Студент: Иванов Иван Иванович, ВМИ-399

Руководитель: Петров Петр Петрович

Тема Разработка сервис-ориентированной системы удаленной визуализации сложных 3d-моделей

**Анализ предметной области и существующих работ по тематике курсового проекта**

1. **Предметная область проекта**

Основная задача заключается в построении системы удаленной визуализации и веб-службы на стороне сервера, декларирующей API и требования для построения клиентских приложений. В рамках текущей работы предлагается следующее решение поставленной задачи: система состоит из трехступенчатого конвейера, состоящего из системы визуализации, расположенного на высокопроизводительном сервере, веб-службы, осуществляющей связь “сервер-клиент”, и, непосредственно, клиента.

1. **Анализ аналогичных проектов и существующих решений для реализации проекта**

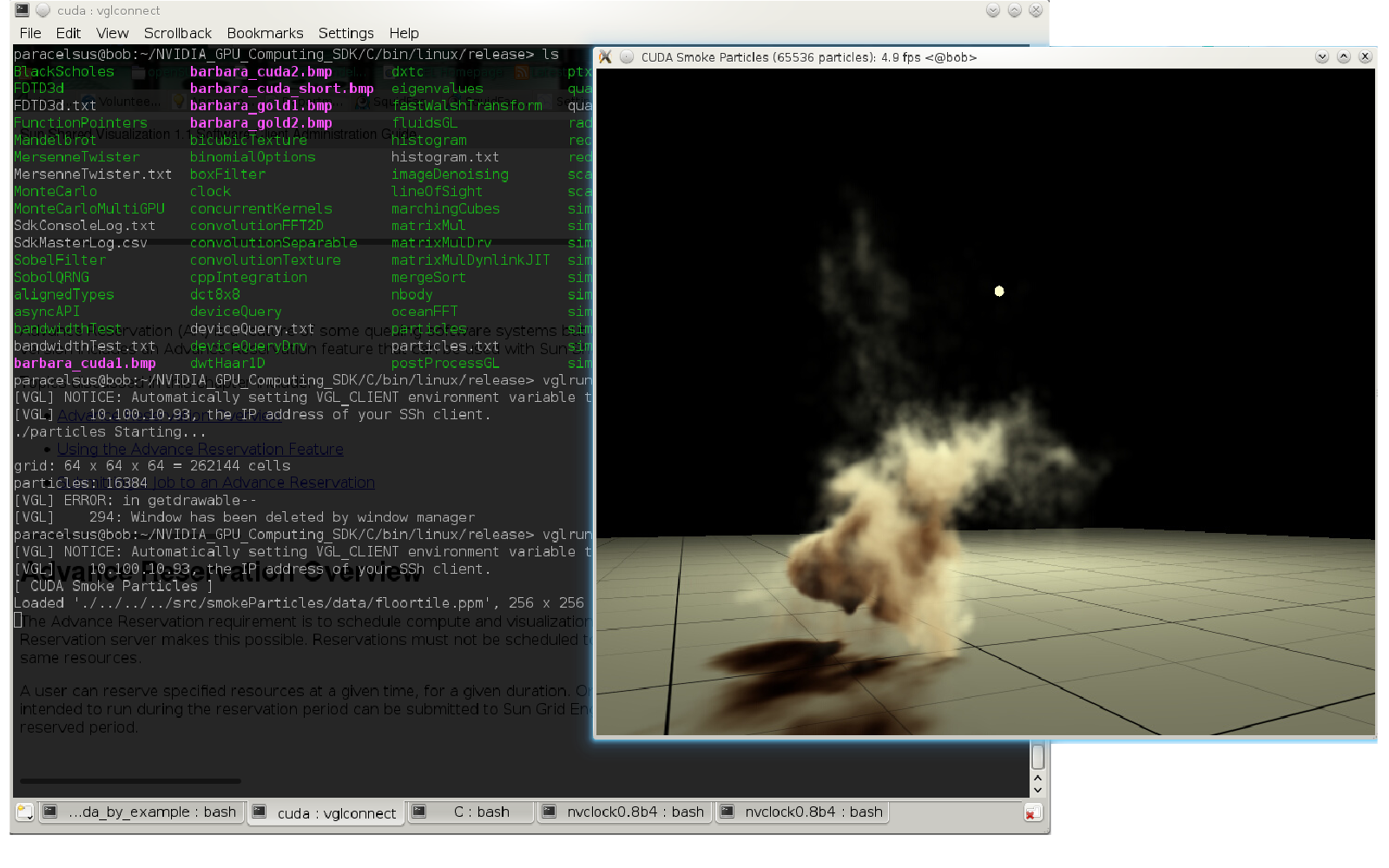
В течение последнего десятилетия многие исследователи занимались задачей удаленной визуализации, применительно к широкому кругу устройств и различных задач [2, 6, 12].

Общая концепция, в том числе архитектура удаленного сервера, получающего запрос на визуализацию, и клиента, формирующего запрос и получающего набор изображений от сервера, наиболее полно освещена в работе [9]. Подобный подход с минимальными изменениями лежит в основе большинства реализованных систем.

Альтернативный подход, описанный в работе [5], предполагает передачу клиенту не результирующего изображения, а набора примитивов (линий), сформированного сервером, на основе которого устройство-получатель самостоятельно производит построение изображения.

Реализации рассматриваемой концепции в основном используют программный интерфейс OpenGL, реже – собственные разработки. Использование OpenGL [9, 10, 14, 19] предоставляет возможность создания аппаратно-независимой системы.

Также были предложены методы удаленной визуализации для распределенных систем [7, 10], рассматривающие особенности концепции в условиях существования множества серверов, обрабатывающих исходный материал для визуализации. В статье [2] особое внимание уделено проблемам взаимодействия удаленных серверов при формировании результирующего изображения, а в работе [4] рассматривается механизм компрессии готового изображения с потерями качества для оптимизации работы системы.



**Рис. 1.** VirtualGL. Пример визуализации.

VirtualGL [20] (Рис. 1) - еще одно очень популярное решение, ориентированное на системы UNIX\Linux. Это программная система, предназначенная для перенаправления команд 3D-рендеринга OpenGL на аппаратный 3D ускоритель, располагающийся на выделенном сервере. Результаты визуализации интерактивно отображаются на других компьютерах в сети с помощью тонкого клиента.

##### Система OnLive

Примером такой задачи, в которой необходимо построение сложного изображения в реальном времени, являются компьютерные игры. Идея разделить выполнение программы и игрока впервые была предложена компанией OnLive [11, 18], с 2000 года разрабатывающей одноименный интернет-сервис цифровой дистрибуции компьютерных игр. В основу сервиса положен принцип, согласно которому пользователь удаленно запускает на выделенной части сервера необходимое приложение, а результат его работы, т.е. изображение получает в виде потока видео, который непрерывно формируется и посылается сервером, в свою очередь принимающим команды от игрового контроллера пользователя (пример диаграммы взаимодействия приведен на Рис. 2).

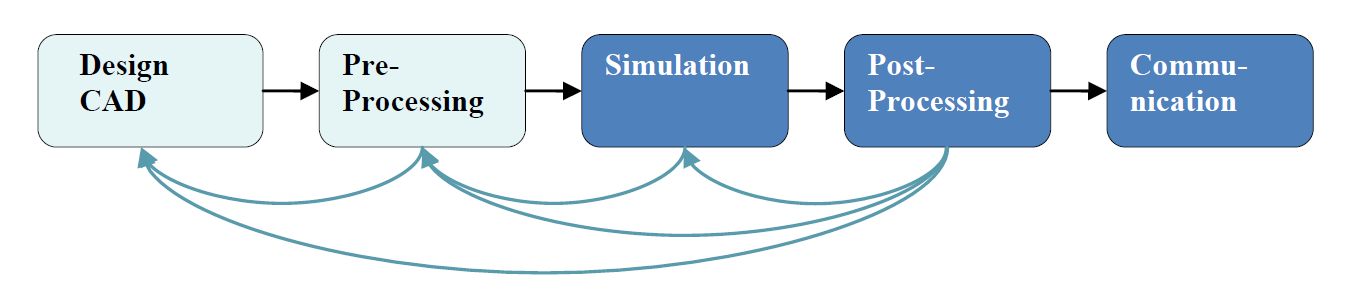
##### course_onlive

**Рис. 2.** Процесс работы системы OnLive.

Критичные для данной области проблемы, такие как скорость отклика, решаются с помощью высокого уровня компрессии видео, достижимого благодаря специальному видеокодеку, реализация и принцип работы которого не афишируется. Схожим образом организована архитектура сервиса Gaikai [17] , прямого конкурента OnLive.

##### Системы визуализации инженерных данных

CAE (Computer-aided engineering) – это общий термин для ряда программ и программных пакетов, предназначенных для решения различных инженерных задач: расчётов, анализа и симуляции физических процессов. CAE-системы позволяют при помощи расчётных методов оценить, как поведёт себя компьютерная модель изделия в реальных условиях эксплуатации.



**Рис. 3.** Схема CAE-процесса.

Поток решения задачи инженерного моделирования посредством CAE-пакета (Рис. 3):

* CAD-дизайн – моделирование трехмерной геометрии
* Препроцессинг –обработка и оптимизация геометрии
* Моделирование – создание имитационной модели
* Симуляция – генерация 3D-модели и симуляция моделируемых процессов
* Пост-процессинг – обработка полученных результатов
* Коммуникация – передача обработанных результатов в систему визуализации

Существует целый ряд CAE-систем, таких как ANSYS, DEFORM, FlowVision [15, 16, 21], чаще всего слишком объемных и сложных для построения на ПК, и тем более на ноутбуке или другом, менее производительном мобильном устройстве. Для решения этой и подобных задач были предложен ряд решений [8, 14].

**Особенности реализации сервис-ориентированных систем удаленной визуализации**

Для создания системы визуализации в работе [14] используется архитектура, состоящая из высокопроизводительного сервера, выполняющего работу по созданию изображения (модуль визуализации), сервиса-посредника и интерфейса пользователя. Сервис-посредник в данном случае работает по протоколу SOAP. Изображение первоначально передается клиенту в низком качестве, затем, при отсутствии новых команд управления от пользователя, передается изображение более высокого качества.

Взаимодействие с удаленной системой должно осуществляться посредством протокола, обеспечивающего быстрый обмен данными между клиентом и сервером. На данный момент широко используются протоколы SOAP и REST. Протокол SOAP является действующим стандартом W3C, применяемым в приложениях, осуществляющих активную передачу данных. REST (Representational State Transfer) – ориентирован на скорость обмена и использует меньшие по размеру пакеты, в сравнении с протоколом SOAP. URL интерпретируется как ресурс и действие над ним, а методы HTTP-запросов – как глаголы, уточняющие смысл запроса.

Для создания сервиса возможно использование одного из существующих фреймворков, непосредственно связанных с языком реализации – Django (Python), CakePHP(PHP), RubyOnRails (Ruby). Различные фреймворки имеют различное быстродействие и удобство разработки, что позволяет выбирать, исходя из целей и задач создаваемого сервиса. Альтернативой может послужить использование технологий Flash и Silverlight, позволяющих упростить создание клиент-серверного взаимодействия, но со значительными ограничениями в быстродействии и выборе ОС и поддерживаемых устройств.

Фреймворк RubyOnRails написан на Ruby и предоставляет возможность гибкой разработки Веб-сервисов. Язык реализации позволяет создавать сервисы, более удобные в сопровождении и масштабировании по сравнению с другими фреймворками, благодаря особенностям синтаксиса Ruby – обилию интуитивно понятных синтаксических структур, а так же развитым механизмам тестирования и сборки мусора. RubyOnRails минимизирует затраты на непосредственное написание кода и развертывание веб-сервиса. Важной особенностью в рамках создания веб-сервиса с высокими требованиями к быстродействию является ориентация Rails на использования протокола обмена сообщениями REST.

Клиент должен использовать возможность осуществления кросс-доменного асинхронного запроса. Эта опция запрещена политикой безопасности, использованной при разработке функции XMLHttpRequest, но тем не менее возможна. Для этого разработан и широко применяется ряд техник с использованием языка PHP, обходящих ограничения безопасности функции асинхронного вызова. Так как в данной разработке используется язык Ruby, то более удобно использование функции XMLHttpRequest2 [13], находящейся в разработке Консорциумом Всемирной Паутины (W3C), но уже поддерживаемой в наиболее широко используемых браузерах (Mozilla Firefox 3.5, Internet Explorer 8.0, Safari 4). Эта функция позволяет обращаться к произвольным доменам, добавляя в запрос информацию о домене, отправившем запрос, и дает возможность ограничить набор доменов, запросы от которых считаются допустимыми.

1. **Заключение**

В результате обзора литературы был выявлен набор инструментальных средств для реализации поставленной задачи, наиболее полно удовлетворяющий требованиям к подобного рода системам. Было принято решение реализовать систему на основе сервис-ориентированной парадигмы на базе технологий Ruby On Rails и AJAX.

**Литература**

1. ACM Tech Pack on Cloud Computing // ACM, 2010.
2. 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.
3. Cloud Computing: An Overview // ACM Queue, 2009.
4. Constantinescu Z., Vlădoiu M. Adaptive Compression for Remote Visualization // Universităţii Petrol – Gaze din Ploieşti, 2009. Vol. 61, №2, P. 49-58.
5. Diepstraten J., GoЁrke M., Ertl T. Remote Line Rendering for Mobile Devices // Computer Graphics International, 2004. P. 454-461.
6. Lamberti F., Zunino C., Sanna A. An Accelerated Remote Graphics Architecture for PDAs // Web3D Symposium 8th International Conference on 3D Web Technology, 2003. P. 55-62.
7. Repplinger M., LoЁﬄer A., Rubinstein D., Slusallek P. DRONE: A Flexible Framework for Distributed Rendering and Display // ISVC, 2009. Part I, LNCS 5875, P. 975–986.
8. Starvakakis J., Takastuka M. Out-of-Order Execution for Avoiding Head-of-Line Blocking in Remote 3D Graphics // PSIVT, 2007. LNCS Vol. 4872, P. 62–74.
9. Stegmaier S., Magallуn M., Ertl T. A Generic Solution for Hardware-Accelerated Remote Visualization // IEEET CVG Symposium on Visualization, 2002.
10. Stefen L., Oliver M. A CUDA-Supported Approach to Remote Rendering // ISVC 2007. Part I, LNCS, Vol. 4841, P. 724–733.
11. SuЁselbeck R., Schiele G., Becker C. Peer-to-Peer Support for Low-Latency Massively Multiplayer Online Games in the Cloud // NetGames, 2009. № 14.
12. XiaoHui L., QinPing Z., ZhiYing H., Ke X., YuBo X. Point-based Rendering Approach for Real-time Interaction on Mobile Devices // Science in China Series F: Information Sciences, 2009. Vol. 52, № 8. P. 1335-1345.
13. XMLHttpRequest Level 2 http://www.w3.org/TR/2008/WD-XMLHttpRequest2-20080930
14. 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.
15. 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. C. 117-126.;
16. 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. C. 1-7.;
17. Официальный сайт проекта Gaikai. URL: http://www.gaikai.com/.
18. Официальный сайт проекта OnLive. URL: http://www.onlive.com/.
19. Официальный сайт проекта OpenGL. URL: http://www.opengl.org/
20. Официальный сайт проекта VirtualGL. URL: http://www.virtualgl.org/.
21. 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. C. 438-443.