Invocação Dinâmica em CORBA

Stubs Pré-Compilados

Até agora vimos que clientes necessitam *stubs* pré-compilados para invocarem operações sobre um objeto-servidor.

Um *stub* é suprido para cada interface que o cliente pode usar em um objeto-servidor.

Applets fazem download de bytes-code de *stubs* em tempo de execução, quando é necessário invocar uma operação num objeto-servidor.

Stubs Pré-Compilados

Contudo, no caso de applets, ainda se necessita de *stubs* para a invocação de operações num objeto-servidor.

Assim, deve existir uma maneira para clientes acessarem e executarem objetos sobre uma rede, sem requerer *stubs* pré-compilados para cada interface especificada para o servidor.

Dynamic Invocation Interface - DII

Uma abordagem de ligação em tempo de execução sem *stubs*.

DII em CORBA permite qualquer programa-cliente descobrir um objeto alvo, no servidor, em tempo de execução, e dinamicamente invocar seus métodos.

Dynamic Invocation Interface - DII

O programa-cliente pode invocar qualquer operação sobre qualquer objeto sem requerer *stubs* pré-compilados.

Isto significa que o cliente descobre a informação relacionada à interface em tempo de invocação. O cliente não requer conhecimento em tempo de compilação.

Dynamic Invocation Interface - DII

Servidores oferecem novos serviços e interfaces sempre que estes se tornam disponíveis.

Clientes descobrirão estas interfaces em tempo de execução e sabem como chamá-las.

DII provê um ambiente dinâmico que permite seus sistemas permanecerem flexíveis e extensíveis. Esta é uma característica muito desejável em ambientes como a Internet.

Descobrindo Objetos Remotos

Mas, como os programas-clientes primeiro descobrem objetos remotos numa rede ?

Na abordagem mais simples, uma referência a objeto na forma de string é passada ao cliente. Neste caso, o cliente pode então converter a string em um referência de objeto e fazer a conexão.

Descobrindo Objetos Remotos

Localizar objetos pelo nome, usando o Naming Service do CORBA.

Podem descobrir esses objetos remotos via o Trade Service, procurando eles pelas suas propriedades.

Descobrindo Objetos Remotos

Na Object Web, objetos serão dinamicamente descobertos por agentes de todos os tipos (spiders, crawlers, bots, search engines, Publish-and-Subscribe Services).

Uma vez que clientes descubram esses objetos remotos, eles necessitarão do CORBA DII para invocar suas operações.

O Cliente Dinâmico

Assim, o programa-cliente deve suportar as linhas de código para se implementar uma interface de invocação dinâmica. Deve-se escrever o *Cliente Dinâmico*.

Escrevemos mais linhas de código no cliente.

O lado do servidor não é, neste caso, modificado.

O Cliente Dinâmico

É mais complexo que o seu correspondente estático, com *stubs* pré-compilados.

Seremos capazes de comparar a performance de uma invocação em CORBA dinâmico com ama invocação do CORBA estático.

Invocações Dinâmicas

Antes de podermos dinamicamente invocar um método sobre um objeto, devemos primeiro descobrir o objeto e obter sua referência.

Pode-se usar essa referência para recuperar a interface do objeto e dinamicamente construirmos o pedido.

Deve-se especificar no pedido o método que se deseja executar e seus parâmetros. Esta informação é obtida de um REPOSITÓRIO DE INTERFACE (IR).

O Grande Quadro (1)

Assume-se que adquirimos, por qualquer meio, uma referência para o objeto que deseja-se dinamicamente invocar.

Uma descrição de alto nível de como se invoca um método remoto sobre esse objeto.

O Grande Quadro (2)

1. Obtém-se o nome da interface:

Objetos CORBA são **introspectivos**; eles podem nos prover uma porção de informação sobre eles próprios.

Conseqüentemente, pode-se perguntar a este objeto pelo nome de sua interface, por invocar o método *get_interface()*. Esta chamada retorna uma referência para o objeto **InterfaceDef**. Este é um objeto dentro do Repositório de Interface.

O Grande Quadro (3)

2. Obtém a descrição do método a partir do Repositório de Interface.

Pode-se usar a InterfaceDef com uma entrada para navegação no IR. Pode-se obter então toda a informação detalhada sobre a interface e os métodos que ela suporta. CORBA especifica dez chamadas para se navegar no IR e descrever os objetos que ela contém.

O Grande Quadro (4)

O cliente emite um *lookup_name* para descobrir o método que ele deseja invocar. Então emite uma chamada *describe* para obter a definição completa dos métodos. Pode-se emitir *describe_interface* para obter uma descrição completa da interface e encontrar o método que se deseja invocar.

3. Cria a lista de argumentos

CORBA especifica uma estrutura de dados auto-definida para passar parâmetros, qie é chamada a *Named Value List*.

O Grande Quadro (5)

Implementa-se esta lista usando-se um pseudo-objeto **NVList**. Cria-se esta lista invocando-se *create_list* e invoca-se a chamada *add-item* tantas vezes quantos forem os argumentos na lista. Alternativamente, pode-se deixar o ORB criar a lista de argumentos, invocando-se *create_operation_list* sobre um objeto **CORBA::ORB**. Deve-se passar o nome da operação, para a qual é retornada uma lista.

O Grande Quadro (6)

4. Criar o pedido

Um pedido é um pseudo-objeto CORBA que contém o nome do método, a lista de argumentos e o valor retornado. Cria-se um pedido por invocar *create_request.* Deve-se passar nele o nome do método a invocar, o **NVList** e um apontador para o valor de retorno. Se o método invocado não cont`me parâmetros usar *_request* do pseudo-objeto

CORBA:: *Object.*

O Grande Quadro (7)

5. Invocar o pedido.

Pode-se invocar um pedido em três modos:

1. A chamada *invoke* envia o pedido e obtém os resultados.
2. A chamada *send_deferred* retorna o controle ao programa, o qual deve então “ poll” para a resposta com *get_response* ou *poll_response*.
3. A chamada *send* pode ser definida para ser um datagrama por emitir um *send_oneway*, neste caso nenhuma resposta é necessária.

Pseudo-Objetos

Um pseudo objeto é um objeto que o ORB cria diretamente, mas pode-se invocá-lo como qualquer outro objeto.

O próprio ORB é um pseudo-objeto.

Pseudo-Objetos

Os serviços (métodos) de que necessitamos para dinamicamente invocar um objeto são partes do núcleo do CORBA.

Esses métodos estão em quatro interfaces no módulo CORBA, identificadas como interfaces de pseudo-objetos.

CORBA::Object

É uma interface de pseudo-objeto que define operações que todo objeto CORBA deve suportar. Estas operações são realizadas pelo ORB. Simplesmente herda-se elas quando se cria um objeto.

É a interface-raiz para todos os objetos CORBA. Esta interface inclui três métodos que se pode invocar para usar construir invocações dinâmicas.

Operações sobre CORBA::Object

get_interface

create_request

_request

CORBA::Request

É uma interface de pseudo-objeto que define as operações sobre um objeto remoto. Algumas operações são:

add_arg

invoke

send_oneway

send_deferred

get_response

poll_response

delete

CORBA::NVlist

É uma interface de pseudo-objeto que auxilia a construção de listas de parâmetros.

Um objeto NVList mantém uma lista de itens de dados auto-descritos chamados NamedValues.

A interface NVList define operações para se manipular um lista de parâmetros.

IDL que define uma lista de NVList

```
struct  NamedValue  {
    Identifier  name;  //argument name
    any         argument;  //argument
    long        len;  //length/count of argument
                                value
    Flags       arg_modes;  // in, out, inout
};
```

Operações de CORBA::NVList

add_item

add-value

get_count

remove

free-Memory

free

CORBA::ORB

É uma interface de pseudo-objeto que define operações de propósito geral do ORB.

Pode-se invocar as operações sobre um pseudo objeto CORBA::ORB, a partir do Cliente ou a partir do Servidor.

Seis dessas operações são para a construção de um “ request” dinâmico.

Operações do CORBA::ORB

create_list

create_operation-list

send_multiple_requests_oneway

send_multiple_requests_deferred

poll_next_response

get_next_response

Objetos do Repositório de Interface

Em adição a estas quatro interfaces, também pode-se usar objetos do Repositório de Interfaces para construir uma invocação remota, como foi usado o objeto **InterfaceDef**.

Cenários de Invocação

Do-It-Yourself

ORB-Can-Help

Yet-Another-Way

Quando usar CORBA Dinâmico

Situações de Uso	Técnica de Invocação Recomendada
Cliente invoca objeto servidor frequentemente.	Stubs Pre-compilados estáticos
Cliente invoca objeto-servidor infrequentemente.	DII
Cliente descobre objeto servidor em tempo de execução.	DII
Cliente roda dentro de browser.	Usar applet e stubs estático; Applet torna-se cliente para esse objeto.

Rodar o Cliente/Servidor Programa

Executar o OSAgent (ORB do Visibroker)

Executar o Naming Service.

Executar o Servidor.

Construir um Repositório de Interface

Preencher o Repositório

Executar o Cliente com a DII.