



PRÁTICA CORBA

Objetos Distribuídos - CORBA

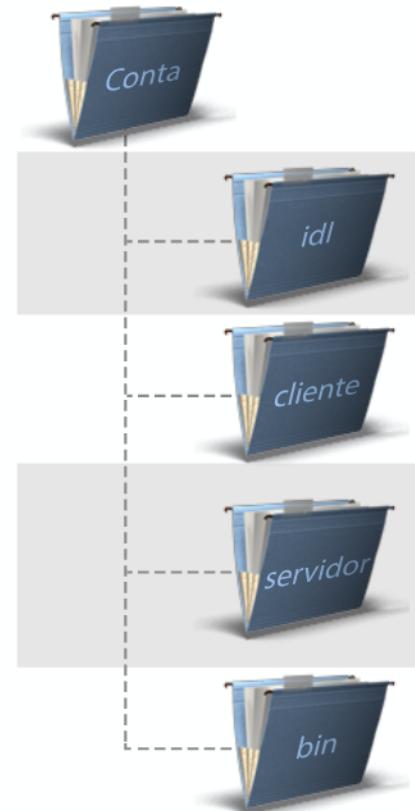


Implementação do Cliente

Objetos Distribuídos - CORBA

EXEMPLO: Cliente de Conta

- Criar **estrutura de diretórios** (recomendado):
- Gerar **stub**:
 - `tao_idl -Gstl Conta.idl`
- Criar `cliente.cpp`



Conta.idl +
stubs

cliente.cpp

ContaI.h
ContaI.cpp
servidor.cpp

servidor_conta
cliente_conta



Cliente CORBA: Passos Típicos

- Cliente

- Declarar variável do **tipo da interface**:
 - ◆ *Interface ref;*
- Obter **referência** para o objeto servidor:
 - ◆ *ref = obtém_referência (...);*
- Usar a referência para **chamar os métodos**:
 - ◆ *res = ref.método (parâmetros);*



cliente.cpp

```
//...
#include "ContaC.h"    // stub
//...
using namespace CORBA;

int main(int argc, char* argv[])
{
    try {
        ORB_var orb = ORB_init(argc, argv, "ORB");
```

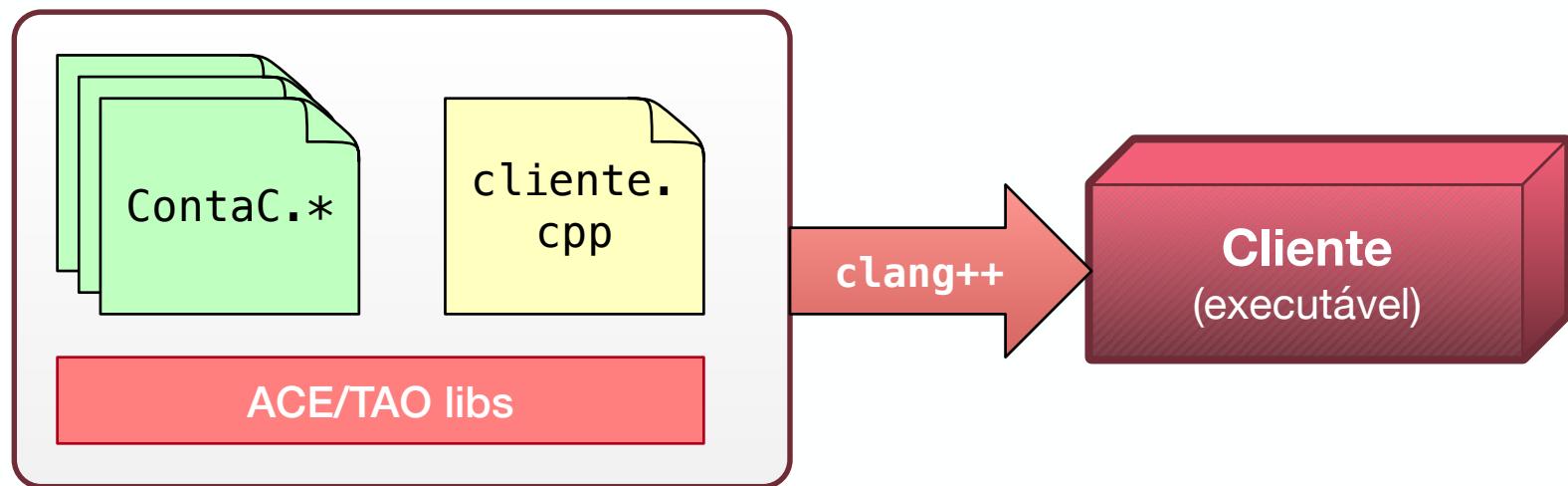
```
Object_ptr ref = orb->string_to_object(argv[1]);
Conta_var conta = Conta::_narrow(ref);

conta->deposito(123.45);
cout << "Saldo = " << conta->saldo() << endl;

orb->destroy();
} catch (Exception& e) {
    ...
}
```

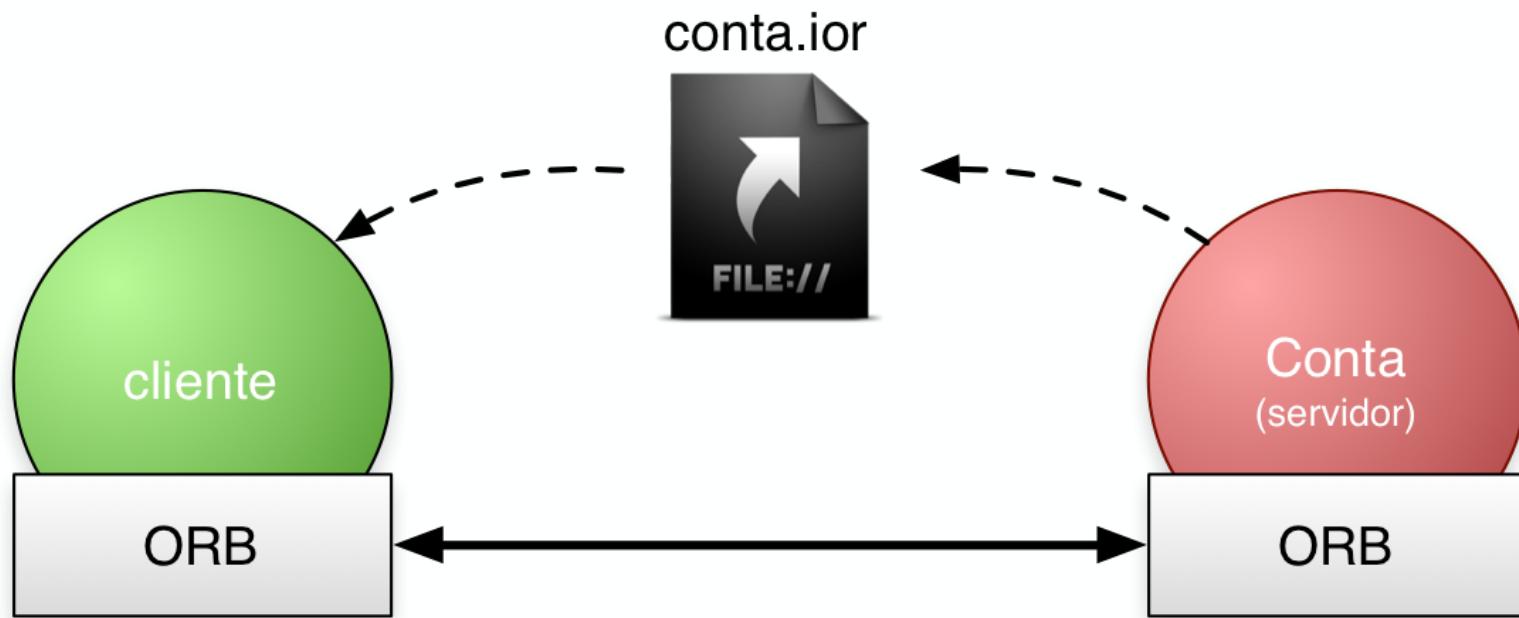
Implementação Cliente Conta (C++ ACE/TAO)

- Compilar arquivos do cliente



Implementação Cliente Conta (C++ ACE/TAO)

- Executar **conta** e depois, **cliente**





IDL: Visão Geral

Objetos Distribuídos - CORBA

- Linguagem puramente declarativa:
 - Não é possível programar em IDL
- Sintaxe similar a C++ ou Java
- Compilada para gerar:
 - Código de comunicação (stub e skeleton) na linguagem-alvo;
 - Metadados da interface para o repositório de interfaces.