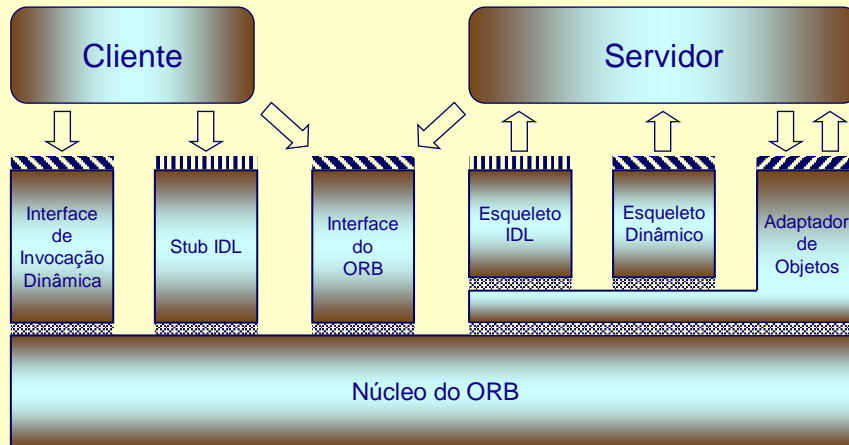


# Interfaces do ORB

# Reverendo as Partes de CORBA



- Independe de ORB
- Depende das definições IDL
- Depende do adaptador
- Interface proprietária

## Pseudo-Objetos

- As interfaces do ORB são especificadas em pseudo-IDL (PIDL)
- PIDL parece IDL
  - Em PIDL podemos ter operações fora de interfaces
- PIDL é usada para pseudo-objetos, que não são objetos CORBA
- Esses pseudo-objetos tem pseudo-interfaces definidas em PIDL
- O usuário não define pseudo-interfaces
  - Pseudo-objetos fazem parte do ambiente do ORB
  - Suas pseudo-interfaces são pré-definidas



## Por que Pseudo-Objetos?

- As interfaces do ORB não são implementadas por objetos CORBA “normais” e sim por bibliotecas fornecidas com o ORB
- As operações de um objeto CORBA “normal” podem ser invocadas remotamente (por outros processos)
- As operações de uma pseudo-objeto residente num processo só podem ser chamadas por esse processo
- Pseudo-objetos servem para se acessar módulos de “run-time” do ORB incorporados ao processos
- Duas pseudo-interfaces muito importantes: `CORBA::ORB` e `CORBA::Object`
- Aplicações usam essas pseudo-interfaces para chamar funções do ORB



## Pseudo-Objetos X Objetos CORBA “Normais”

- Operações de pseudo-interfaces não podem ser invocadas remotamente
- Pseudo-interfaces não são implicitamente derivadas de `CORBA::Object`
- Em geral pseudo-objetos não podem ser passados como argumentos para operações de objetos CORBA “normais”
  - Porém alguns podem (`CORBA::Object` e `CORBA::TypeCode`)
- Operações de pseudo-interfaces não podem ser invocadas pela DII
- Definições de pseudo-interfaces não vão para o Repositório de Interfaces
- O mapeamento de uma pseudo-interface para uma linguagem de programação pode ser especial



## Inicialização do ORB

```
// PIDL
module CORBA {
    typedef string ORBid;
    typedef sequence<string> arg_list;
    interface ORB; // forward declaration

    // N.B.: A operação abaixo não está
    // dentro de uma interface
    ORB ORB_init(inout arg_list argv,
                in ORBid orb_identifier);

    ...
};
```



## Inicialização do ORB em C++

```
namespace CORBA {  
    ...  
    ORB_ptr ORB_init(int& argc,  
                    char** argv,  
                    const char* orb_identifier = "");  
    ...  
};
```

- **Uso:**

```
int main (int argc, char* argv[]) {  
    CORBA::ORB_ptr orb;  
    try {  
        orb = CORBA::ORB_init(argc, argv);  
    }  
    catch (...) {  
        cerr << "Erro ..." << endl;  
        exit(1);  
    }  
    ... // usa o ORB  
    CORBA::release(orb);  
    return 0;  
}
```



## Inicialização do ORB em Java

```
package org.omg.CORBA  
abstract public class ORB {  
    ...  
    public static ORB init(String[] args,  
                          java.util.Properties props);  
    ...  
}
```

- **Uso:**

```
public static void main(String[] args) {  
    org.omg.CORBA.ORB orb;  
    try {  
        orb = org.omg.CORBA.ORB.init(args, null);  
    }  
    catch (Exception e) {  
        e.printStackTrace();  
        System.exit(1);  
    }  
    ... // usa o ORB  
}
```



## A Pseudo-Interface CORBA::ORB

```
// PIDL
module CORBA {
    ...
    interface ORB {
        string object_to_string(in Object obj);
        Object string_to_object(in string str);
        ...
    };
    ...
};
```



## A Pseudo-Interface CORBA::Object

```
// PIDL
module CORBA {
    ...
    interface Object {
        Object duplicate();
        void release();
        boolean is_nil();
        boolean is_a(in string repository_id);
        boolean non_existent();
        boolean is_equivalent(in Object other);
        unsigned long hash(in unsigned long max);
        ...
    };
    ...
};
```



## CORBA::Object em C++

- `static Foo_ptr Foo::_duplicate(Foo_ptr src);`
- `void CORBA::release(Object_ptr p);`
- `Boolean CORBA::is_nil(Object_ptr p);`
- `Boolean Object::_is_a(const char* id);`
- `Boolean Object::_non_existent();`
- `Boolean Object::_is_equivalent(  
Object_ptr other_obj  
);`
- `ULong Object::_hash(ULong max);`



## CORBA::Object em Java

```
Package org.omg.CORBA;  
public interface Object {  
    org.omg.CORBA.Object _duplicate(); // nop  
    void _release(); // nop  
    // não há is_nil() (basta comparar com null)  
    boolean _is_a(String repository_id);  
    boolean _non_existent();  
    boolean _is_equivalent(Object other);  
    int _hash(int max);  
    ...  
};  
...  
}
```

