СОЗДАНИЕ СЕРВИС-ОРИЕНТИРОВАННОЙ СИСТЕМЫ УДАЛЕННОЙ ВИЗУАЛИЗАЦИИ СЛОЖНЫХ 3D-МОДЕЛЕЙ ДЛЯ СИСТЕМЫ CAEBEANS

Д.А. Диков

Введение

На сегодняшний день основная вычислительная нагрузка при постановке виртуальных экспериментов посредством систем инженерного моделирования (CAE (Computer-Aided Engineering) – технология применения программ и программных пакетов, предназначенных для решения различных инженерных задач: расчётов, анализа и симуляции физических процессов) все чаще переносится на высокопроизводительные суперкомпьютерные системы.

Существует целый ряд широко используемых CAE-систем, таких как ANSYS, DEFORM, FlowVision [8, 9, 11], решающих задачи инженерного моделирования. Для расчета инженерных задач в рамках системы CAEBeans [10] конечному пользователю предоставляется возможность работы с высокопроизводительными ресурсами распределенной грид-среды посредством простого проблемно-ориентированного пользовательского интерфейса. В основе технологии CAEBeans лежит обеспечение сервис-ориентированного предоставления программных ресурсов базовых компонентов CAE-систем и формирование иерархий проблемно-ориентированных оболочек, инкапсулирующих процедуру постановки и решения определенного класса задач. Технология CAEBeans регламентирует процесс декомпозиции задачи в иерархию подзадач [12]:

1. поиск вычислительных ресурсов;
2. сопоставление задачам соответствующих базовых компонент CAE-систем;
3. мониторинг хода решения задач;
4. передача результатов решения задач пользователю.

Большой объем данных, полученных в результате экспериментов, приводит к сложностям в процессе их обработки на пользовательском ПК. Возникает необходимость быстрой визуализации результатов моделирования без необходимости полного копирования всего объема полученных результатов на компьютер пользователя [1, 3, 5, 6]. Это позволило бы проводить эксперименты даже с ПК или мобильной системы без специального ПО посредством доступа к облачной CAE-системе посредством сети Интернет.

В качестве альтернативы разработанному решению [7] в текущей работе предложен поход на основе технологии REST [2, 4]. Такой подход позволяет увеличить быстродействие системы за счет меньшего объема служебной информации передаваемой между клиентом и сервером.

Архитектура системы удаленной визуализации

В данной работе рассматривается сервис-ориентированная архитектура системы удаленной визуализации 3D-моделей, состоящая из следующих основных компонентов (Рис. 1):

1. клиентское веб-приложение;
2. веб-служба;
3. система визуализации.

В качестве клиента может выступать любое приложение, совместимое с веб-службой. В данной работе в качестве клиентского веб-приложения выступает веб-браузер. Сервер образован веб-службой и системой визуализации. Веб-служба взаимодействует с клиентом, используя протокол REST. Система визуализации генерирует изображение на основе построенной 3D-модели.

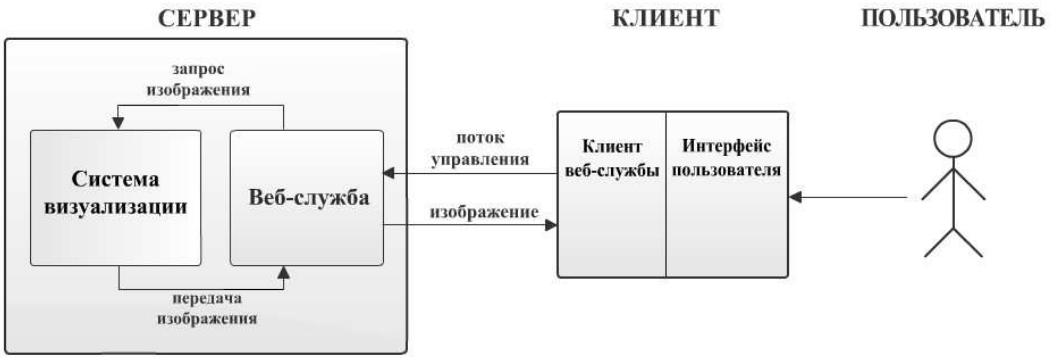

Рис. 1. Архитектура системы удаленной визуализации.

Команды на изменение положения камеры передаются на сервер в виде запросов, формирующих поток управления, клиентом веб-службы. Веб-служба принимает и анализирует запросы, которые затем передаются

системе визуализации. Сгенерированное системой визуализации изображение передается клиенту и отображается в пользовательском интерфейсе.

Веб-служба представляет собой приложение, развернутое на платформе Ruby on Rails, что позволяет организовать REST-интерфейс взаимодействия с сервером таким образом, что запрос передается напрямую через адресную строку и не требует передачи объемного XML-пакета. REST-запрос представляет собой обращение к действию Rails-контроллера с набором задаваемых параметров, при этом используемый HTTP-метод воспринимается как глагол, уточняющий смысл запроса.

Запрос на перемещение камеры и генерацию изображения выглядит следующим образом:

POST /visroom/3/move

- POST – HTTP-метод, воспринимаемый как глагол “записать (данные) ”;
- visroom – контроллер, обрабатывающий запрос;
- move – действие контроллера, которое выполнится в результате обработки запроса;
- “3” – параметр, передаваемый действию (для действия move — идентификатор направления движения камеры) .

Посредством REST-интерфейса система предоставляет ряд сервисов, доступных клиентскому приложению:

- авторизация в системе удаленной визуализации;
- выбор определенной модели, доступной для визуализации;
- перемещение камеры и генерация изображения;

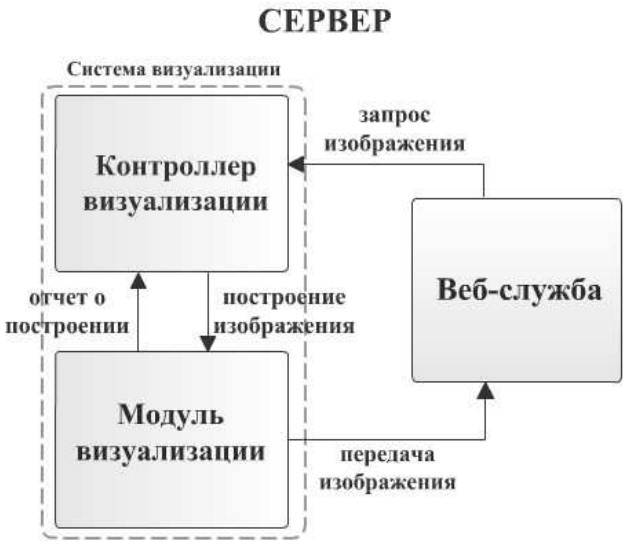

Рис. 2. Архитектура сервера.

Система визуализации включает в себя два компонента (Рис. 2):

1. Контроллер визуализации;
2. Модуль визуализации.

Контроллер визуализации представляет собой Rails-контроллер, выполняющий пересчет координат камеры и передачу обновленных координат модулю визуализации. Модуль визуализации строит 3D-сцену и на ее основе генерирует изображение.

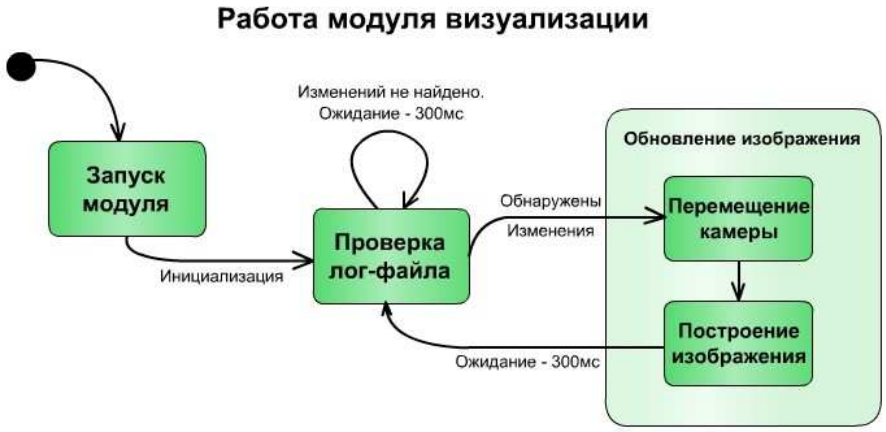

Рис. 3. Работа модуля визуализации.

Взаимодействие модуля визуализации с контроллером визуализации происходит посредством лог-файла, содержащего координаты камеры. Контроллер обновляет лог-файл, записывая обновленные координаты. Модуль визуализации циклически прочитывает данные лог-файла и, в случае обнаружения обновленных данных, выполняет генерацию изображения. Принцип работы модуля визуализации отображен на Рис. 3.

Реализация и тестирование рабочего прототипа

Реализованный прототип системы использует в качестве клиентского веб-приложения HTML-страницу, выполняющую запросы серверу асинхронно, используя технологию AJAX. Так как концептуально сервер и клиент размещены удаленно друг от друга, то необходимо выполнять кросс-доменные запросы, что становится возможно при использовании функции XMLHttpRequest2, на данный момент не включенной в стандарт W3C.

Веб-служба, принимая запрос на генерацию изображения от клиентского приложения, направляет команды конкретному модулю визуализации, обрабатывающим выбранную пользователем модель, согласно данным пользовательской сессии, хранящимся в базе данных *sessions* Rails-приложения. Сгенерированное изображение размещается директории /public в уникальной папке пользователя. Внешний доступ к этой папке не контролируется веб-службой, что повышает быстродействие системы в целом. Обновление изображения на клиентской стороне происходит посредством перезагрузки изображения с сервера, с помощью URL, передаваемого сервером в качестве ответа на запрос.

Для демонстрации работы системы удаленной визуализации, компоненты системы были размещены на одном компьютере под управлением операционной системы Windows XP. Клиентская страница была размещена по адресу http://localhost/client.html. Сервер был запущен средствами сервера Rails-приложений WEBrick и находился по адресу http://localhost:3000/vs.

Для подключения необходимо заполнить поля “логин” и “пароль”, после чего нажать кнопку “вход”. После этого будет осуществлен запрос серверу на аутентификацию, содержащий введенные логии и пароль пользователя. Этот запрос будет направлен контроллеру login, предоставляющему действие login, которое отвечает за аутентификацию пользователей. Сервер отвечает html-кодом, содержащим список моделей, доступных для визуализации. Далее необходимо выбрать модель из списка и нажать кнопку “запустить”. Будет сформирован запрос серверу на запуск модуля визуализации и генерацию изображения, содержащий идентификатор выбранный модели, направляемый действию *init:*

```
GET /visroom/1/init
```

В случае успешной обработки запроса, сервер запускает модуль визуализации и загружает модель с выбранным идентификатором, а клиент отображает навигационную панель, содержащую кнопки “Вверх”,“Вниз”,“Влево” и “Вправо” и загружает сгенерированное изображение с сервера. При нажатии пользователем любой из навигационных кнопок, клиент выполняет запрос на перемещение камеры и генерацию изображения. После обработки запроса на сервере, клиент отобразит сгенерированное изображение, обновив пользовательский интерфейс. Результат обработки запроса и визуализации приведен на Рис. 4.

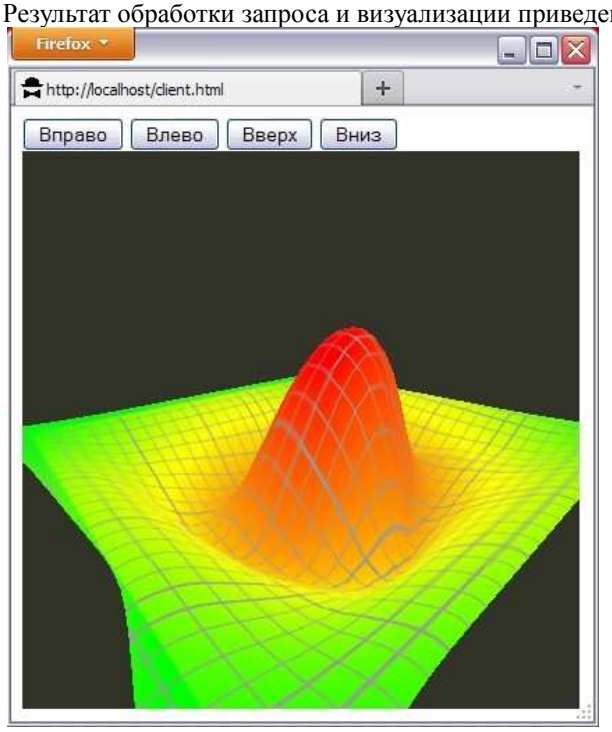

Рис. 4. Результат визуализации.

Заключение

В статье приведена сервис-ориентированная архитектура системы удаленной визуализации сложных

3D-моделей, а также были предложены способы организации интерфейса взаимодействия клиента и сервера по протоколу REST и архитектура для компонентов, входящих непосредственно в систему визуализации. Также был разработан и протестирован рабочий прототип системы удаленной визуализации, позволяющий выполнять поставленную в рамках данной работы задачу визуализации с приемлемым быстродействием.

Проект выполнен при поддержке Совета по грантам Президента Российской Федерации (номер проекта МК-1987.2011.9) и Российского фонда фундаментальных исследований (проект № 11-07-00478).

ЛИТЕРАТУРА:

1. Ammirati P., Clematis A., D'Agostino D., Gianuzzi V. Using a Structured Programming Environment for Parallel Remote Visualization // Euro-Par, 2004. LNCS, Vol. 3149, P. 477–486.
2. Boyer J. M., Wiecha C. F., Akolkar R. P., A REST protocol and composite format for interactive web documents // DocEng '09 Proceedings of the 9th ACM symposium on Document engineering, P. 139-148, 2009.
3. Diepstraten J., GoËrke M., Ertl T. Remote Line Rendering for Mobile Devices // Computer Graphics International, 2004. P. 454-461.
4. Ohara M. Aggregating REST requests to accelerate Web 2.0 applications // IBM Journal of Research and Development, Vol. 54, Issue 1, P.91-102, 2010.
5. Stefen L., Oliver M. A CUDA-Supported Approach to Remote Rendering // ISVC 2007. Part I, LNCS, Vol. 4841, P. 724–733
6. Stegmaier S., Magallyn M., Ertl T. A Generic Solution for Hardware-Accelerated Remote Visualization // IEEET CVG Symposium on Visualization, 2002.
7. Бахтерев М.О., Васев П.А., Казанцев А.Ю., Манаков Д.В. Система удаленной визуализации для инженерных и суперкомпьютерных вычислений // Вестник ЮУрГУ -2009 -№17(150) – с. 4-11.
8. Долганина Н.Ю., Сапожников С.Б., Маричева А.А. Моделирование ударных процессов в тканевых бронежилетах и теле человека на вычислительном кластере "СКИФ Урал" // Вычислительные методы и программирование: Новые вычислительные технологии Т. 11. 2010. С. 117-126.;
9. Московский А.А., Перминов М.П., Л.Б. Соколинский, Черепенников В.В., Шамакина А.В. Опыт использования суперкомпьютера "СКИФ Аврора" для решения научно-технических задач // CAD/CAM/CAE Observer. 2010. № 3. С. 1-7.;
10. Радченко Г.И., Соколинский Л.Б. Технология построения виртуальных испытательных стендов в распределенных вычислительных средах // Научно-технический вестник Санкт-Петербургского государственного университета информационных технологий, механики и оптики. № 54. 2008
11. Радченко Г.И., Соколинский Л.Б., Шамакина А.В. Разработка компонентно-ориентированных CAEBean-оболочек для пакета ANSYS CFX // Параллельные вычислительные технологии: Труды международной научной конференции (28 января - 1 февраля 2008 г., г. Санкт-Петербург). Челябинск: Изд-во ЮУрГУ. 2008. С. 438-443.
12. Радченко Г.И. Технология построения проблемно-ориентированных иерархических оболочек над инженерными пакетами в грид-средах // Системы управления и информационные технологии. 2008. № 4(34).