

CORBA als Ordnung in verteilten Anwendungen

«Callback» Example

Anton Böhm

anton.boehm@itServe.ch

itServe AG

P.O.Box

Länggass-Str.26

CH-3000 Bern 9

Tel. +41 31 305 16 16

«Push» Fähigkeit

⇒ Ziel: Zusätzliche Anforderungen, d.h.
der Server «informiert» den Client

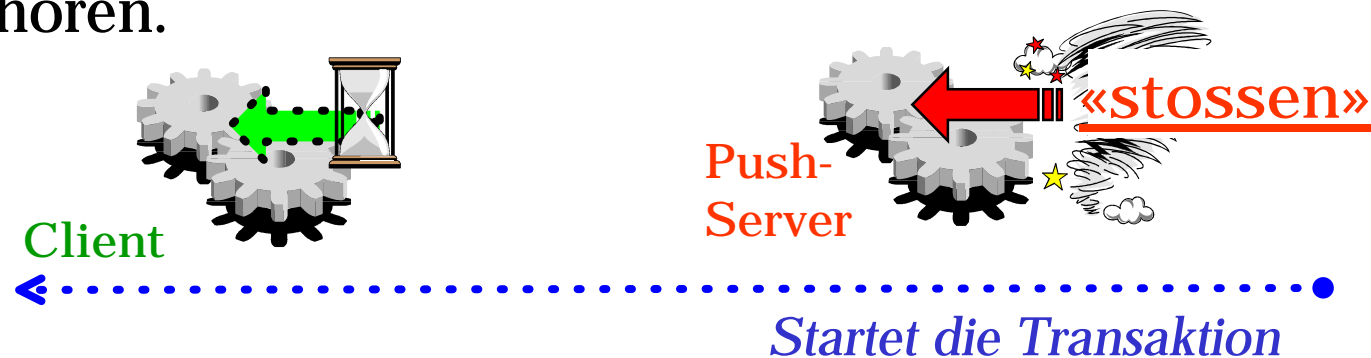
⇒ Bisher im klassischen Client/Server

«Pull-Modell»



«Push» Fähigkeit (II)

⇒ Beim «Push-Modell» drückt der Server Informationen zum Client, d.h., der Server initialisiert die Transaktion. Wenn sich beispielsweise ein Aktienkurs ändert, kann ein Server die aktualisierten Informationen per Push an seine Clients senden. Push kann wesentlich effizienter sein als Pull, wenn Informationen unregelmäßig aktualisiert werden, besonders wenn mehrere Clients einem System angehören.



IDL Example

```
module abHelloCallback{  
    interface>HelloCallback{  
        void>callback (in string message);  
    };  
    interface>Hello{  
        string>sayHello (  
            in>HelloCallback objRef, in string message);  
        oneway void>doCallback (  
            in>HelloCallback objRef, in long count,  
            in string message );  
    };  
};
```

Hello Impl Server Example

```
public class HelloImpl extends HelloPOA {  
:  
    public String sayHello(HelloCallback callobj,  
                           String msg)  
    {  
        System.out.println( "[Info::HelloImpl] ....  
        if(callobj != null) callobj.callback(msg);  
        return "\nMessage from server...\n";  
    }  
:  
}
```

Hello Client Example

```
class HelloCallbackListener extends Thread{
    public HelloCallbackListener() { ...try...
        // servant
        helloCallbackServant= newHelloCallbackImpl();
        poa= POAHelper.narrow(orb.res..ces("RootPOA"));
        helloCallbackRef= poa.servant_to_reference(
            helloCallbackServant );
    :
    public HelloCallback getHelloCallback(){...}
    public void run(){...
        poa.the_POAManager().activate();
        orb.run();
    }
```

Hello Client Example (II)

```
public static void main(String args[]){  
    HelloCallbackListener aHelloCallbackListener=  
        new HelloCallbackListener();  
    aHelloCallbackListener.start();  
    :  
    obj=...solve(...("abHelloCallback..."));  
    Hello helloRef = HelloHelper.narrow( obj );  
    String hello = helloRef.sayHello(  
        aHelloCallbackListener.getHelloCallback()  
        ,"\nClient calls server..\n");  
    System.out.println("[Info::Client] hello-string  
        is " + hello);  
}
```

Hello Client Example (III)

```
public class HelloCallbackImpl extends  
    HelloCallbackPOA{  
    :  
    // method  
    public void callback (String notification)  
    {  
        System.out.println( "__@__[Info::HelloC....  
        System.out.println (notification);  
    }  
    :  
}
```