Programmation
.NET Remoting

Didier DONSEZ

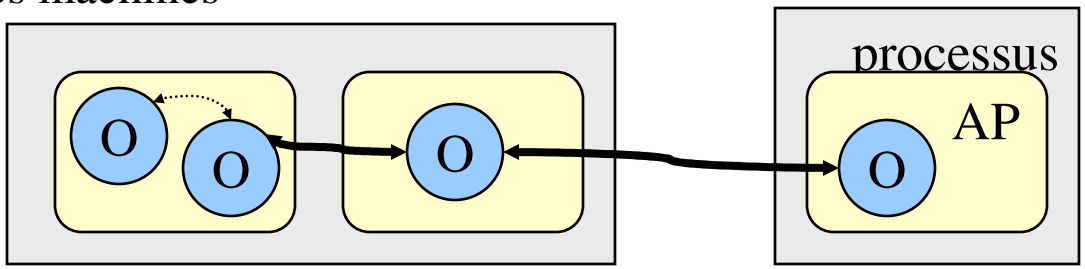
Université Joseph Fourier (Grenoble 1)

IMA – LSR/ADELE

Didier.Donsez@imag.fr, Didier.Donsez@ieee.org

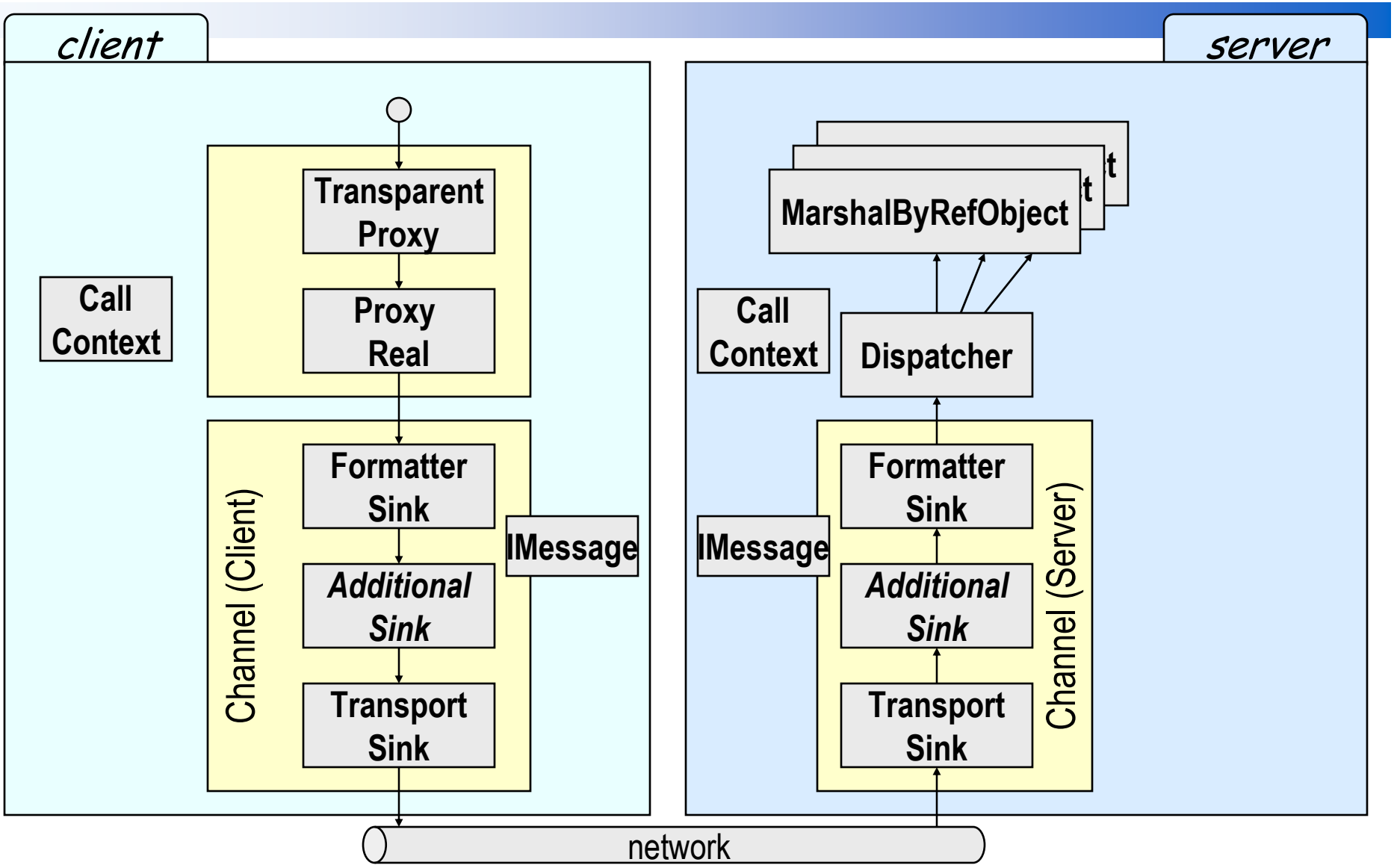
.NET Remoting

- Motivation
  - Invocation de méthodes sur des objets « distants »
    - Semblable aux Java RMI
  - Local
    - dans le même Application Domain
  - Distant
    - dans des Application Domain différents, dans des processus différentes, dans des machines



- Architecture
  - Channel, Sink, Formatter, Proxy, Nommage, IMessage, CallContext, Lease (Bail), Sponsor

Architecture .NET Remoting



Types d'objets distants

- Implémente MarshalByRefObject
- Well-Known
  - Identifié par une référence globale
  - 2 types
    - Singleton : une seule instance pour toutes les invocations
      État global partagé par tous les clients
    - SingleCall : une nouvelle instance à chaque invocations
      Sans état (lié à un client) *Stateless*
- Client Activated
  - Une instance par client
    - Avec un état (lié à un client) *Stateful*
    - Paramêtres d'instanciation donné par le client
    - Requiert un bail (lease) pour le GC de ce type d'objets

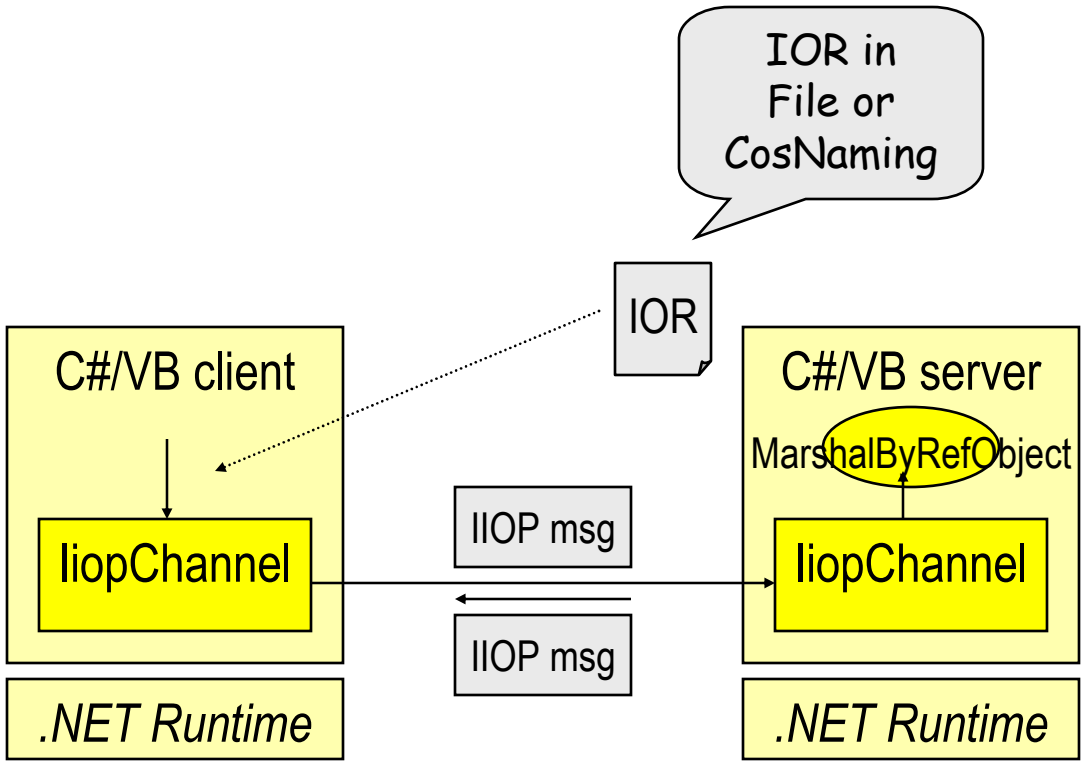
Channel

- Transport des requêtes/réponses
  - TcpChannel
    - Formatter binaire (semblable à DCOM) O-RPC
    - Transport Socket
  - HttpChannel
    - Formatter SOAP
    - Transport HTTP (HTTPS)
  - IpcChannel (depuis .NET2)
    - Transport en mémoire partagée
      pour 2 processus colocalisés sur la même machine
- Autres channels
  - IiopChannel: messages au format IIOP
    - http://remoting-corba.sourceforge.net/
    - http://www.dotnetguru.org/articles/articlets/iiopchannel/iiopremoting.htm
  - JrmpChannel
    - DotNetJ

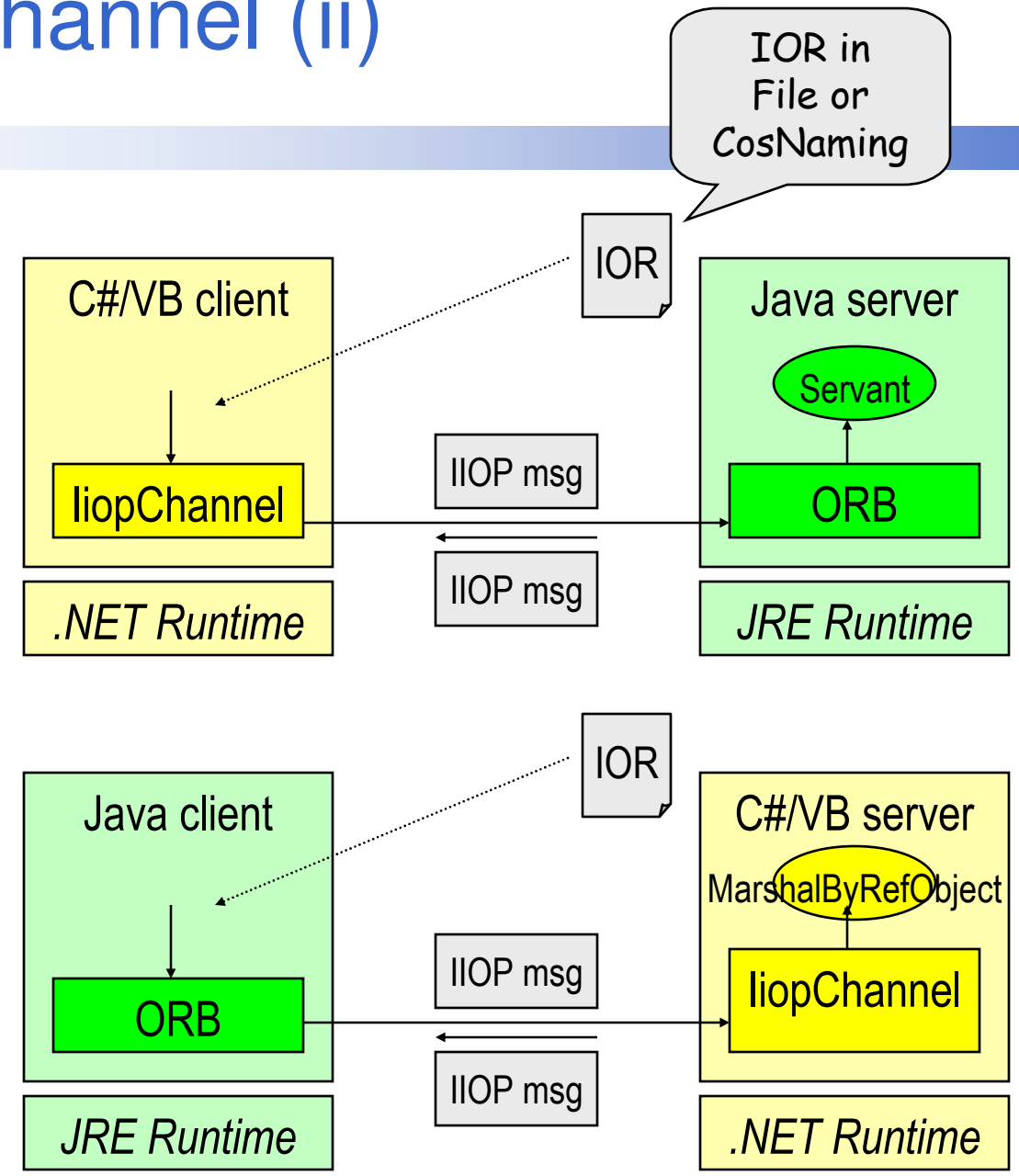
IIOPChannel

- IiopChannel: messages au format IIOP
- Interoperabilité
  - Java (J2SE, J2EE)
- 3 implémentations non MS
  - IIOP.NET
    - http://iiop-net.sourceforge.net/
  - Janeva
    - http://www.borland.com/janeva/
  - Remoting.Corba
    - http://remoting-corba.sourceforge.net/
    - http://www.dotnetguru.org/articles/articlets/iiopchannel/iiopremoting.htm

IiopChannel (i)



IiopChannel (ii)



IIOP.NET
http://iiop-net.sourceforge.net/

- Fournit
  - Un implémentation de IIOPChannel
  - Un compilateur d'IDL vers CLS
  - Un générateur CLS vers IDL
- Voir
  - Building a Distributed Object System with .NET and J2EE Using IIOP.NET, By Patrik Reali
    - http://www.codeproject.com/csharp/dist_object_system.asp
  - Accessing an EJB from .NET Using IIOP.NET: an Example, By Patrik Reali
    - http://www.codeproject.com/csharp/iiop_net_and_EJB.asp

Désignation des objets distants

- URL
  - protocole://hote:port/nom
    - protocole désigne une des Channel enregistrés : tcp, http, https…

- Remarque :
  - Pas de service distant de nommage/trading
    - CosNaming, rmiregistry (JNDI), …

Configuration

- Motivation
  - Enregistrement des Channels
  - Activation descriptive des objets

```
<wellknown mode="SingleCall"
    type="Cours.Remoting.HelloService, HelloService"
    objectUri="HelloSingleCall"/>
<activated type="Cours.Remoting.HelloService, HelloService"/>
```

- Fichiers de configuration
  - Un pour le client HelloServer.exe.config
  - Un pour le serveur HelloServer.exe.config
  - Un fichier global à la machine

    `%SystemRoot%\Microsoft.NET\Framework\<vx.x.x>\CONFIG\machine.config`
- Méthode
  - RemotingConfiguration.Configure("HelloServer.exe.config");

Exemple de configuration par fichier pour un objet Client-Activated

Client

```
<configuration>
 <system.runtime.remoting>
  <application name="HelloClient">
   <client url="tcp://localhost:8321/Cours/HelloWorld/ClientActivated">
    <activated type="Cours.Remoting.HelloServicePolyglot, HelloService" />
   </client>
   <channels><channel ref="tcp client" /></channels>
  </application>
  </system.runtime.remoting>
</configuration>
```

Serveur

```
<configuration>
 <system.runtime.remoting>
  <application name="Cours/HelloWorld/ClientActivated">
   <service>
    <activated type="Cours.Remoting.HelloServicePolyglot, HelloService" />
   </service>
   <channels><channel ref="tcp server" port="8321" /></channels>
  </application>
 </system.runtime.remoting>
</configuration>
```

Extrait de machine.config

```
<channel id="tcp client" type="System.Runtime.Remoting.Channels.Tcp.TcpClientChannel, System.Runtime.Remoting,
      Version=1.0.3300.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" />
<channel id="tcp server" type="System.Runtime.Remoting.Channels.Tcp.TcpServerChannel, System.Runtime.Remoting,
      Version=1.0.3300.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" />
```

Exemple Well-Know le service

```
namespace Cours.Remoting {
 public class HelloService
      : MarshalByRefObject {
  protected string _hellostr;
  public HelloService(){
    _hellostr="Hello";
  }
  public virtual string sayHello(){
       return _strhello+" World!";
  }
  public virtual string sayHello(string name) {
    return _strhello+" "+name+"!";
  }
  public virtual string sayHello(Person person){
   return _strhello+" "+person.GetFormatted()+"!";
  }
 }
}
```

```
// une classe d'objet serialisable
[Serializable]
public class Person  {
  private string _firstName;
  private string _lastName;
  public Person(string firstname,
                string lastname) {
    _firstName=firstname;
    _lastName=lastname;
  }
  public String GetFormatted() {
    return _firstName+" "+_lastName;
   }
}
```

Exemple Well-Know : le serveur

```
public class HelloServer {
  public static int Main(string [] args) {
   // Create an instance of a channel
   ChannelServices.RegisterChannel(new TcpChannel(8123));
    // Register as two available services
    System.Console.WriteLine("Register SingleCall");
    RemotingConfiguration.RegisterWellKnownServiceType(
       typeof(Cours.Remoting.HelloService),
       "Cours/HelloWorld/SingleCall",
       WellKnownObjectMode.SingleCall
     );
    System.Console.WriteLine("Register Singleton");
    RemotingConfiguration.RegisterWellKnownServiceType(
       typeof(Cours.Remoting.HelloService),
       "Cours/HelloWorld/Singleton",
       WellKnownObjectMode.Singleton
     );
    System.Console.WriteLine("Press the enter key to
    exit...");System.Console.ReadLine();
}
```

Type des objets à instancier

Désignation

Une instannce par appel
Une instance partagée

Exemple Well-Know : le client (i)

```
public class HelloClient {
 public static int Main(string [] args){
  // Create a channel for communicating w/ the remote object
  ChannelServices.RegisterChannel(new TcpChannel());
  // SingleCall Test
  obj= (HelloService) Activator.GetObject(
          typeof(Cours.Remoting.HelloService),
          "tcp://localhost:8123/Cours/HelloWorld/SingleCall"
   );
  if( obj.Equals(null) ){ System.Console.WriteLine("Error: unable to locate
  server");
  } else {
   System.Console.WriteLine("SingleCall Test");
   Person person=new Person("Didier","DONSEZ");
     for (int i=0; i < 3; i++) {
          Console.WriteLine(obj.sayHello());
          Console.WriteLine(obj.sayHello("Didier"));
          Console.WriteLine(obj.sayHello(person));
               // le paramêtre est un objet sérialisé
     }}
```

Exemple Well-Know : le client (ii)

```
// Singleton Test
    obj= (IHelloService) Activator.GetObject(
            typeof(Cours.Remoting.HelloService),
            "tcp://localhost:8123/Cours/HelloWorld/Singleton"
     );
    if( obj.Equals(null) ){
      System.Console.WriteLine("Error: unable to locate server");
    }else{
      Person person=new Person("Didier","DONSEZ");
      System.Console.WriteLine("Singleton Test");
         for (int i=0; i < 3; i++) {
                Console.WriteLine(obj.sayHello());
                Console.WriteLine(obj.sayHello("Didier"));
                Console.WriteLine(obj.sayHello(person));
                        // le paramêtre est un objet sérialisé
         }
    }
```

Exemple de Client Activated Le Service

```
public class HelloServicePolyglot {
 private string _language;

 public HelloServicePolyglot(string language){
   _language=language;
 }

 public string Language{
    get      { return _language; }
    set { this._language=value; }
 }

 public string sayHello() {
  return strHello()+" "+strWorld()+"!";
 }
 public string sayHello(string name) {
  return strHello()+" "+name+"!";
 }
 public string sayHello(Person person){
  return strHello()+"
    "+person.GetFormatted()+"!";
 }
```

```
private string strHello(){
 switch(_language){
    case "en-us": return "Hi";
    case "fr": return "Bonjour";
    case "es": return "Hola";
    case "it": return "Bongiorno";
    case "ru": return "Добрый день";
    default: return "Hello";
  }
 }
 private string strWorld(){
     switch(_language){
        case "fr": return "tout le monde";
        case "es": return "a todos";
        case "it": return "a tuti";
        case "ru": return "Весь мир";
        default: return "World";
     }
  }
 }
}
```

Exemple de Client Activated
Le Client

```
RemotingConfiguration.Configure("HelloClient.exe.config");
object[] constr = new object[1];
object[] attrs = { new UrlAttribute("tcp://localhost:8123/Cours/HelloWorld/ClientActivated") };
constr[0] = "fr";
HelloServicePolyglot obj1 = (HelloServicePolyglot)Activator.CreateInstance(
                            typeof(HelloServicePolyglot), constr,attrs);
constr[0] = "es";
HelloServicePolyglot obj2= (HelloServicePolyglot)Activator.CreateInstance(
                            typeof(HelloServicePolyglot), constr,attrs);
for (int i=0; i < 2; i++) {
  Console.WriteLine(obj1.sayHello()); // Bonjour tout le monde
  Console.WriteLine(obj2.sayHello()); // Hola a todos
 }
obj1.Language=null; obj2.Language="ru";
for (int i=0; i < 2; i++) {
Console.WriteLine(obj1.sayHello()); // Hello World
Console.WriteLine(obj2.sayHello()); // Добрый день весь мир
}
```

Exemple de Client Activated Le Serveur

```
namespace Cours.Remoting
{
  /// <remarks>
  /// Hello server to demonstrate the use of .NET Remoting.
  /// </remarks>
  public class HelloServer
  {
   public static int Main(string [] args)
   {
     System.Console.WriteLine("Register ClientActivated with configration file");
     RemotingConfiguration.Configure("HelloServer.exe.config");

     System.Console.WriteLine("Press the enter key to exit...");
     System.Console.ReadLine();
     return 0;
   }
  }
}
```

Propagation de contexte applicatif

- CallContext
  - Permettre d'ajouter des informations additionnelles à l'entête d'un IMessages qui transporte la requête/réponse (transparent aux paramêtres de la méthode)
- Applications
  - Contexte de sécurité, transaction, localisation, session …
- Exemple
  - SetData, GetData, SetHeaders, GetHeaders

Exemple de CallContext

- Coté Service

```
RemotingConfiguration.Configure("HelloClient.exe.config");
LocaleContextData ctx = new LocaleContextData(); // 3 membres
ctx.Country = "France";
ctx.Language = "fr";
ctx.Currency = "EUR";
CallContext.SetData("Locale", ctx);
HelloService obj = new HelloService();
obj.sayHello(); // Bonjour tout le monde
```

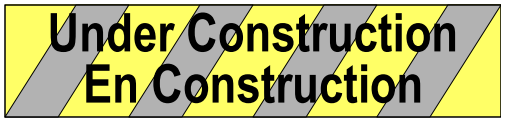
- Coté Service

```
public string sayHello() {
    LocaleContextData ctx = (LocaleContextData )CallContext.GetData("Locale");
    if (ctx != null) {
       return strHello(ctx.Language)+" "+strWorld(ctx.Language);
    } else {
       return strHello(null)+" "+strWorld(null);
    }
}
```

Bail et Sponsor

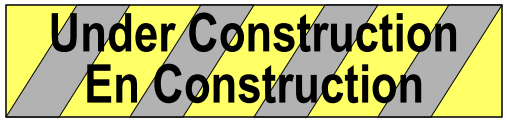
- Motivation
  - Permettre la déactivation des objets d'un client en cas de crash de ce dernier
- Principe
  - Une durée de bail (lease) est attribuée à chaque invocation
  - En cas de dépassement, l'objet peut être considère comme inutilisé et être GC
  - Cependant, un Sponsor peut automatiquement renouvellé le bail avant son expiration

Marshaller



- MarshalByValue
- MarshalByObj

Objets de callback

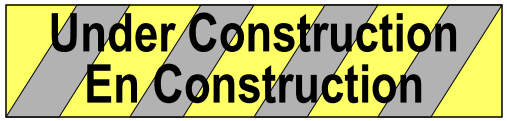


- Voir
  - http://www.codeproject.com/csharp/remotingcallbacks.asp

Sécurité

- Communication
  - TcpChannel est efficace mais n'est pas securisé
  - HttpChannel peut utiliser une couche SSL et IIS peut authentifier les logins NT en environnement LAN
- Objet

Invocation Asynchrone



Autre channel : IiopChannel

```
interface IiopHelloService  {
  string sayHello(string name);
};
```

- Coté client

```
// read the IOR from the file
string ior;
using (StreamReader iorFile = new StreamReader( @"c:\hello.ior")) {
    ior = iorFile.ReadToEnd();
}
// Initialise le canal IIOP
ChannelServices.RegisterChannel(new IiopClientChannel());
Console.WriteLine(".NET Remoting call Corba server...");
IiopHelloService service =
    (IiopHelloService) Activator.GetObject(typeof(IiopHelloService), ior);
Console.WriteLine( service.sayHello("Didier") );

Console.Out.WriteLine("Press [Enter] to exit"); Console.ReadLine();
```

Comparaison

	RPC	RMI	CORBA	.NET Remoting	SOAP
Qui	SUN/OSF	SUN	OMG	MicroSoft/ECMA	W3C
Plate-formes	Multi	Multi	Multi	Win32, FreeBSD, Linux	Multi
Langages de Programmation	C, C++, …	Java	Multi	C#, VB, J#, …	Multi
Langages de Définition de Service	RPCGEN	Java	IDL	CLR	WSDL (XML)
Réseau	TCP, UDP	TCP, HTTP, IIOP customisable	GIOP, IIOP, Pluggable Transport Layer	TCP,HTTP, IPC *IIOP*	RPC,HTTP
Marshalling	XDR	Sérialisation Java	Représentation IIOP	Formatteurs Binaire, SOAP	SOAP +XML Schema
Nommage	IP+Port	RMI, JNDI,JINI	CosNaming	IP+Nom	IP+Port, URL
Intercepteur	Non	depuis 1.4	Oui	Oui CallContext	Extension applicative dans le header
Extra		Chargement dynamique des classes	Services Communs Services Sectoriels	Pas de Chargement dynamique des classes	*(rem: binary XML)*

Bibliographie

- Banerjee et al, C# Web Services, Ed Wrox, ISBN 1861004397
  - Chapitre 6
- Templeman, Vitter, « Visual Studio .NET: The .NET Framework Black Book », Ed Coriolis, ISBN 1-57610-995-X
  - Chapitre 13
- http://staff.develop.com/woodring/dotnet/#remoting
  - Une mine d'exemple

Bibliographie

- http://www.csharphelp.com/archives2/archive421.html
- http://www.csharphelp.com/archives2/archive422.html
- http://www.csharphelp.com/archives2/archive423.html
- http://www.csharphelp.com/archives2/archive424.html