Сервер приложений C++

Андрей Шетухин
Rambler Internet Holding

Сервер приложений C++

DevPoint

- первые наработки - 2003 год
- нагруженный проект, много запросов, мало памяти, мало CPU
- компьютеры выросли, но выросли и нагрузки; ничего не изменилось
- новые требования: модульность, упрощение API, переносимость
- необходимость поддержки инструментария Web-2.0 (AJAX, XML, JSON)

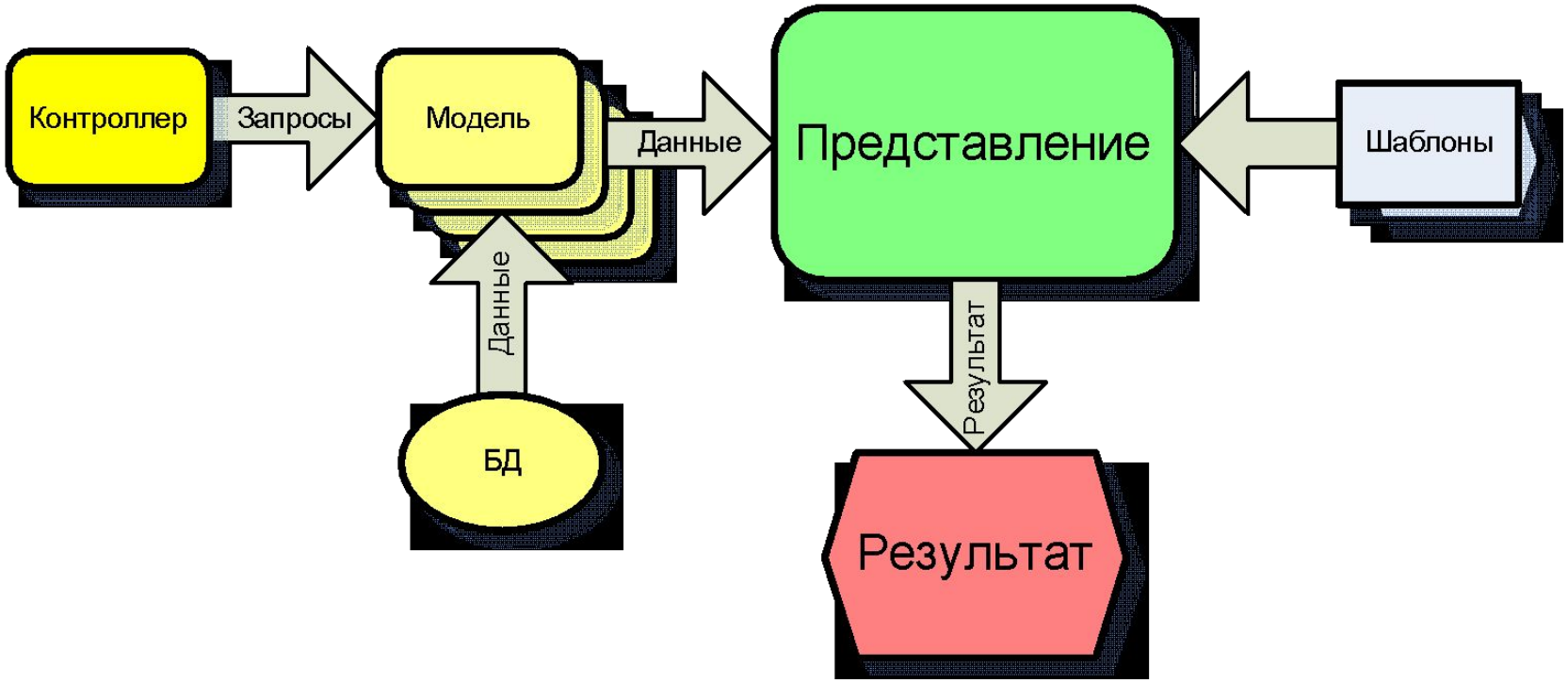
DevPoint

Парадигма MVC

- что такое MVC и зачем оно нужно?
- необходимые модификации
- достоинства предложенной схемы
- архитектура CAS

DevPoint

Классическая архитектура MVC



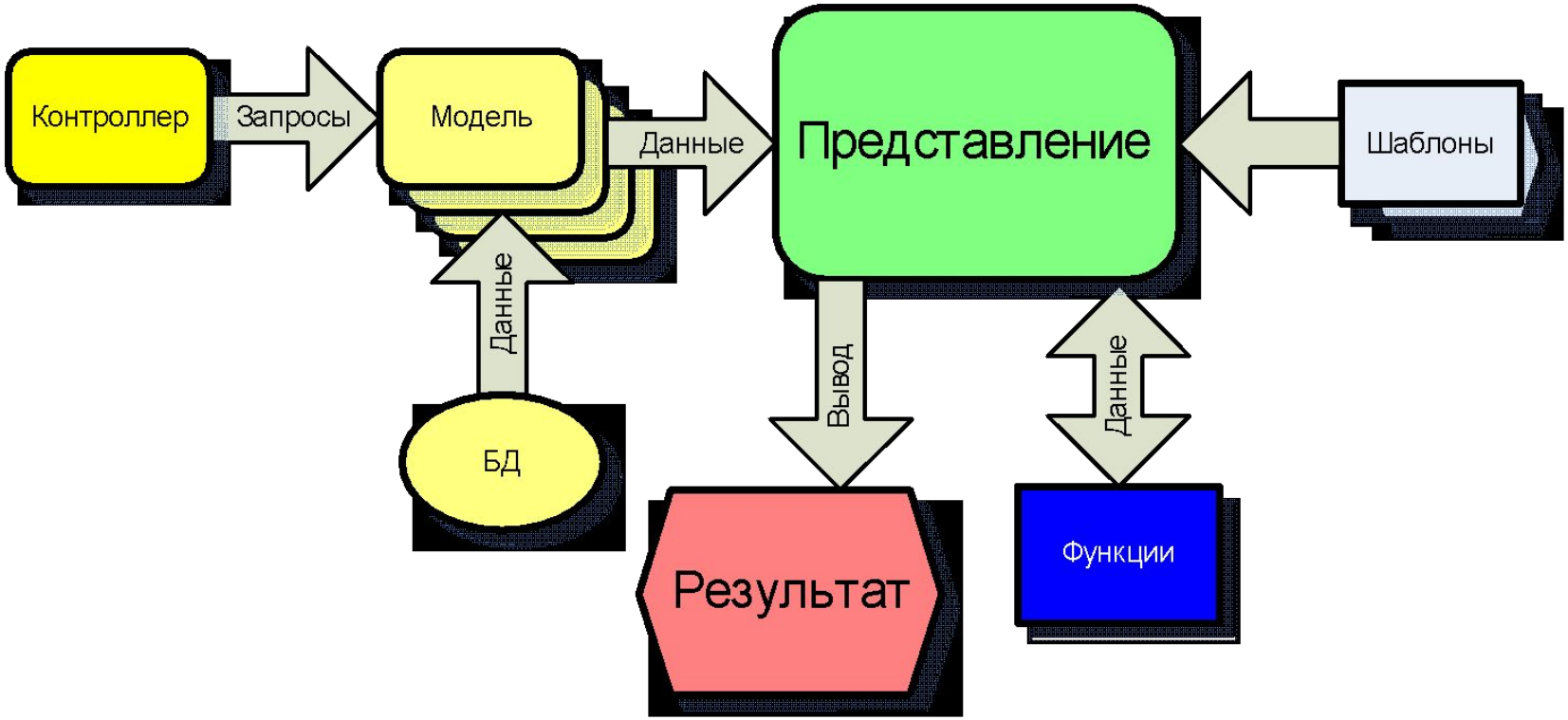
DevPoint

Критика

- плохо работает под большой нагрузкой
- сложность разработки моделей, контроллеров и представлений
- ненужный код в моделях
- проблемы с масштабированием

DevPoint

Модификация MVC (mMVC)



DevPoint

Достоинства mMVC

- модели – универсальные
- формирование ответа – только в представлении
- для генерации HTML/JSON/XML кроме шаблона и View ничего не требуется
- простота внесения правок
- высокая скорость работы

DevPoint

CAS сегодня

- динамично развивается
- проверен временем
- текущая версия – 3.3.X
- распространяется как Open Source продукт
- лицензия – BSD
- широкий набор модулей для работы с СУБД, memcached, POP3/IMAP и т.п.

Сервер приложений C++

DevPoint

CAS обеспечивает

- модульность
- взаимную изоляцию сущностей
- интегрируемость с другими технологиями
- универсальность кода
- расширяемость
- простоту сопровождения проекта
- низкую стоимость разработки

DevPoint

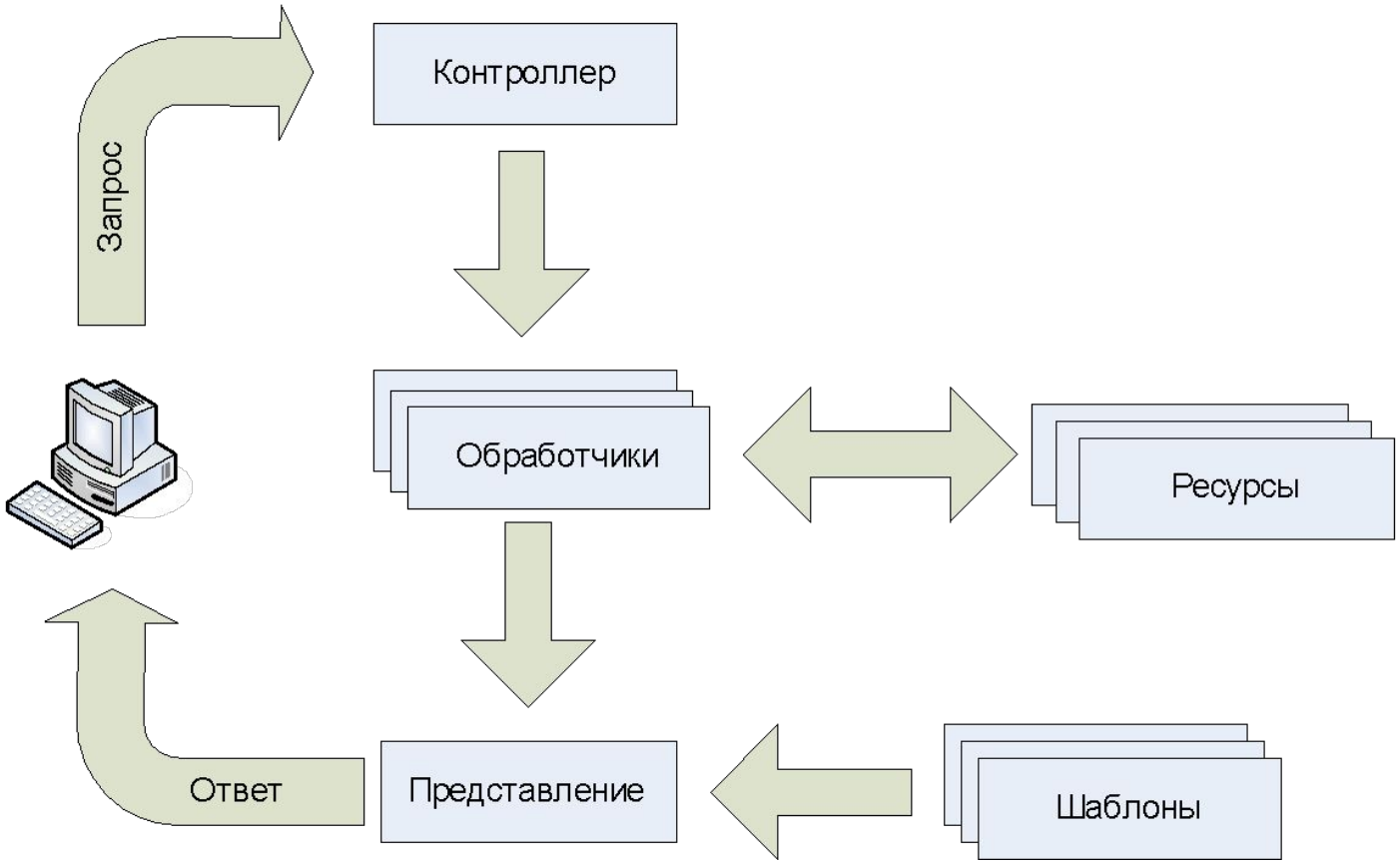
Интерфейсы

- клиентские: HTTP, XML-RPC, AJAX(XML, JSON)
- серверные: Apache 1.3 и 2.X, FastCGI
- пользовательские: API для создания собственных модулей и плагинов

Сервер приложений C++

DevPoint

Архитектура CAS

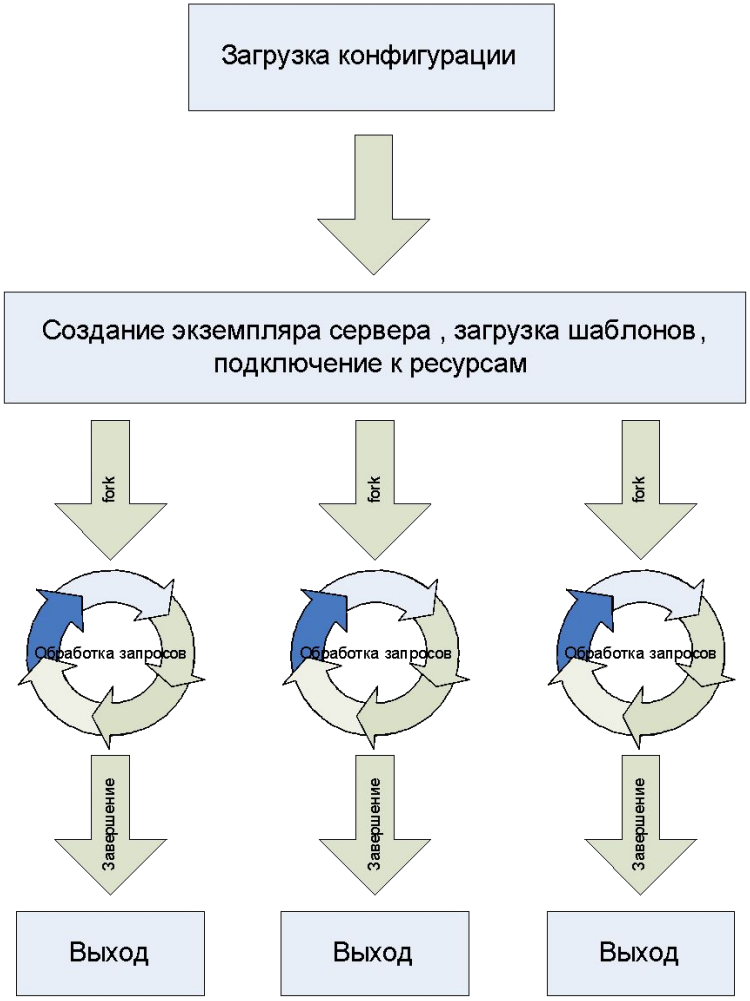


Сервер приложений C++

DevPoint

Жизненный цикл

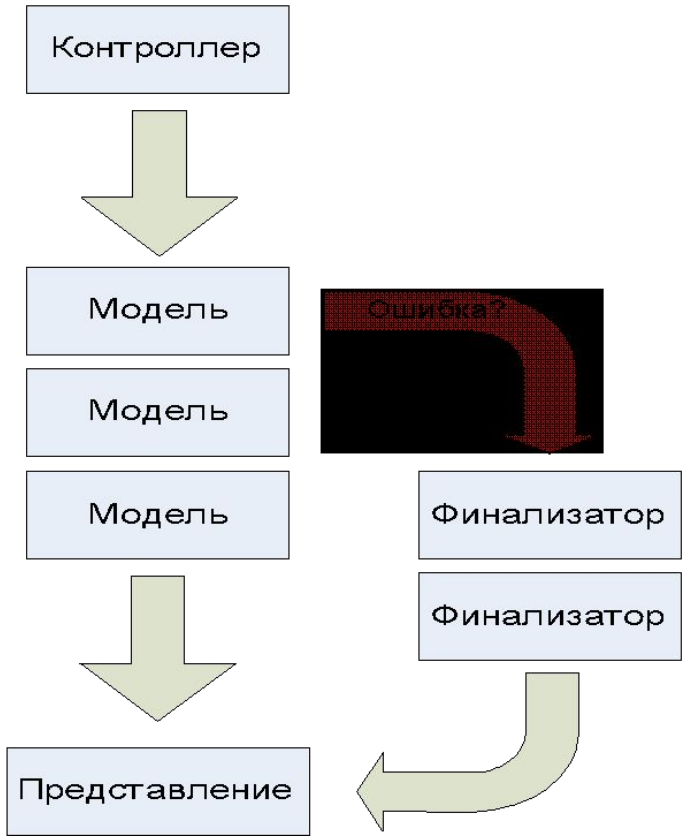
- загрузка конфигурации
- загрузка модулей
- создание сервера приложений
- обработка запросов
- выход



DevPoint

Обработка запроса

- контроллер исполняется первым
- модели запускаются последовательно
- если возникла ошибка – работают финализаторы
- представление формирует данные



DevPoint

От слов – к делу!

- необходимый инструментарий
- пишем “Hello, World!”
- тестируем результаты работы
- пример посложнее – лента новостей
- сравним с mod_perl
- и с PHP – тоже сравним

Сервер приложений C++

DevPoint

Инструменты

- компилятор C++
- система сборки cmake
- сервер приложений C++
- 10 минут свободного времени

DevPoint

Hello, World!

- создаем модуль

```
cas-xt -t handler -g -n Hello

Using templates from directory
"/usr/share/cas/xt"
Output directory is "."
Creating [DIR]  Hello
Creating [DIR]  Hello/include
Creating [DIR]  Hello/src
Creating [FILE] Hello/src/Hello.cpp
Creating [FILE] Hello/CMakeLists.txt
```

DevPoint

Hello, World!

- пишем код

```
INT_32 Hello::Handler(CTPP::CDT & oData,
ASRequest & oRequest, ASResponse & oResponse,
ASLogger & oLogger)
{
    // Put your code here
    oData["hello"] = "Hello, World!";
    // 200 OK
    oResponse.SetHTTPCode(200);
    // Write to log
    oLogger.Debug("Hello!");
return HANDLER_OK;
}
```

DevPoint

Hello, World!

- создаем шаблон

```
<html>
<head>
    <title>My first example</title>
</head>
<body>
   <TMPL_var hello>
</body>
</html>
```

DevPoint

Hello, World!

- проверяем результат

```
lynx -source http://localhost/hello.html

<html>
<head>
        <title>My first example</title>
</head>
<body>
        Hello, World!
</body>
</html>
```

DevPoint

Hello, World!

- тоже саме – на Perl

```
package CAS::Hello;
use strict;
use Apache::Constants qw(:common);

my $T = new HTML::CTPP2();
my $B = $T -> parse_template('news.tmpl');
sub handler
{
    my $r = shift;
    $r -> content_type('text/html');
    $T -> param({hello => 'Hello, World!'});
    print $T -> output($B);
return OK;
}
1;
```

DevPoint

Hello, World!

- и на PHP

```
<?php
  $T = new ctpp();

  $Bytecode = $T -> parse_template("hello.tmpl");

  $T -> emit_params(Array(hello => "Hello, World!"));

  echo $T -> output($Bytecode);
?>
```

DevPoint

Пример посложнее

- Лента новостей

```
SQL::NonTransaction oNT =
GetSQLConnector(oGlobalPool).NewNonTransaction();

oData["newslist"] =
   NTSQLayerCDT(oNT, "SELECT * FROM news ORDER BY date")
<< SQL::GetRowMap;
```

DevPoint

Интеграция

- единые шаблоны для всего проекта, независимо от “движка”
- простота миграции между технологиями
- высокая скорость прототипирования и разработки
- поддержка популярных языков и сред: Perl, PHP, Python

DevPoint

Платформы и архитектуры

- Linux
- FreeBSD
- Solaris

- i386
- amd64
- UltraSPARC

DevPoint

Развитие проекта

- поддержка SOAP
- Web Sockets
- модули для работы с разнообразными поставщиками данных
- поддержка вставок кода на Lua, Python и PHP

Сервер приложений C++

DevPoint