КОМПЛЕКС ИНФОРМАЦИОННЫХ СИСТЕМ ПОДДЕРЖКИ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ В РАМКАХ ИНФОРМАЦИОННО-ВЫЧИСЛИТЕЛЬНОГО ПОРТАЛА

А.Ю. Власенко, Н.Н. Окулов

ВВЕДЕНИЕ

В настоящее время преобладающей технологией для работы с суперкомпьютерами является ssh-доступ, предоставляющий интерфейс командной строки. Исследователю приходится изучать основы UNIX-систем, и команды для работы со специальными программными средствами. К таковым, например, относятся системы пакетной обработки. Таким образом, сложность использования и необходимость приобретения специальных знаний являются серьезной проблемой при проведении вычислений на высокопроизводительных ресурсах.

В данной статье описан типовой информационно-вычислительный портал (ИВП), представляющий собой единую среду в web-интерфейсе для работы с ресурсами центра коллективного пользования.

Один из самых затратных по времени этапов разработки приложения – отладка. Наиболее распространенный тип программных средств, служащих для поддержки пользователя в этом процессе – диалоговые отладчики. Однако эти средства, поддерживающие параллельные программы, являются дорогими коммерческими системами. Кроме того, для диалоговой отладки требуется интерактивная работа с запущенной программой. Эта работа заключается в наблюдении значений переменных по ходу исполнения. Но число процессов, а, следовательно, и экземпляров таких переменных может достигать тысяч. Поэтому выяснить, в какой ветви параллельного приложения, на каком шаге произошла ошибка крайне затруднительно.

Следовательно, отладка параллельных программ – также большая проблема, для решения которой необходимы новые подходы. В ИВП интегрирована система автоматического контроля корректности, обнаруживающая логические ошибки в MPI-приложениях без участия пользователя.

Еще одним компонентом ИВП является «Виртуальная лаборатория». Ее задача – автоматизация проведения серии вычислительных экспериментов с различными комбинациями входных параметров и выдача аналитической информации в текстовом и графическом виде (диаграммы, графики).

В работе представлен программный комплекс, содержащий перечисленные и другие информационные подсистемы.

ИНФОРМАЦИОННО-ВЫЧИСЛИТЕЛЬНЫЙ ПОРТАЛ

Основным достоинством ИВП являются его вычислительные возможности, которые включают: организацию удаленного доступа к высокопроизводительным ресурсам центра через удобный web-интерфейс, не требующий от пользователей специальных знаний, автоматизацию проведения численного эксперимента и анализ результатов вычислений, а также обеспечение дополнительных функций, упрощающих разработку параллельных приложений. ИВП выступает платформой для интеграции нескольких информационных систем, образующих единый программный комплекс, позволяющий пользователю решать широкий перечень задач по разработке и оптимизации параллельных программ.

Состав ИВП:

- система удаленного доступа и управления распределенными вычислительными ресурсами (УД и УРВР) [3];
- система отладки параллельных программ;
- система «Виртуальная лаборатория» [5];
- система мониторинга состояния вычислительных узлов;
- библиотеки параллельных программ (HydroParaLib);
- система учета процессорного времени (биллинга).

Основой ИВП является **система УД и УРВР**, предназначенная для организации проведения численного эксперимента на вычислительных ресурсах, предоставляемых по запросу пользователя в удаленном режиме.

Одним из инструментов, существенно сокращающим цикл разработки параллельного приложения, является **система отладки**, производящая анализ на наличие логических ошибок в MPI-программах.

Система «Виртуальная лаборатория» используется в качестве средства для анализа зависимости ускорения и эффективности параллельной программы от значений различных параметров задачи.

В портал интегрирована библиотека параллельных программ HydroParaLib, предназначенная для решения задач гидродинамики со свободными границами. Предусмотрен механизм подключения библиотек программ для других областей численных расчетов. Возможность использования готовых библиотек

экономит время разработки и повышает удобство работы.

Системы **мониторинга** и **биллинга** предназначены для оперативного отслеживания состояния вычислительных ресурсов и учета использованного процессорного времени соответственно и носят служебный характер.

Далее более подробно рассмотрены системы, составляющие ядро ИВП.

СИСТЕМА УДАЛЕННОГО ДОСТУПА И УПРАВЛЕНИЯ РАСПРЕДЕЛЕННЫМИ ВЫЧИСЛИТЕЛЬНЫМИ РЕСУРСАМИ

Система сочетает в себе возможности удаленного доступа и пакетной обработки заданий с графическим интерфейсом пользователя (свидетельство о государственной регистрации программы для ЭВМ №2008611206). Для обеспечения заданной функциональности в системе реализованы следующие возможности [3]:

- хранение файлов пользовательских расчетных программ в виде исходного и/или бинарного кода, файлов начальных данных, результатов и другой информации;
- передача файлов с кодами программ, начальными данными или результатами между пользовательским приложением и базой данных (БД);
- создание, удаление и управление пользователем своими объектами (файлами, расчетами);
- хранение информации о доступных вычислительных ресурсах;
- отслеживание состояния расчетных программ на удаленных вычислительных ресурсах;
- передача файлов и команд на удаленные вычислительные ресурсы;
- предоставление пользователю удобного web-интерфейса.

Система УД и УРВР построена по клиент-серверной архитектуре. Логически систему можно разбить на 2 взаимодействующие части [3]:

- Система управления распределенными вычислительными ресурсами (СУ РВР). Она отслеживает состояние кластеров, загружает на них новые задания и сохраняет результаты. В эту подсистему входят менеджер вычислительных ресурсов, БД и кластерные агенты.
- Система удаленного доступа к распределенным вычислительным ресурсам (СУД РВР). Ее задача – предоставить пользователю интерфейс для размещения новых заданий в БД и получения результатов. Данная подсистема реализована с помощью web-доступа через специальный сервер приложений и включает в себя также интерфейс к системе отладки и «виртуальной лаборатории».

Для работы с системой используется web-браузер. Хранение пользовательской и служебной информации осуществляется на сервере БД. Пользователь взаимодействует с базой при помощи сервера приложений, который является промежуточным звеном в СУ РВР, выполняющим обращение к серверу БД и обработку данных перед их представлением пользователю. Сервер БД также участвует в обработке данных посредством хранимых процедур и функций.

Система с данной архитектурой позволяет многим пользователям одновременно работать с высокопроизводительными ресурсами в удаленном режиме, предъявляя при этом минимальные требования к клиентскому аппаратному и программному обеспечению (ПО). Применение web-интерфейса снимает с пользователя необходимость обладания специальными знаниями (команды операционной системы и др.) Кластерные агенты позволяют унифицировать работу сервера с вычислительными ресурсами, имеющими различную платформу.

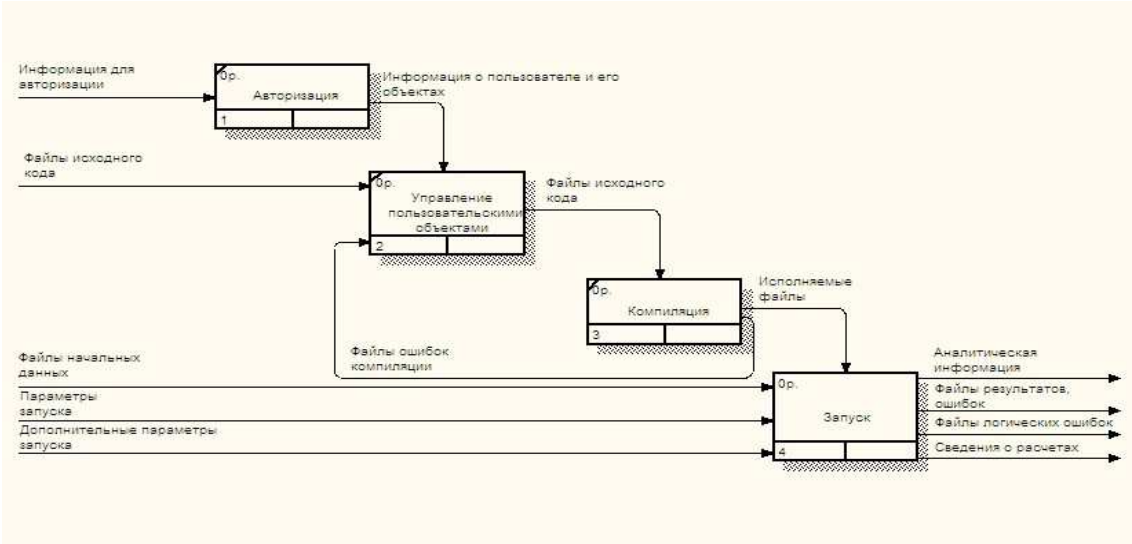

Рис. 1 Функциональная модель программного комплекса

Рисунок 1 демонстрирует общую последовательность обработки данных программным комплексом. Пройдя авторизацию на ИВП, пользователь получает доступ к своим файлам и проектам, куда может загрузить исходные коды новой программы, забрать файлы результатов и ошибок проведенных расчетов. В специальной web-форме он создает задание на компиляцию, где указывает интересующий его компилятор и файлы исходного кода. После того, как задание закончено, и в проекте появился исполняемый файл, можно сформировать задание на запуск, где указываются необходимые вычислительные ресурсы (конкретный кластер, количество узлов, время счета) и параметры запуска (аргументы программы, используется ли система отладки, количество необходимых запусков, пределы варьирования параметров). На выходе пользователь получает файлы результатов расчетов и системных ошибок (потоки stdout и stderr), найденные системой отладки логические ошибки и аналитическую информацию в виде графиков и диаграмм, предоставляемую «Виртуальной лабораторией».

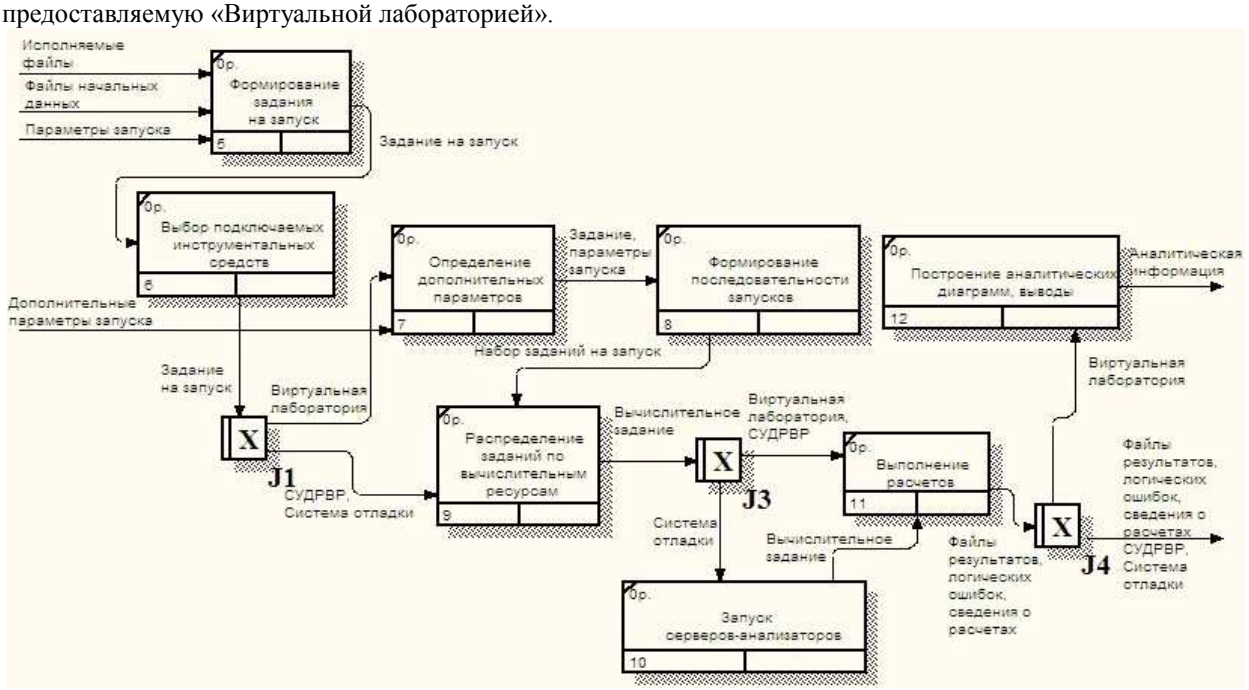

Рис. 2 Запуск расчета в системе УД и УРВР

На рисунке 2 представлена декомпозиция блока запуска. Данный процесс получает на входе пользовательские файлы и параметры запуска, такие как вычислительная архитектура и используемая операционная система (ОС). Дополнительные параметры представляют собой совокупность входных данных, определяемых пользователем при работе с «Виртуальной лабораторией». Механизм обработки задания имеет особенности, в зависимости от выбранных пользователем инструментальных средств. Выход процесса составляют файлы результатов вычислений, сведения о расчетах, файлы логических ошибок, формируемых системой отладки, и аналитическую информацию как результат использования «Виртуальной лаборатории».

ВИРТУАЛЬНАЯ ЛАБОРАТОРИЯ

Основное направление использования данного компонента ИВП - исследование эффективности параллельных программ и получение сведений для их дальнейшей оптимизации. Через web-интерфейс пользователь загружает свою параллельную программу в систему, задает набор параметров исследования, диапазон их значений и отправляет задачу на запуск. Система самостоятельно формирует последовательность запусков исследуемой программы с различными комбинациями параметров, выполняет расчеты и производит анализ зависимости времени и результатов выполнения программы от параметров. Следует отметить, что один численный алгоритм может быть скомпилирован различными компиляторами для выполнения на кластерах с разными ОС и вычислительными архитектурами. Таким образом, система позволяет сравнивать эффективность выполнения программ в различных вычислительных средах. Результаты расчетов представляются в виде таблицы, с указанием всех использованных при запуске параметров. Блок анализа результатов вычислений позволяет получить значения эффективности и ускорения по выбранным пользователем расчетам, а также построить диаграммы, отражающие зависимость времени выполнения от набора использованных при запуске параметров. Результатом анализа является вывод о степени влияния каждого параметра параллельной программы на ее эффективность [5].

Другое направление использования данной системы – учебный процесс. «Виртуальной лабораторией» могут пользоваться студенты и преподаватели в рамках обучения курсам, связанным с высокопроизводительными вычислениями. Система позволяет организовать проведение виртуального лабораторного практикума студентов. Задача обучающихся состоит в том, чтобы протестировать

параллельную программу (написанную самостоятельно или предоставленную преподавателем) с использованием различного ПО, различных вычислительных ресурсов и входных параметров задания (может варьироваться число вычислительных узлов кластера, размерность матрицы и пр.) По полученным экспериментальным данным студент должен построить графики ускорения и эффективности и сделать выводы [5].

СИСТЕМА АВТОМАТИЧЕСКОГО КОНТРОЛЯ КОРРЕКТНОСТИ MPI-ПРОГРАММ

В полном цикле разработки даже последовательной компьютерной программы этап отладки является одним из самых затратных по времени процессом. Отладка параллельных же программ является еще гораздо более сложным процессом, поскольку необходимо анализировать поведение множества одновременно протекающих процессов. Кроме того, добавляется широкий класс ошибок, являющихся следствием взаимодействия данных ветвей параллельного приложения и применения технологий параллельного программирования, самой популярной из которых является MPI (Message Passing Interface) [1].

Все множество специфических ошибок, свойственных MPI-программам, можно разделить на 2 группы:

- **Локальные**, возникающие в пределах одного MPI-процесса.
- **Глобальные**, следующие из взаимодействия нескольких MPI-процессов.

Подробная классификация ошибок в параллельных программах приведена в [2].

При построении инструментальных средств обнаружения семантических ошибок в параллельных приложениях применяются несколько подходов [4]. Средства, использующие подход автоматического контроля корректности выделяются среди других тем, что не требуют от пользователя практически никаких действий, могут обнаруживать потенциальные ошибки и способны работать в offline-режиме (диалог с пользователем не требуется). Основная идея данного метода заключается в том, что на этапе компиляции программы к ней прилинковывается профилировочная библиотека. Эта библиотека перехватывает вызовы MPI-функций и производит некоторую аналитическую обработку параметров вызова с целью обнаружения логических ошибок. Затем вызывается функция MPI-реализации, которая производит необходимые пользователю действия, после этого опять управление передается библиотеке, которая снова может выполнить какую-либо работу по анализу MPI-функции и вернуть управление обратно в пользовательскую программу [4].

В настоящее время существуют несколько инструментальных средств автоматического контроля корректности MPI-программ [2], но они обладают существенными архитектурными недостатками, ввиду которых:

1. Не способны к масштабированию на большое количество вычислительных узлов кластера.
2. Несут значительные накладные расходы.

Разрабатываемая распределенная система отладки также использует данный подход и, благодаря оригинальной архитектуре, решает проблемы существующих программных средств этого класса.

На рис. 3 изображена компонентная модель создаваемой программной системы. Для многих коммуникационных функций стандарта MPI в системе существует ряд проверок на появление глобальных и локальных ошибок. Локальные ошибки в пределах каждого MPI-процесса отыскиваются служебным потоком. Обнаружением глобальных ошибок занимаются серверы-анализаторы, которым MPI-процессы передают параметры вызываемых функций для обработки. Масштабируемость системы достигается благодаря тому, что таких серверов-анализаторов может быть множество и работа по обнаружению глобальных ошибок распределяется по всем серверам.

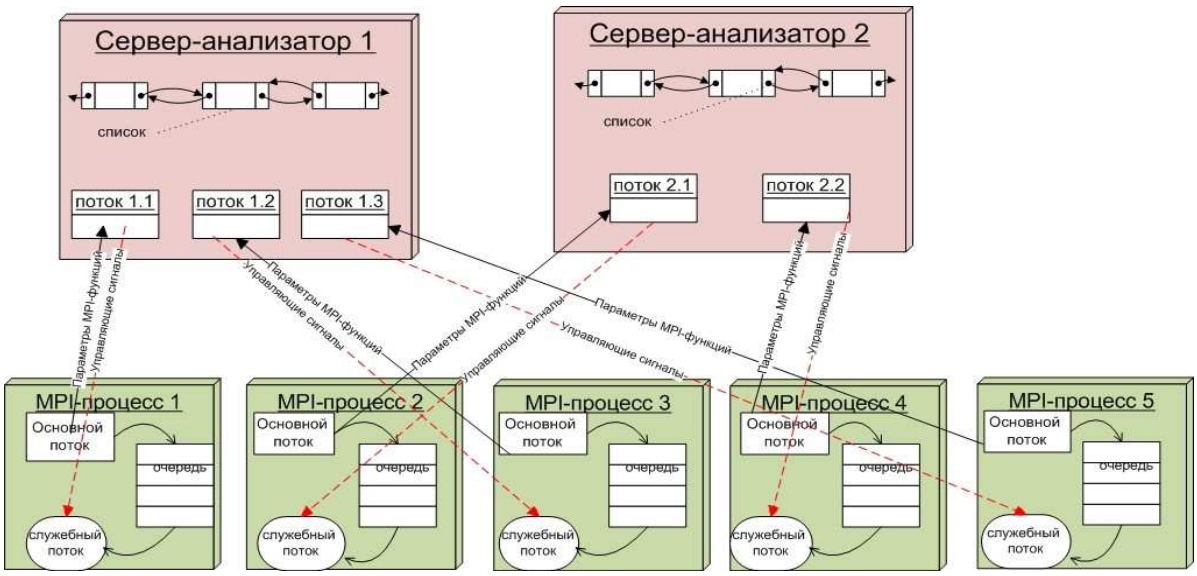

Рис. 3 Компонентная модель системы отладки

Ниже приведен алгоритм работы системы. Файлы исходного кода проходят через консольную программу-препроцессор. Вначале каждой строки каждого файла препроцессор вставляет операторы, заполняющие определенные глобальные переменные именем файла и номером текущей строки кода. Данный препроцессор введен в систему по той причине, что профилировочный интерфейс стандарта MPI не предоставляет информацию о номере строки, в которой вызвана MPI-функция. В то же время сообщения об ошибках, которые выдает система, должны содержать помимо имен функций также и номера соответствующих строк кода.

Далее, инструментированный код подвергается компиляции, в ходе которой к программе прилинковывается профилировочная библиотека. Затем утилита **runner** на одной группе узлов кластера запускает созданный исполняемый файл посредством mpiexec, а на другой – процессы серверов-анализаторов **analyze.exe**. Утилита оперирует несколькими параметрами, которые для удобства сведены в один конфигурационный файл.

В пределах MPI-процессов порождаются дополнительные служебные потоки, занимающиеся обнаружением локальных ошибок. Для обеспечения взаимодействия между асинхронными основным потоком MPI-процесса и служебным потоком в систему введена очередь заданий (рис. 3). В пределах же analyze.exe для каждого процесса из обслуживаемых данным сервером создается по одному потоку.

При вызове в параллельной программе многих коммуникационных операций их параметры передаются одной строкой на соответствующий сервер. Таких посылок MPI-процесс отправляет две: перед фактическим вызовом операции (PMPI-функции) и после неё. «Закрепленный» за данным процессом поток сервера производит парсинг принятой строки и заполняет элемент основной структуры данных сервера – списка сообщений. Затем на сервере производится ряд проверок, в ходе которых просматривается список, и выявляются глобальные ошибки. Если функция отработала корректно и посылка была успешно передана от отправителя получателю в случае операции точка-точка или все участвующие в групповой операции процессы отправили/получили свои порции данных, то соответствующие элементы списка удаляются.

В некоторых случаях возникает необходимость передать какую-либо управляющую информацию от сервера MPI-процессам. Такие управляющие сигналы принимают служебные потоки на вычислительных узлах. При выявлении семантических ошибок, как серверы, так и служебные потоки вносят записи во внутренние структуры данных в компактном виде для экономии оперативной памяти. После того, как все ветви параллельной программы завершили свою работу, данные из всех таких структур перемещаются на один сервер-анализатор, который формирует единую структуру и выгружает из нее всю информацию в файл журнала ошибок. Далее журнал передается на головной узел кластера.

В настоящее время разработаны алгоритмы для обнаружения следующих типов ошибок.

Локальные:

- процесс не вызвал одну или обе функции MPI_Init и MPI_Finalize;
- вызванная функция не была завершена (процесс завис на данной функции);
- указанный тип передаваемых данных не определен;
- вторичное использование неосвобожденного объекта типа MPI_Request;
- гонки данных, обусловленные затиранием буфера, выделенного для второй буферизованной отправки, когда еще не успела завершиться первая.

Глобальные:

- дедлоки, обусловленные операциями встречной отправки или приема;

- гонки данных, возникающие в результате того, что одной операции приема удовлетворяют несколько операций отправки;
- несоответствие типов данных в операциях отправки и приема;
- изменение данных, участвующих в коммуникационной операции, основной ветвью MPI-процесса во время неблокирующей передачи/приема;
- отсутствие вызова парной функции к операции типа точка-точка;
- несоответствие размеров сообщений в операциях передачи/приема;
- некоторая подгруппа процессов, входящих в коммуникатор, вызвала коллективную операцию, а среди остальных какой-либо процесс вызвал другую коммуникацию.

Благодаря своей архитектуре, система отладки обладает низкими накладными расходами. Действительно, потоки, выполняющие программу пользователя; служебные потоки, отыскивающие локальные ошибки и серверы-анализаторы работают асинхронно, не ожидая друг друга. Что касается накладных расходов на сеть, то объем данных, циркулирующих между серверами и MPI-процессами, крайне мал – это только параметры функций и управляющие сигналы.

Система отладки может использоваться как самостоятельное программное средство, так и в интеграции с ИВП. На веб-формах добавлены элементы, позволяющие передать соответствующие настройки компилятору и утилите runner. Эти параметры, соответственно, помещаются в базу данных. После окончания работы программы вместе с файлами потоков вывода (stdout) и системных ошибок (stderr), пользователю выдается еще и журнал найденных логических ошибок.

ЛИТЕРАТУРА:

1. Афанасьев К.Е. Многопроцессорные вычислительные системы и параллельное программирование / К.Е. Афанасьев, С.В. Стуколов. – Кемерово: Кузбассвузиздат, 2003. – 233 с.
2. Власенко, А. Ю. Модель масштабируемой системы автоматического контроля корректности параллельных программ. [Текст] / А. Ю. Власенко // Вестник НГУ. Т.7, выпуск 4, 2009. С.53-66.
3. Григорьева, И.В. Система удаленного доступа и управления распределенными вычислительными ресурсами [Текст]/ И.В. Григорьева, А.В. Демидов // Вычислительные технологии. - 2008. - том 13, специальный выпуск 5 - С. 28-32.
4. Ефимкин, К. Н. Средства отладки MPI-программ. Анализатор корректности [Электронный ресурс]/ К. Н. Ефимкин, О. Ф. Жукова, В.А.Крюков// http://www.keldysh.ru/papers/2006/prep28/prep2006_28.html. - 2006.
5. Окулов, Н.Н. Компонент "Виртуальная лаборатория" системы удаленного доступа к распределенным вычислительным ресурсам / Н.Н. Окулов // Вестник томского государственного университета. Управление, вычислительная техника и информатика, №2(7), 2009, стр. 95-100.