaxe abstraite
    - Sémantique
    - Pragmatique
  - Interface concrète
    - Styles d'échanges
    - Formats des messages
  - Liaisons
    - Conventions de codage
    - Protocoles de transport
  - Ports de réception
  - Chaînes d'acheminement

Contrat de service

- Interface du service
  - Interface abstraite
    - Représente l'ensemble des actes de communication entre client et prestataire, indépendamment des moyens utilisés
    - Syntaxe abstraite : description des composants d'un acte de communication
      - Type (nom) d'acte
      - Description abstraite du contenu de l'acte (message)
      - Direction de l'acte

Contrat de service

- Interface du service
  - Interface abstraite
    - Sémantique
      - Décrit l'association entre un acte de communication et l'action que l'émetteur accomplit
      - Fixe les conditions sémantiques de succès de l'acte
        - L'émetteur a la capacité, le pouvoir, le droit et l'autorisation d'émettre l'acte de communication
        - Le récepteur a la capacité, le pouvoir, le droit et l'autorisation de réceptionner, d'analyser et d'évaluer l'acte, ainsi que d'accomplir ses effets pragmatiques
        - L'acte est transmis dans un contexte où il est correct et pertinent

Contrat de service

- Interface du service
  - Interface abstraite
    - Pragmatique
      - Décrit les effets intentionnels de l'acte de communication sur le récepteur, en termes
        - de changements d'états
        - d'effets de bords
      - Décrit les situations d'erreurs et les conséquences des actes qui ne remplissent pas les conditions de succès ou de satisfaction

Contrat de service

- Interface du service
  - Interface abstraite
    - Pragmatique
      - Décrit les protocoles de conversation et processus métiers abstraits
        - Protocole de conversation : échange contractuel d'actes de communication
        - Processus métier abstrait : description d'un enchaînement d'actes de communication, de changements d'état et d'effets de bord (voir conclusion)

Contrat de service

- Interface du service
  - Interface concrète
    - Styles d'échanges
      - Message à sens unique (pas de réponse attendue)
      - Requête/réponse
        - synchrone du point de vue de la connexion
        - pas nécessairement bloquant au niveau du thread
        - ex : appel RPC (appel d'une méthode sur un objet distant)/données résultant de cet appel
      - Séquence de messages (Streaming)
      - Requête/réponse multiple : 1 requête et 1 séquence de réponses

Contrat de service

- Interface du service
  - Interface concrète
    - Formats des messages
      - Syntaxe (concrète) détaillée des messages SOAP

        Enveloppe (obligatoire)

        En-tête (facultative) : partie opérationnelle, pour l'infrastructure de traitement des messages

        Corps (obligatoire) : partie fonctionnelle, contenu de l'acte de communication

Contrat de service

- Interface du service
  - Liaisons
    - Conventions de codage
      - Définit un système de types de données simples et complexes
      - Utilise la syntaxe XML-Schema
      - Spécifie la façon dont sont codées les données dans les messages (pour éviter les erreurs d'interprétation entre systèmes de codage des applications)
    - Protocoles de transport
      - Fixe le ou les protocoles de transport utilisés pour chaque type de message

Contrat de service

- Interface du service
  - Ports de réception
    - Désignation du ou des adresses des prestataires via
      - Leurs URI + numéros de ports
      - Des mécanismes d'indirection permettant de découvrir ces adresses au moment de la réalisation de la prestation (ex : un annuaire)
  - Chaînes d'acheminement
    - Informations sur les éventuels intermédiaires
    - Peuvent être dynamiques
    - Doivent être incluses dans l'en-tête SOAP des messages

Contrat de service

- Qualité du service
  - Ensemble de propriétés opérationnelles liées à la réalisation de la prestation
    - Périmètre de la prestation (caractère optionnel de la prestation, exclusions, droits et obligations du client, conformité aux normes et standards…)
    - Qualité de fonctionnement (dimensionnement des objets manipulés, exactitude, précision, performance, accessibilité de l'application)
    - Sécurité (authentification, contrôle d'accès, confidentialité, intégrité, non-répudiation)
    - Robustesse (fiabilité, disponibilité, gestion d'arrêt de l'application prestataire, gestion des transactions)

Contrat de service

- Cycle de vie du service et du contrat
  - Service de gestion du cycle de vie du service primaire (activation, suspension, redémarrage, arrêt)
  - Service de pilotage du service primaire
  - Service d'interrogation de l'état du service primaire
  - Service de journalisation des activités du service primaire
- Description des termes de l'échange
  - Décrit si le service est gratuit, payant, troqué ou mixte

Introduction aux Services Web

- Les différents langages et protocoles
  - SOAP : messages échangés lors des actes de communication
  - WSDL : description du service pour le contrat de service
  - UDDI : annuaire des services disponibles pour la découverte dynamique et l'agrégation de services

Conception d'une AOS

- Différents types d'AOS
  - Agrégation
    - Plusieurs applications se répartissent les tâches et les publient sous forme de services
    - *Cf.* programmation modulaire
  - Dissémination
    - Décentralisation des données
    - Plusieurs applications sont prestataires du même service sur des données différentes

Conception d'une AOS

- Différentes approches de conception
  - Outside-in
    - On établit d'abord le contrat de service, à partir des besoins des clients potentiels
    - On cherche ensuite à utiliser au mieux les services disponibles
    - Avantage : pertinence ; inconvénient : faisabilité
  - Inside-out
    - On propose un service à partir de l'utilisation de services existants
    - Avantage : faisabilité ; inconvénient : adéquation aux besoins

⇒ La démarche communément suivie est de mener les deux approches en parallèle

Conclusion sur les services web

- Autres types d'AOS
  - Architectures dynamiques
  - Composition sémantique de services
- Autres langages/protocoles
  - WSCL : Web Services Conversation Language
  - BPML : Business Process Modeling Language
  - BPSS : Business Process Modeling Specification Schema
  - WSCI : Web Services Choreography Interface
  - BPEL : Business Process Execution Language
  - BPEL4WS : BPEL for Web Services

Autres technologies

- Remote Procedure Call (RPC)
  - Origine
    - Créé à l'origine pour le système de fichiers NFS
    - Antérieur à CORBA
  - Principe
    - Appel de fonctions distantes
    - Langages de programmation hétérogènes
    - Interface en IDL
    - Génération d'un stub et d'un skeleton
    - Version « originale » : Sun RPC
  - Remarques
    - Plusieurs versions, incompatibles entre elles
    - Implémentation sur HTTP : XML-RPC

Autres technologies

- REpresentational State Transfer (REST)
  - Origine
    - Thèse de Roy Fielding (IETF), 2000
  - Principes fondateurs
    - Une application serveur répond à une requête sur une URL en renvoyant une *représentation* de la ressource demandée
    - Chaque représentation place l'application cliente dans un état (*state*) donné
    - Pour chaque nouvelle représentation, l'application cliente change d'état (*state transfer*)

➔ Applications « restful »

Autres technologies

- REpresentational State Transfer (REST)
  - Principes opérationnels
    - Utilisation des différents standards existants
      - HTTP (utilisation de toutes les méthodes : GET, POST, PUT, DELETE, CONNECT…)
      - URI (doit suffire pour accéder à la ressource)
      - Types MIME
    - Simplicité de mise en œuvre
      - Pas de « surcouche » au dessus de HTTP (*vs* SOA et XML-RPC)
      - Pas d'état côté serveur
        - Moins de ressources consommées côté serveur
        - Plus de ressources consommées côté client
        - Augmentation de la bande passante consommée

Autres technologies

- Asynchronous Javascript And XML (AJAX)
  - Principe
    - Applications web avec interface utilisateur
    - Déporter un maximum de code sur le client
      - Réduction des ressources consommées côté serveur
      - Réduction de la bande passante réseau
  - Exemples d'utilisation
    - Google (mail, earth…)
    - Suggestions automatiques
    - Traitement de texte
  - Frameworks
    - openAjax (IBM) : Dojo, Rico ; Ruby / Ruby on Rails (RoR)
    - Plugin Eclipse : Rich Ajax Platform

Autres technologies

- Asynchronous Javascript And XML (AJAX)
  - Fonctionnement
    - Requête asynchrone au serveur dans une fonction javascript (déclenchée par un événement quelconque)
    - Transfert asynchrone de données en XML
    - Chargement d'un document XML (API DOM) dans un
      - Objet XMLHttpRequest (Mozilla)
      - Contrôle ActiveX XMLHTTP (IE)
    - Traitement dynamique côté client
      - Affichage (transformation XSLT)
      - Logique applicative (fonctions JavaScript dédiées)

Autres technologies

- Asynchronous Javascript And XML (AJAX)
  - Exemple de code : création d'un objet requête

```
var req = null;

function getRequest()
{
  if (window.XMLHttpRequest)
  {
     req = new XMLHttpRequest();
  }
  else if (typeof ActiveXObject != "undefined")
  {
     req=new ActiveXObject("Microsoft.XMLHTTP");
  }
  return req;
}
```

Safari / Mozilla

Internet Explorer

Autres technologies

- Asynchronous Javascript And XML (AJAX)
  - Exemple de code : chargement asynchrone

```
function GetDataUsingAJAX (HttpMethod, url, params, elt)
{
  if(req != null)
  {
     // méthode avec paramètres
     req.onreadystatechange = function() {stateChange(elt)};
     // méthode sans paramètre
     // req.onreadystatechange = stateChange;

     req.open(HttpMethod, url, true);
     req.setRequestHeader("Accept", "application/xml");
     req.send(params);
  }
}
```

Assiciation de la fonction de gestion de l'état

Autres technologies

- Asynchronous Javascript And XML (AJAX)
  - Exemple de code : gestion de l'état

```
function stateChange (elt)
{
  if(req.readyState == 4) {
     if (req.responseXML != null) {
        var docXML= req.responseXML;
     } else {
        var docXML= req.responseText;
        docXML=parseFromString(docXML);
     }
     var docXMLresult = traiteXML(docXML);
     var str = (new XMLSerializer()).serializeToString(docXMLresult);
     document.getElementById(elt).innerHTML += str;
  }
}
```

READY_STATE_COMPLETE

Autres technologies

- Asynchronous Javascript And XML (AJAX)
  - Exemple de code : transformation XSLT

```
//Après chargement asynchrone des documents XML et XSLT
function transform XSLT (XMLDoc, XSLDoc, id)
{
   if(XMLDoc == null || XSLDoc == null) {return}
   try {
      if (window.ActiveXObject)
      {
         var target = document.getElementById(id);
         target.innerHTML = xml.transformNode(xsl);
      }
```

Internet Explorer

Autres technologies

- Asynchronous Javascript And XML (AJAX)
  - Exemple de code : transformation XSLT

```
      } else if (window.XSLTProcessor) {
         var xsltProcessor = new XSLTProcessor();
         xsltProcessor.importStylesheet(xsl);
         var fragment = xsltProcessor.transformToFragment(xml,
document);

         var target = document.getElementById(id);
         target.appendChild(fragment);
         document.appendChild(target);
      }
   } catch (e) {
      return e;
   }
}
```

Safari / Mozilla

Autres technologies

- Asynchronous Javascript And XML (AJAX)
  - implémentation de la logique applicative
    - Programmation directe en JavaScript
      - Fonctionnalités limitées
      - Fait parfois un peu « fouillis »
    - Utilisation de frameworks
      - Programmation dans un autre langage
      - Génération du code JavaScript

Conclusion

- Dans ce cours, nous avons vu
  - Applications web (client-serveur) « simples »
  - Applications réparties
    - Middleware (CORBA, RMI, services web, REST)
    - Avec interface utilisateur (EJB + servlets/JSP, AJAX)
- Règles de bonnes pratiques
  - Modélisation de l'application (UML)
  - Séparation des couches
  - Conception de l'architecture répartie
  - Utilisation de design patterns

Références

- Services Web
  - W3C Web Services activity group
    http://www.w3.org/2002/ws/
  - Tutorial web services
    http://www.w3schools.com
- RPC
  http://www.crevola.org/francois/?content=articles&show=1
- XML-RPC
  http://www.xmlrpc.com/
  http://developpeur.journaldunet.com/tutoriel/php/020109php_xmlrpc.shtml

Références

- REST
  http://fr.wikipedia.org/wiki/Representational_state_transfer
  http://opikanoba.org/tr/fielding/rest
- AJAX
  http://fr.wikipedia.org/wiki/AJAX
  http://javascript.developpez.com/cours/#ajax
  Ajax en pratique, D. Crane, E. Pascarello, D. James, CampusPress, ISBN : 2-7440-2060-5