**Démo 1 :**

**Producteur WCF : Service Somme**

1. Création du projet Host (Application Console)
   1. Import de servicemodel -> WCF (Framework 3.0)
2. Ajouter un projet de librairie de classe => Bonne pratique : Définit le contrat du service et son implémentation (Pour changer de host)
   1. Interface du contrat (Les Tags + Mode d’encodage Doc Literal Wrapped ou RPC …)
   2. Implémentation du contrat (Pas besoin de préciser les tag WCF) => Implémentation indépendante de WCF
3. Lancement du Host (Program.cs)=> Instruction non bloquante, Chaque service aura un conteneur différents
4. Fichier de configuration de l’application (App.Config)
   1. Ouvrir le fichier XML (system.servicemodel)
   2. Ouvrir le fichier avec le plugin pour WCF et WPF pour VS05 CTP November
      1. A B C : Adresse, Binding, Contrat
         1. Copier implémentation du service à partir de program.cs
         2. Interface que l’on veut exposer (qui a les tags WCF)
         3. Présenter les bindings
         4. Présenter l’adresse
5. Lancer le service web et lancer IE pour montrer le contrat WSDL (Garder l’url du wsdl dans le presse papier)
   1. http://localhost:5001/OperationSomme?wsdl
   2. Montrer le mode d’encodage dans le WSDL

**Consommateur WCF : Service Somme (Proxy)**

1. Création du projet (Application Console)
   1. Add Service reference (Avec adresse du WSDL : <http://localhost:5001/OperationSomme?wsdl>, name = OperationSomme)
   2. Afficher program.cs
      1. Ajouter les using WCF & Proxy
   3. Lancer le client

**Démo 2 :**

**Producteur JAX-WS 2.1 : Service Somme**

1. Montrer le serveur d’application GlassFish (Ecoute sur le port 8282) et lancer le serveur
2. Création du projet (Web / Web Application) : Demo2\_ProducteurSomme, Sur le serveur glassfish 2.1
3. Ajouter un service web au projet : MySomme
   1. Tag Webservice
   2. Tag pour soap 1.2
   3. Pas besoin de tag sur les méthodes (exposées par défaut)
   4. Implémenter le service Somme
4. Compiler et déployer le service web sur serveur d’application glassfish
5. Afficher le contrat WSDL (http://localhost:8282/Demo2\_ProducteurSomme/MysommeService?wsdl)
   1. Mode d’encodage (Document / Literal)

**Consommateur JAX-WS 2.1 du Service Somme WCF**

1. Création du projet General / Java Application
2. Add New Service Client (Génération du proxy : http://localhost:5001/OperationSomme?wsdl)
3. Drag and Drop pour la génération de l’appel du service web
4. Exécution

**Consommateur WCF du Service Somme JAX-WS 2.1**

1. Création du projet (Application Console)
   1. Add Service reference
      1. Avec adresse du WSDL : http://localhost:8282/Demo2\_ProducteurSomme/MysommeService?wsdl
      2. name = MySommeService
   2. Afficher program.cs
      1. Ajouter les using WCF & Proxy
   3. Lancer le client

**Démo 3 :**

**WS-I**

Montrer le site : <http://ws-i.org/deliverables/workinggroup.aspx?wg=testingtools>

* 1. **WS-I Monitor** entre le Client WCF Somme et le Serveur WCF Somme
     1. Lancer le serveur d’application de la Demo1
     2. Modifier le Client WCF pour utiliser le Proxy de Monitor (9000->5001)
     3. Modifier le fichier : monitorConfig.xml
     4. Lancer le programme Monitor
        1. >Monitor.bat -config monitorConfig.xml
     5. Fermer Monitor (En tapant Exit)
     6. Afficher le résultat C:\wsi-test-tools-Java\java\bin\log.xml
  2. **WS-I Validator** avec les traces de Monitor et le WSDL du serveur WCF Somme
     1. Modifier le fichier analyzerConfig.xml (Montrer en parallèle le contrat WSDL : <http://localhost:5001/OperationSomme?wsdl>)
        1. Utilise les traces de Monitor
        2. Utilise le contrat WSDL
     2. Lancer le Validator
        1. Analyzer.bat -config analyzerConfig.xml
     3. Afficher le résultat : C:\wsi-test-tools-Java\java\bin\report.xml
     4. WCF Respecte bien le BP 1.1
  3. **Log WCF** : Activer les logs de WCF (Faire un Appel de client WCF)
  4. Visualiser les log de WCF avec SvcTraceViewer

C:\Program Files\Microsoft SDKs\Windows\v6.0\Bin\SvcTraceViewer.exe

**Démo 4 :**

1. **Producteur WCF** : Services Complexes ( Tableau de int, Dictionary, Classe)
   1. Interface : Présenter les types échangés. Expliquer les tags DataContract et DataMember (DataContractSerilizer)
   2. Implémentation
   3. App.Config : Adresse, Binding, Contrat
2. **Consommateur JAX-WS** des Services Complexes WCF
3. **Producteur JAX-WS**: Services Complexes ( Tableau de int, HashMap, Classe)
4. **Consommateur WCF** des Services Complexes JAX-WS
   1. Problème avec HashMap -> Impossible à consommer

Essayer de le consommer en JAX-WS => Comme en WCF on ne peut rien faire avec l'objet mappé (pas d'accesseurs aux données)

**Démo 5:**

1. **Producteur WCF** : Services avec Exceptions (Fault de base, MyException, Exception non déclarée)
   1. **Interface**
   2. **Implémentation**
   3. **Host**
   4. **App.Config (A,B,C)**
   5. **Exécution**
   6. Montrer le WSDL avec les SOAP Fault
2. **Consommateur JAX-WS** des Services avec Exceptions WCF
   1. Nouveau projet Java Application (Demo5\_ConsommeWCF)
   2. Nouveau proxy : OperationsAvecExceptions
3. **Producteur JAX-WS**: Services avec Exceptions(Fault de base, MyException, Exception non déclarée)
   1. Nouveau projet Web / Web Application
   2. Les Tags Fault
   3. Compiler & Deployer
   4. Montrer le contrat WSDL
4. **Consommateur WCF** des Services avec Exceptions JAX-WS
   1. Problème avec MyException-> Mappage incorrect au niveau de WCF
   2. Essayer de consommer MyException avec jax-ws => on a une exception mappé correctement (mais hérite de exception)