

Распределенные вычислительные системы

Лекция №3: Принципы ПО среднего слоя

Алексей В. Бурдаков, к.т.н.
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	
5	02.10	
6	09.10	
7	16.10	
8	23.10	Перенос
9	30.10	
10	06.11	

План лекции

- Определение и типы ПО среднего слоя (middleware)
- Принципы удаленного вызова процедур (RPC)
- Объектно-ориентированное ПО среднего слоя
- Разработка с помощью ОО ПО среднего слоя
- Литература

Эталонная модель ISO/OSI



Концептуальный разрыв

- Объектная заявка: запрос на выполнение операции удаленного объекта
- Обмен данными реализован во всех современных сетевых ОС
- Концептуальный разрыв между вызовом операции и передачей данных
- Средний слой ликвидирует концептуальный разрыв

Прямое использование сетевых протоколов

- Ручное отображение комплексных параметров запросов в последовательности байтов
- Ручное разрешение проблемы гетерогенности представления данных
- Ручная идентификация компонентов с помощью указания доменных имен, номеров портов и т.п.
- Ручная реализация процедур активации компонентов
- Не гарантирована безопасность типов
- Ручная синхронизация взаимодействия между компонентами
- Качество сервиса не гарантировано

Эталонная модель ISO/OSI: ПО среднего слоя

Прикладной	Объекты	Объекты
Представления	?	ПО среднего слоя
Сеанса		
Транспортный	Сетевая ОС	Сетевая ОС
Сетевой		
Канальный		
Физический	Апп. об.	Апп. об.

ПО среднего слоя

- Размещается между приложениями и ОС/сетью
- Делает распределение прозрачным
- Решает проблемы гетерогенности:
 - Аппаратного обеспечения
 - Операционных систем
 - Сетей
 - Языков программирования
- Обеспечивает среду разработки и исполнения для распределенных систем

Виды ПО среднего слоя

- **Транзакционно-ориентированное (2PC, DTP / XA)**
 - IBM CICS
 - BEA Tuxedo
 - Encina
- **Ориентированное на сообщения**
 - IBM MQSeries
 - DEC MessageQueue
 - NCR TopEnd
- **Системы, основанные на RPC**
 - ANSA
 - Sun ONC
 - OSF/DCE

Виды ПО среднего слоя

- **Объектно-ориентированное**
 - **OMG/CORBA**
 - **DCOM**
 - **Java/RMI**

Принципы RPC

- **RPC – основа ПО среднего слоя**
- **Позволяет осуществлять вызовы процедур за границами узлов**
- **Интерфейсы вызовов определены с помощью IDL (Interface Definition Language)**
- **Компилятор RPC генерирует реализацию слоя представления и сессий из IDL**

Пример RPC IDL

```
const NL=64;
struct Player {
  struct DoB {int day; int month; int year;}
  string name<NL>;
};
program PLAYERPROG {
  version PLAYERVERSION {
    void PRINT(Player)=0;
    int STORE(Player)=1;
    Player LOAD(int)=2;
  }= 0;
} = 105040;
```

Задачи, решаемые реализацией уровня представления



- Преобразование между прикладным и транспортным представлением данных называется маршалингом и демаршалингом

Маршалинг и демаршалинг

- Маршалинг: собирает данные в форму для передачи

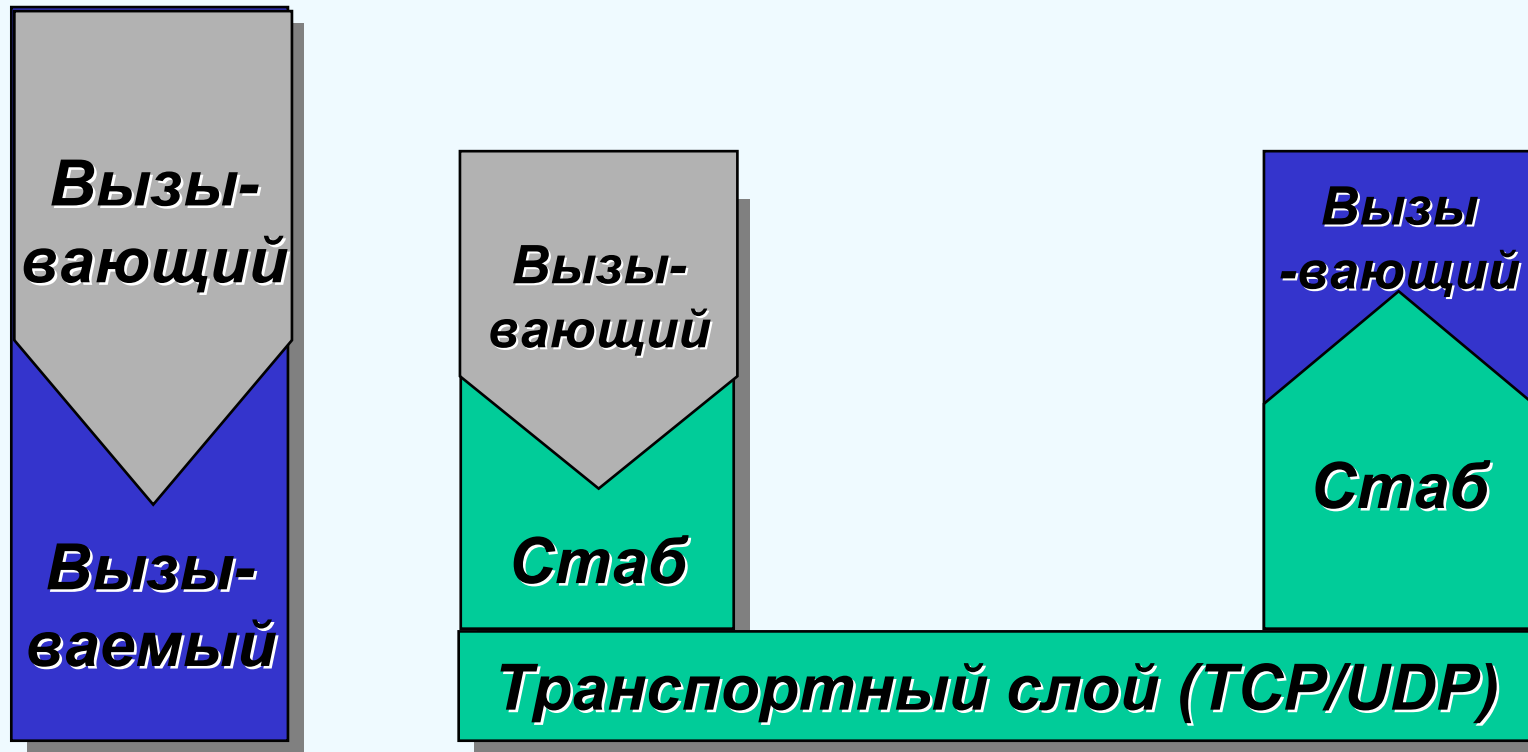
```
char * marshal() {  
    char * msg;  
    msg=new char[4*(sizeof(int)+1) +  
                strlen(name)+1];  
    sprintf(msg,"%d %d %d %d %s",  
            dob.day,dob.month,dob.year,  
            strlen(name),name);  
    return(msg);  
};
```

Маршалинг и демаршалинг

- Демаршалинг: разбирает данные в комплексные структуры

```
void unmarshal(char * msg) {  
    int name_len;  
    sscanf(msg, "%d %d %d %d ",  
           &dob.day, &dob.month,  
           &dob.year, &name_len);  
    name = new char[name_len+1];  
    sscanf(msg, "%d %d %d %d %s",  
           &dob.day, &dob.month,  
           &dob.year, &name_len, name);  
};
```

Вызов метода и объектная заявка



Стабы (Stubs)

- Создание кода для маршалинга и демаршалинга трудоемко и ведет к ошибкам
- Код может быть сгенерирован полностью автоматически из определения интерфейсов
- Код внедрен в стабы клиентов и серверов
- Клиентский стаб представляет сервер для клиента, а серверный – клиентский для сервера
- Стабы безопасны с точки зрения типов
- Стабы также выполняют синхронизацию

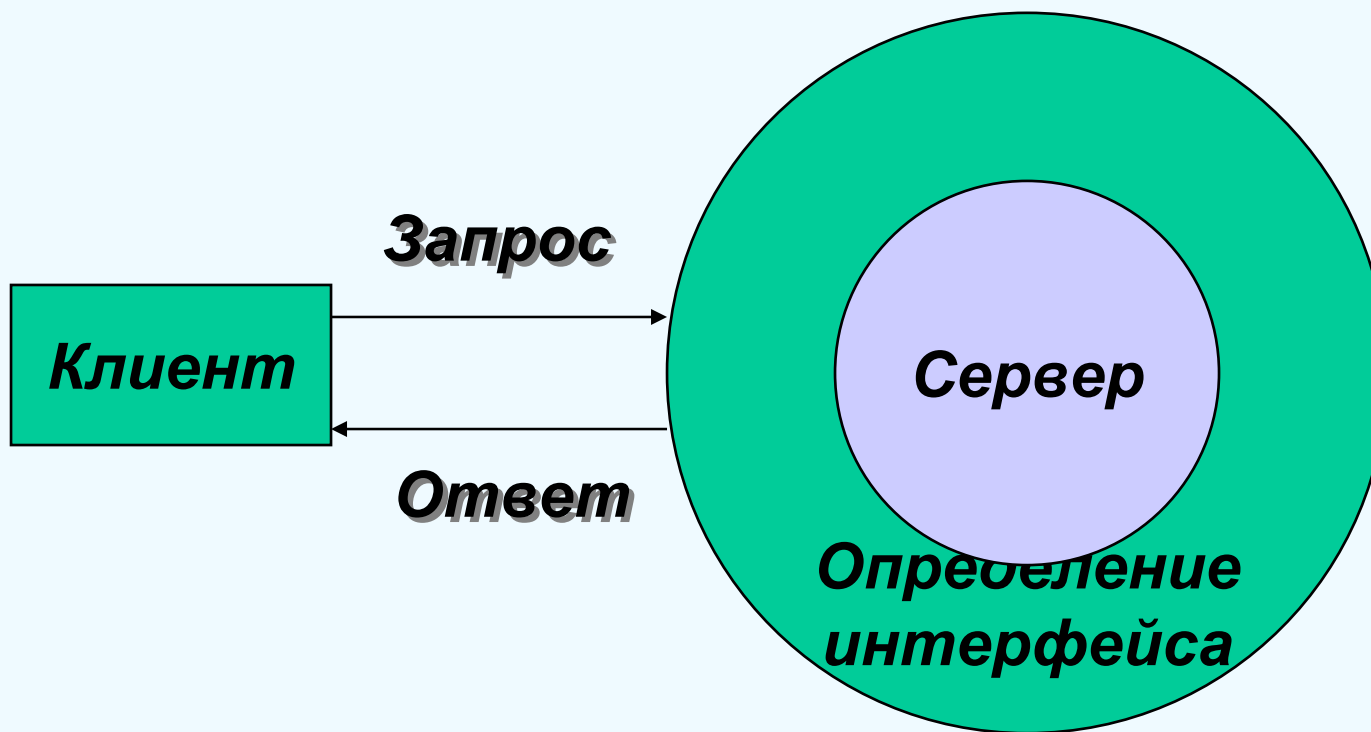
Синхронизация (Synchronization)

- **Цель: достигнуть такой синхронизации как и в случае локальных вызовов**
- **Достигается стабами:**
 - **Клиентский стаб посылает запрос и ожидает до тех пока сервер не вернет ответ**
 - **Сервер ожидает вызовов от серверного стаба и вызывает сервер когда приходит запрос**

Безопасность типов

- Как мы можем гарантировать, что:
 - Серверы могут выполнить операции, запрошенные клиентами?
 - Параметры, переданные клиентом совпадают с ожидаемыми сервером?
 - Результат, передаваемый сервером совпадает с ожидаемым клиентом?
- ПО среднего слоя выступает в роли посредника между клиентом и сервером для обеспечения безопасности типов
- Достигается с помощью определения интерфейса на общем языке

Обеспечение безопасности типов



План лекции

- Определение и типы ПО среднего слоя (middleware)
- Принципы удаленного вызова процедур (RPC)
- Объектно-ориентированное ПО среднего слоя
- Разработка с помощью ОО ПО среднего слоя
- Литература

Язык определения интерфейсов (IDL)

- Все реализации ПО среднего слоя реализуют Язык определения интерфейсов
- Сверх того, что предложено в RPC, IDL ОО ПО среднего слоя поддерживает объектные типы в роли параметров, обработку ошибок и наследование
- ОО ПО среднего слоя предоставляет компиляторы IDL, создающие клиентские и серверные стабы для реализации уровня сеанса и представления модели ISO/OSI

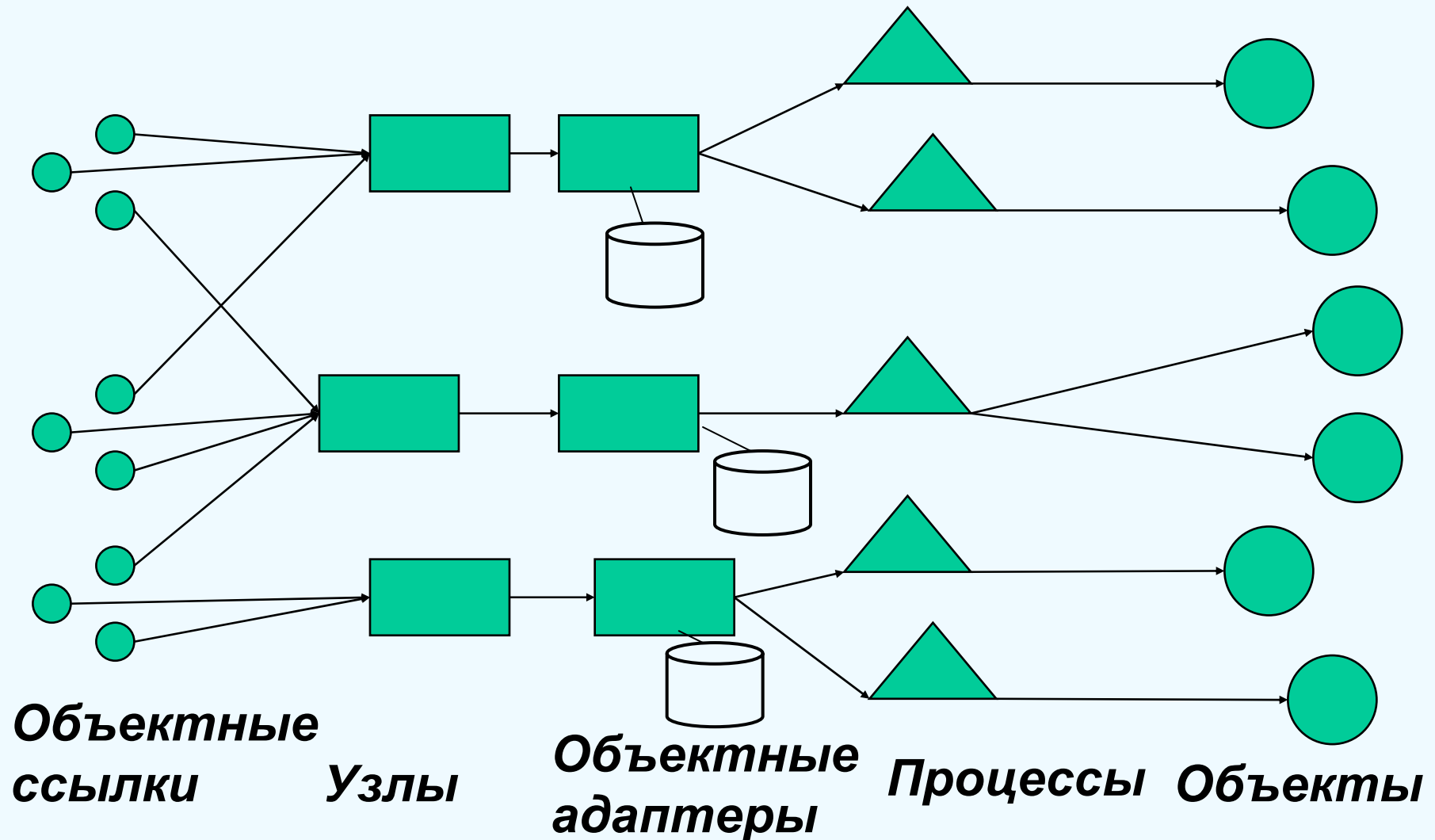
Пример IDL

```
interface Player : Object {  
    typedef struct _Date {  
        short day; short month; short year;  
    } Date;  
    attribute string name;  
    readonly attribute Date DoB;  
};  
  
interface PlayerStore : Object {  
    exception IDNotFound{};  
    short save (in Player p);  
    Player load(in short id) raises(IDNotFound);  
    void print(in Player p);  
};
```

Реализация уровня представления

- В дополнение к уровню представления, реализованному в RPC, ОО ПО среднего слоя:
 - Определяет представление объектных ссылок в транспортном слое
 - Поддерживает исключения
 - Выполняет маршалинг наследованных атрибутов

Реализация уровня сеанса



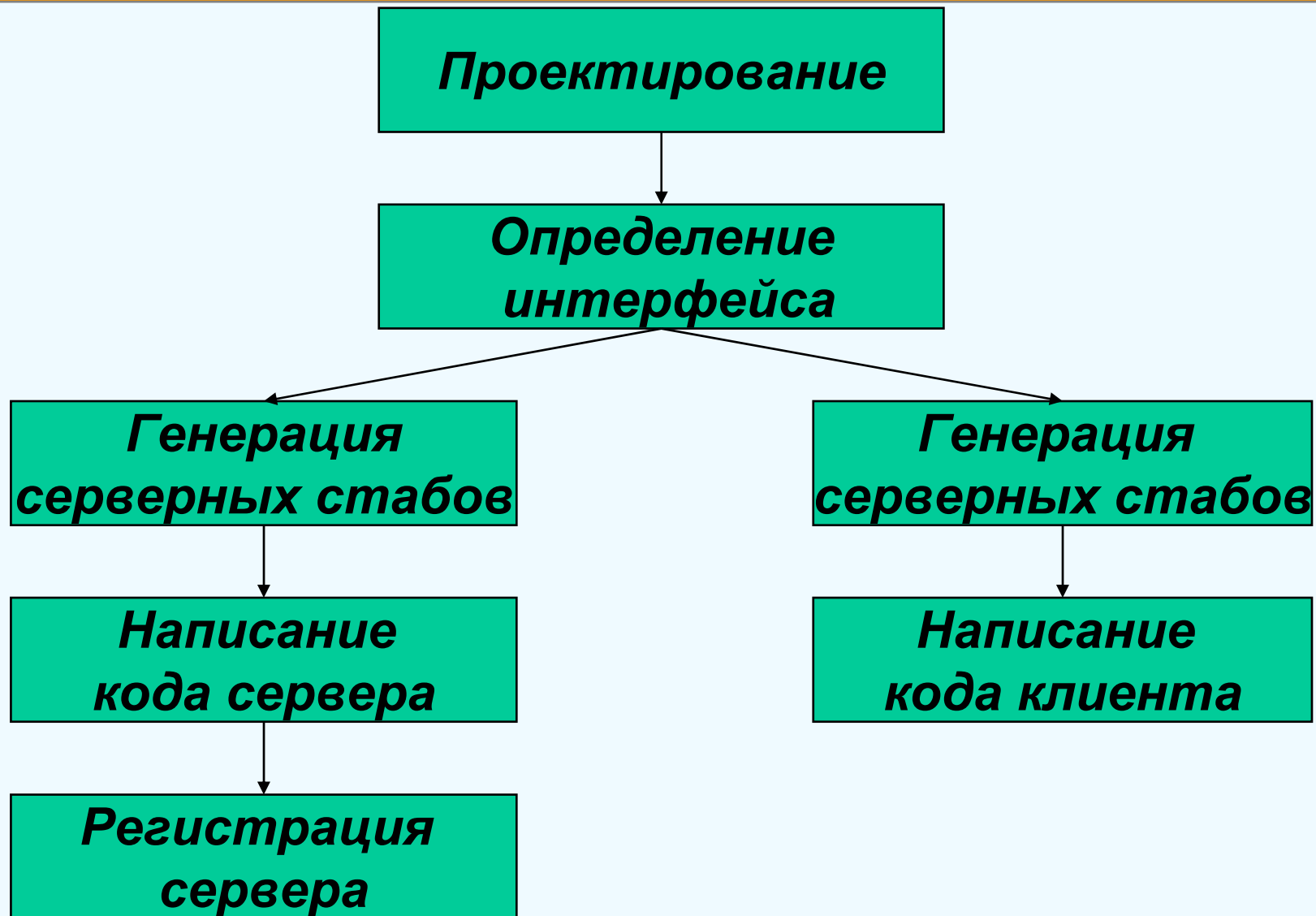
Реализация уровня сеанса

- **Задачи, решаемые уровнем сеанса:**
 - **Отображение объектных ссылок в узлы**
 - **Реализация примитивов активации и деактивации объектов**
 - **Вызов запрашиваемой операции**
 - **Синхронизация клиента с сервером**

План лекции

- Определение и типы ПО среднего слоя (middleware)
- Принципы удаленного вызова процедур (RPC)
- Объектно-ориентированное ПО среднего слоя
- Разработка с помощью ОО ПО среднего слоя
- Литература

Шаги разработки



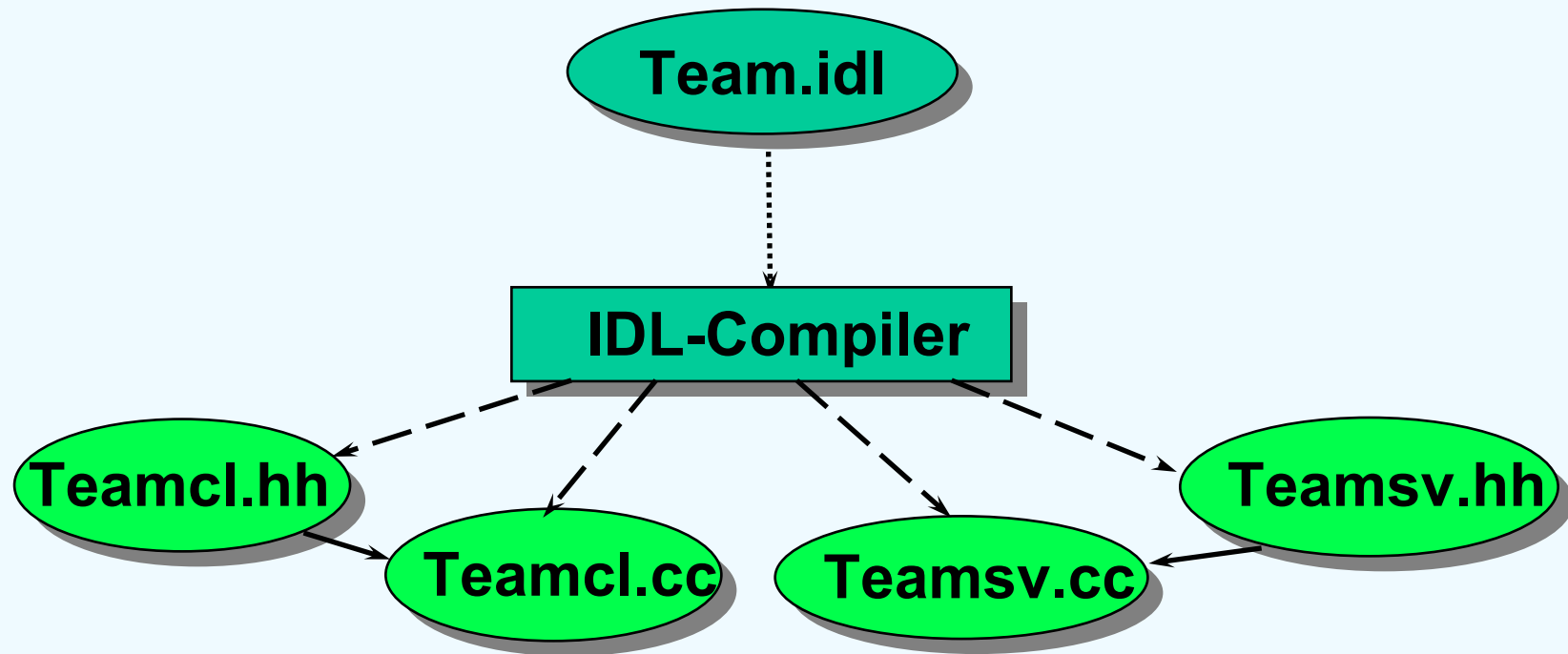
Обеспечение прозрачности доступа

- Стабы клиента декларируют те же операции, что и серверные объекты
- Следовательно клиенты могут:
 - Выполнять локальные вызовы клиентских стабов
 - Или локальные вызовы серверных объектов
(Без изменения кода)
- ПО среднего слоя может ускорить выполнение запросов не используя клиентские и серверные стабы

Обеспечение прозрачности расположения

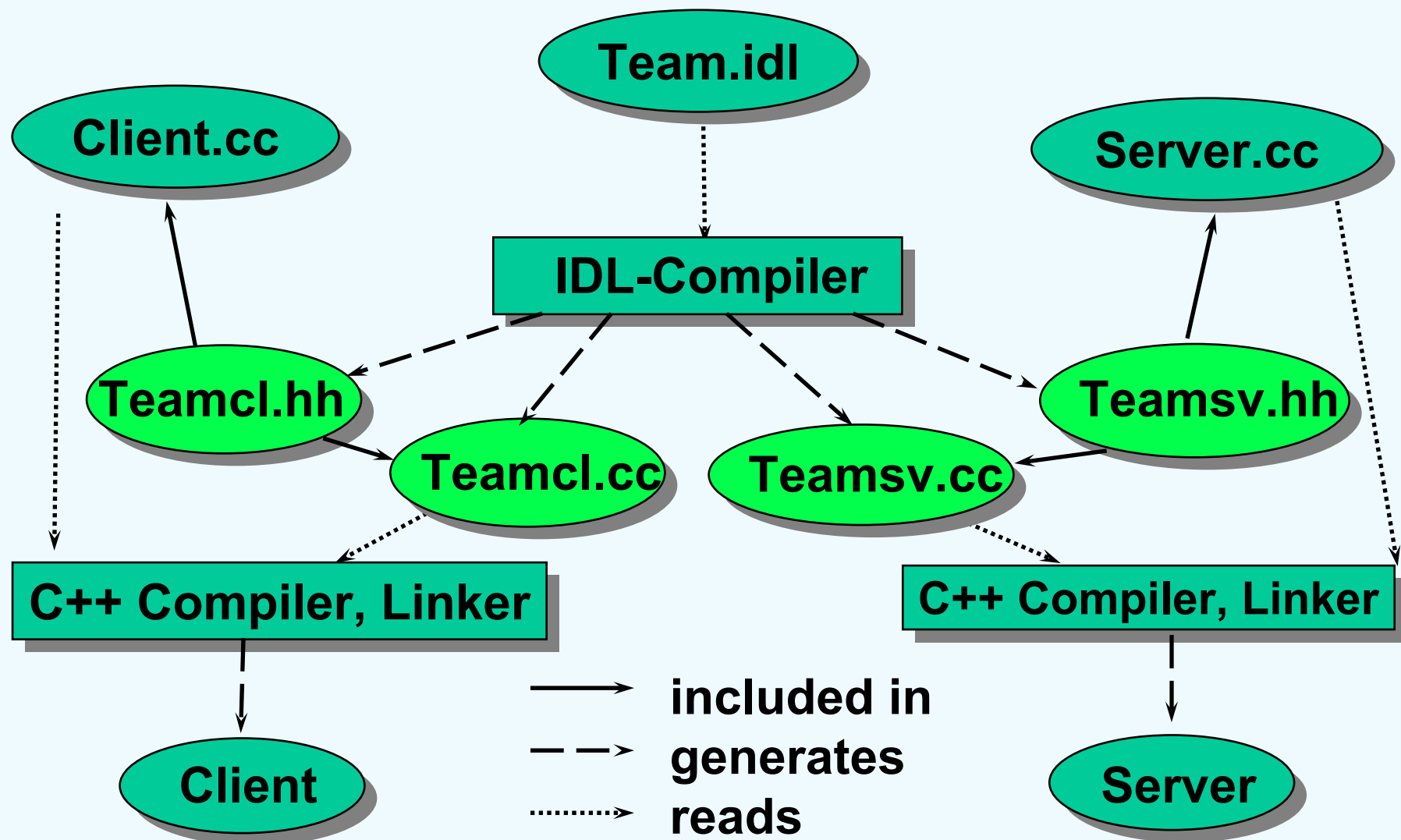
- **Идентификатор объекта**
- **Ссылки на объект**
- **Клиент запрашивает выполнение операции у серверного объекта, идентифицированного объектной ссылкой**
- **Информация о физическом расположении объекта не нужна**
- **Как получить объектные ссылки?**

Генерация стабов



——> included in
- - -> generates
.....> reads

Реализация клиента и сервера



Регистрация на сервере

- Объектные адаптеры должны иметь возможность нахождения и запуска серверов
- Объекты сервера регистрируются в «репозитории реализации» (реестре, и т.п.)
- Процессы регистрации зависят от ПО среднего слоя и продукта
- Объектный адаптер до выполнения запуска серверного объекта обращается к «репозиторию реализации»

Литература / Internet ИСТОЧНИКИ

- В. Эммерих ***Конструирование распределенных объектов.*** - М.:Мир. - 2002.
- М.Р. Когаловский ***Энциклопедия технологий баз данных.*** - М.: ФС. - 2002.
- Ю.А. Григорьев, А.Д. Плутенко. - ***Жизненный цикл проектирования распределенных баз данных.*** - Благовещенск. - 1999.
- www.omg.org