

# **Распределенные вычислительные системы**

## **Лекция №5: Пример приложения на CORBA**

Алексей В. Бурдаков, к.т.н.  
[burdakov@usa.net](mailto:burdakov@usa.net)

# План лекций

---

№	Дата	Тема
1	04.09	Вводная лекция
2	11.09	Эволюция распределенных технологий
3	18.09	Принципы ПО среднего слоя
4	25.09	Стандарты OMG, CORBA и ORB
5	02.10	Пример приложения на CORBA
6	09.10	
7	16.10	
8	23.10	Перенос
9	30.10	
10	06.11	

# План лекции

---

- Пример приложения

# Простое приложение Hello World!

---

```
// C++  
#include <iostream.h>  
int main(int, char*[])  
{  
    cout << "Hello World!" << endl;  
    return 0;  
}
```

# CORBA-приложение Hello World!



# Используемый ORB: ORBacus

---

- Создан фирмой IONA ([www.iona.com](http://www.iona.com))
  - небольшой объем (несколько МБ)
  - распространяется в исходных кодах
  - поддержка платформ:
    - AIX 4.3
    - Compaq Tru64
    - HP-UX 11.00
    - SGI Irix 6.5
    - Linux /x86
    - SUN Solaris 2.6/7/8
    - Windows 95/98/NT/2000/XP
- ЯЗЫКИ :
  - C++
  - Java



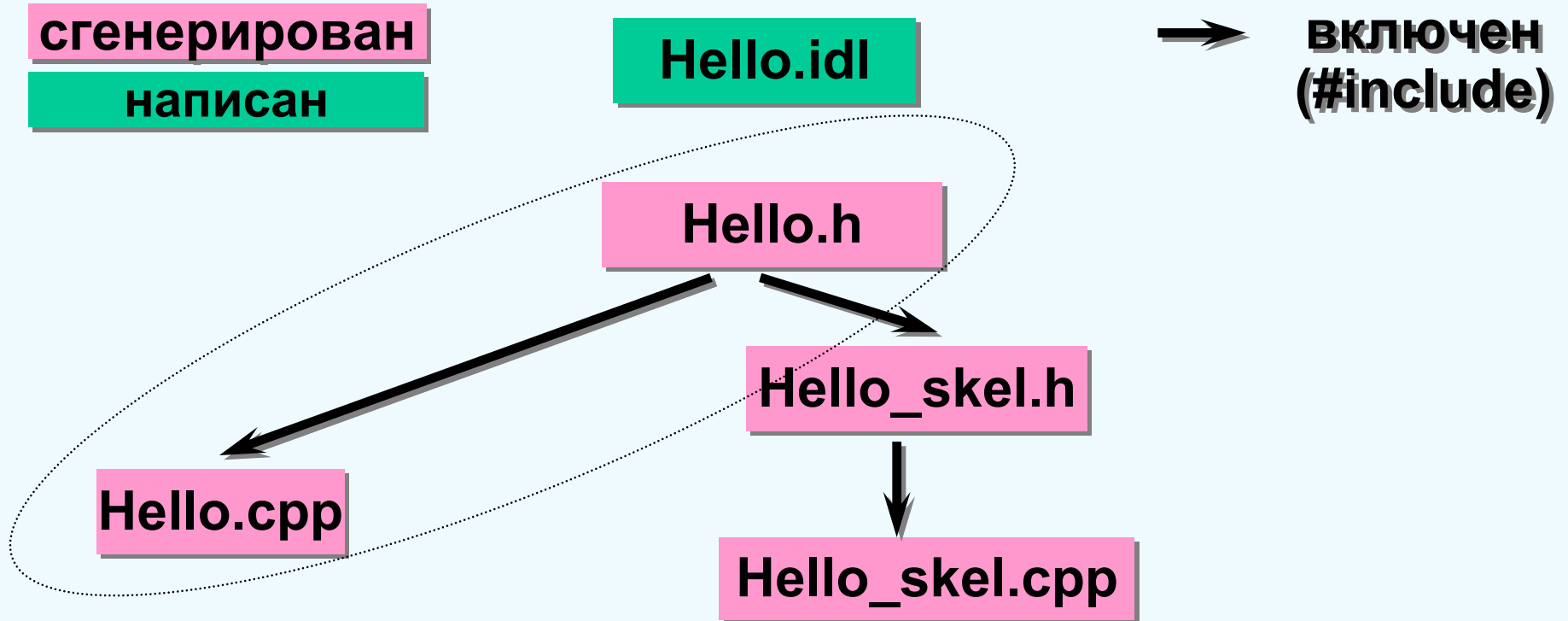
# IDL-определение интерфейса

---

```
1  // IDL -- файл Hello.idl
2  interface Hello
3  {
4      void say_hello();
5  };
```

Генерация C++ фалов:  
> *idl Hello.idl*

# Генерируемые файлы C++



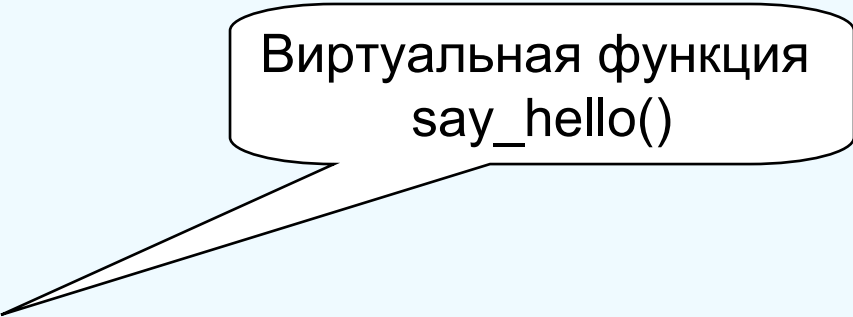
**> *idl Hello.idl***



# Заголовок Hello.h

---

```
// Generated by the ORBacus IDL-to-C++ Translator
// ...
typedef OB::ObjVar< Hello > Hello_var;
// ...
class Hello : virtual public CORBA::Object
{
    // ....
    protected:
    // ....
    public:
    // ....
        virtual void say_hello() = 0;
};
```



Виртуальная функция  
say\_hello()

# Реализация заглушек (stub) Hello.cpp

```
// Generated by the ORBacus IDL-to-C++ Translator
#include <OB/CORBAClient.h>
#include <hello.h>
// ...
void OBMarshalStubImpl_Hello::say_hello()
{
// ...
    OB::Downcall_var _ob_down = _OB_createDowncall("say_hello",
true);
        _OB_preMarshal(_ob_down);
        _OB_postMarshal(_ob_down);
        _OB_request(_ob_down);
        _OB_preUnmarshal(_ob_down);
        _OB_postUnmarshal(_ob_down);
        return;
}
```

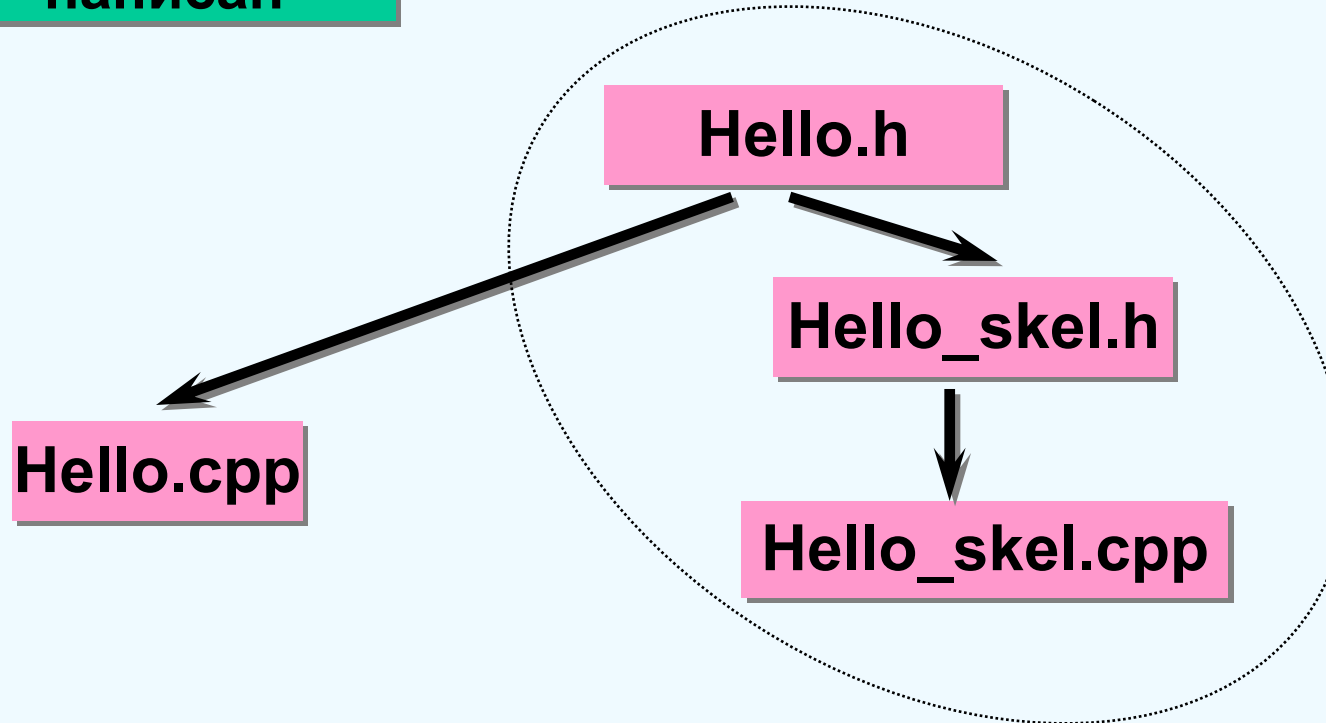
Отправка запроса  
say\_hello на сервер

# Генерируемые файлы C++

сгенерирован  
написан

Hello.idl

→ Включен  
(#include)

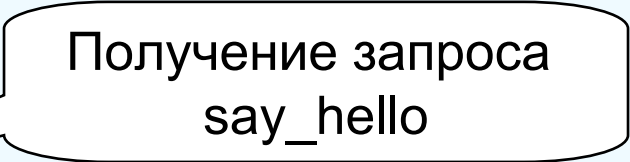


> *idl Hello.idl*

# Реализация серв. заглушек (skeleton) Hello\_skel.h

---

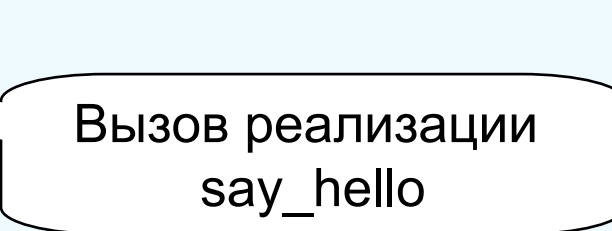
```
// Generated by the ORBacus IDL-to-C++ Translator
#include <hello.h>
class POA_Hello : virtual public PortableServer::ServantBase
{
protected:
    // ...
    void _OB_op_say_hello(OB::Upcall_ptr);
    // ...
public:
    virtual void say_hello() throw(CORBA::SystemException) = 0;
}
```



Получение запроса  
say\_hello

# Реализация серв. заглушек (skeleton) Hello\_skel.cpp

```
// Generated by the ORBacus IDL-to-C++ Translator
#include <OB/CORBAServer.h>
#include <hello_skel.h>
// ...
void POA_Hello::_OB_op_say_hello(OB::Upcall_ptr _ob_up)
{
    _OB_preUnmarshal(_ob_up);
    _OB_postUnmarshal(_ob_up);
    say_hello();
    _OB_preMarshal(_ob_up);
    _OB_postMarshal(_ob_up);
}
// ...
```



Вызов реализации  
say\_hello

*Всего 122 строки*

# Реализация клиента

сгенерирован  
написан

Hello.idl

→ Включен  
(#include)

Hello.h

Hello\_skel.h

Hello.cpp

Hello\_skel.cpp

Client.cpp

# Реализация клиента

## Client.cpp

```
1 // C++
2 #include <OB/CORBA.h>
3 #include <Hello.h>
4
5 #include <fstream.h>
6
7 int run(CORBA::ORB_ptr);
8
9 int main(int argc, char* argv[])
10 {
11 ... // Same as for the server
12 }
13
14 int run(CORBA::ORB_ptr orb)
```

Код повторяет код сервера  
(запуск ORB)

# Реализация клиента

## Client.cpp

```
15 {  
16 const char* refFile = "Hello.ref";  
17 ifstream in(refFile);  
18 char s[2048];  
19 in >> s;  
20 CORBA::Object_var obj = orb -> string_to_object(s);  
21  
22 Hello_var hello = Hello::_narrow(obj);  
23  
24 hello -> say_hello();  
25  
26 return 0;  
27 }
```

Получение ссылки  
на созданный объект

Вызов операции  
удаленного CORBA-объекта

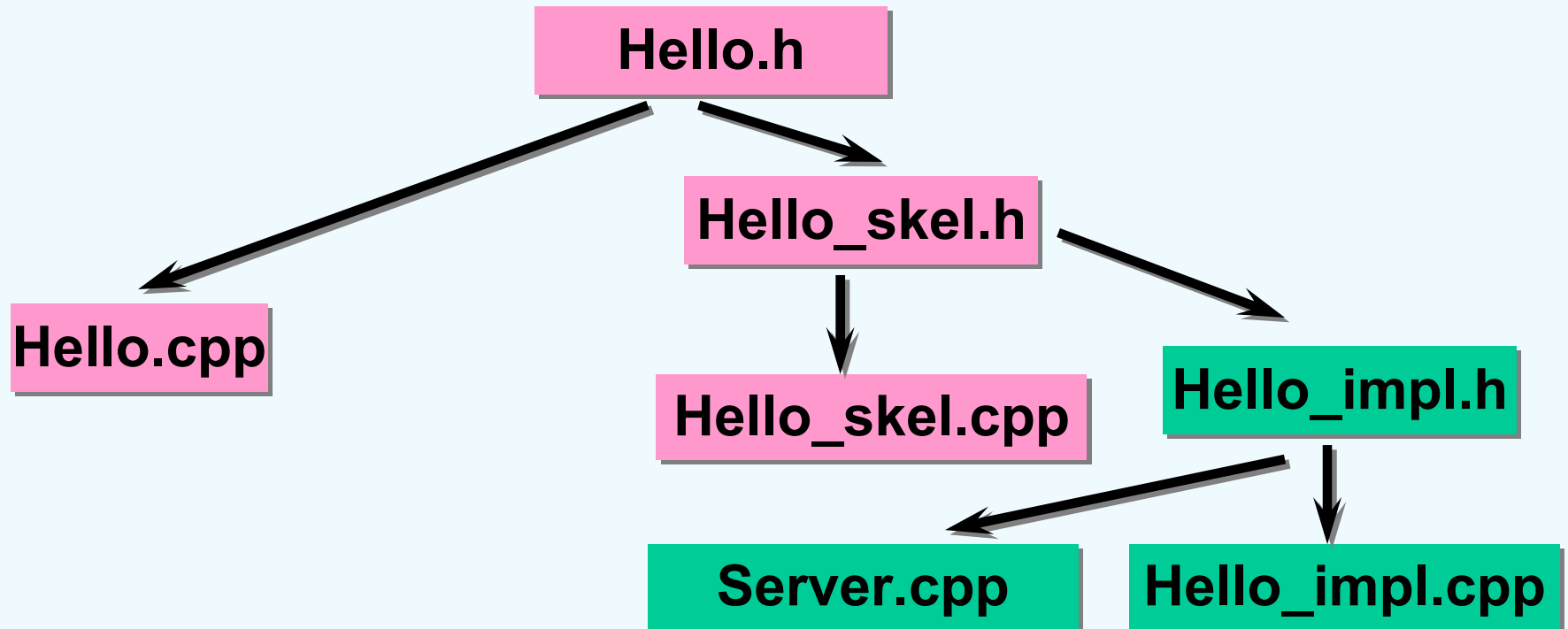


# Реализация сервера

сгенерирован  
написан

Hello.idl

→ Включен  
(#include)



# Реализация Hello\_impl.h

```
1 // C++
2 #include <Hello_skel.h>
3
4 class Hello_impl : public POA_Hello,
5                   public
6   PortableServer::RefCountServantBase
7 {
8 public:
9     virtual void say_hello()
10    throw(CORBA::SystemException);
11 };
```

Определен в  
Hello\_skel.h

# Реализация Hello\_impl.cpp

```
1 // C++
2 #include <iostream.h>
3 #include <OB/CORBA.h>
4 #include <Hello_impl.h>
5
6 void Hello_impl::say_hello()
7     throw(CORBA::SystemException)
8 {
9     cout << "Hello World!" << endl;
```

Определение  
стандартных  
CORBA-классов

Вывод Hello World!

# Реализация сервера

## Server.cpp: main()

```
1 // C++
2 #include <OB/CORBA.h>
3 #include <Hello_impl.h>
4
5 #include <fstream.h>
6
7 int run(CORBA::ORB_ptr);
8
9 int main(int argc, char* argv[])
10 {
11     int status = EXIT_SUCCESS;
12     CORBA::ORB_var orb;
13
14     try
15     {
16         orb = CORBA::ORB_init(argc, argv);
17         status = run(orb);
18     }
```

Инициализация ORB

Вызов функции run()

# Реализация сервера

## Server.cpp: main()

```
19 catch(const CORBA::Exception&)
20 {
21     status = EXIT_FAILURE;
22 }
24 if(!CORBA::is_nil(orb))
25 {
26     try
27     {
28         orb -> destroy();
29     }
30     catch(const CORBA::Exception&)
31     {
32         status = EXIT_FAILURE;
33     }
34 }
36 return status;
37 }
```

Уничтожить объект  
orb

# Реализация сервера

## Server.cpp: run()

```
1 // C++
2 int run(CORBA::ORB_ptr orb)
3 {
4     CORBA::Object_var poaObj =
5         orb ->
6         resolve_initial_references("RootPOA");
7     PortableServer::POA_var rootPoa =
8         PortableServer::POA::_narrow(poaObj);
9     PortableServer::POAManager_var manager =
10        rootPoa -> the_POAManager();
11
12 Hello_impl* helloImpl = new Hello_impl();
13 PortableServer::ServantBase_var servant = helloImpl;
14 Hello_var hello = helloImpl -> _this();
15
```

Получение указателя на RootPOA

Получение ссылки на POAManager

# Реализация сервера

## Server.cpp: run()

```
16 CORBA::String_var s = orb -> object_to_string(hello);
17 const char* refFile = "Hello.ref";
18 ofstream out(refFile);
19 out << s << endl;
20 out.close();
21
22 manager -> activate();
23 orb -> run();
24
25 return EXIT_SUCCESS;
26 }
```

Получение ссылки  
на объект, перевод ее  
в строковый формат и  
сохранение в файле

Запуск менеджера на  
обслуживание

# Компиляция приложения

сгенерирован

написан

Hello.idl



включен  
(#include)

Hello.h

Hello\_skel.h

Hello.cpp

Hello\_skel.cpp

Hello\_impl.h

Client.cpp

Server.cpp

Hello\_impl.cpp

КОМПИЛЯЦИЯ

Клиентский код

Код реализации



# Компиляция приложения

---

## Клиент

Hello.obj  
Client.cpp  
wsock32.lib  
...

**client.exe**

## Сервер

Hello.obj  
Hello\_skel.cpp  
Hello\_impl.cpp  
Server.cpp  
wsock32.lib  
...

**server.exe**

# Обсуждение примера

---

- **На первый взгляд пример достаточно сложен**
- **Однако он:**
  - Решает проблемы коммуникаций между клиентом и сервером
  - Возможность исполнения клиентов и серверов на различных платформах
  - Независимость от языка программирования
  - Независимость от расположения
- **Современные системы достаточно сложны**

# Литература / Internet ИСТОЧНИКИ

---

- В. Эммерих *Конструирование распределенных объектов*. - М.:Мир. - 2002.
- Vinoksi S. *CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments*, IEEE'96.
- Schmidt D.C. And Vinoski S. *Object Adapters: Concepts and Technology*, SIGS C++ Report, Vol. 9, No. 11, Nov-Dec 1997.
- Ю.А. Григорьев, А.Д. Плутенко. *Жизненный цикл проектирования распределенных баз данных*. - Благовещенск. - 1999.
- [www.omg.org](http://www.omg.org)