УДК 004.414.23, 519.876.5

ПАКЕТЫ МОДЕЛИРОВАНИЯ DATAGRID

Кореньков Владимир Васильевич[1], Нечаевский Андрей Васильевич[2]

[1] Кандидат физико-математических наук, профессор Института системного анализа и управления;

ГОУ ВПО «Международный Университет природы, общества и человека «Дубна», Институт системного анализа и управления;

141980, Московская обл., г. Дубна, ул. Университетская, 19;

Объединенный институт ядерных исследований, Лаборатория информационных технологий;

141980, Московская обл., г. Дубна, ул. Жолио-Кюри, 6;

e-mail: korenkov@lxpub01.jinr.ru.

[2] Магистр техники и технологии по направлению «Системный анализ и управление», инженер-программист;

ГОУ ВПО «Международный Университет природы, общества и человека «Дубна», Институт системного анализа и управления;

141980, Московская обл., г. Дубна, ул. Университетская, 19;

Объединенный институт ядерных исследований, Лаборатория информационных технологий;

141980, Московская обл., г. Дубна, ул. Жолио-Кюри, 6;

e-mail: nechav@mail.ru.

Впервые представлено краткое описание систем моделирования Bricks, OptorSim и GridSim. Описываются особенности этих пакетов, их возможности для моделирования DataGrid: представление основных элементов DataGrid, создание топологии сети, осуществление передачи данных, предлагаемая статистика, отображение различного рода ошибок и другие возможности. Рассматриваются достоинства и недостатки систем при создании моделей.При использовании реальных Grid не исключено возникновение различного рода сбоев и ошибок, возможность моделирования которых также рассматривается в работе.

Ключевые слова: моделирование, DataGrid, алгоритм, передача данных.

DATAGRID SIMULATION PACKAGES

Korenkov Vladimir Vasilevich [1], Nechaevskiy Andrey Vasilevich[2],

[1] PhD, professor of Institute of system analysis and management;

International university of the nature, society and man «Dubna», Institute of system analysis and management;

141980, Dubna, Moscow reg., Universitetskaya str., 19;

Join institute for nuclear research, Laboratory of information technologies;

141980, Moscow reg., Dubna, Joliot-Curie, 6;

e-mail: korenkov@lxpub01.jinr.ru.

[2] Master of engineering and technology «Systems analysis and management», engineer-programmer;

International university of the nature, society and man «Dubna», Institute of system analysis and management;

141980, Dubna, Moscow reg., Universitetskaya str., 19;

Join institute for nuclear research, Laboratory of information technologies;

141980, Moscow reg., Dubna, Joliot-Curie, 6;

e-mail: nechav@mail.ru.

The simulation packages Bricks, OptorSim and GridSim are represented, for example, simulation of the DataGrid basic elements, the data transfer possibilities, offered statistics, failures description and other possibilities. The qualities and shortcomings of the systems are shown.The Grid usage does not except a various failures. Failures simulation is also described.

Keywords: simulation, DataGrid, algorithm, data transfer.

Введение

В настоящее время объемы информации, получаемые в физике высоких энергий, в астрономии, в био-информатике и других областях, достигают таких размеров, что становится проблематичным размещать их в локальных вычислительных комплексах. Идея Grid-технологии состоит в том, чтобы объединить гетерогенные и географически распределенные ресурсы для решения качественно новых задач, в частности для размещения, хранения и управления большими массивами данных.

Сейчас проектируется все больше информационных систем сложной конфигурации. Это параллельные архитектуры с тысячами процессоров, десятками тысяч объединенных сетями компьютеров, параллельные базы данных с возможностью обработки до миллиона операций в секунду. В некоторых случаях системы разрастаются значительно больше, чем их проектировали. Затраты на проектирование и развитие таких систем могут быть значительно уменьшены в случае, если использовать эффективные методы моделирования таких систем.

Самым масштабным научным проектом последних лет является создание в ЦЕРНе (Европейская организация для ядерных исследований, ЦЕРН, Женева, Швейцария) большого адронного коллайдера LHC (Large Hadron Collider) [13]. Ожидается, что после запуска в нормальном режиме ускоритель будет производить около 15 петабайт данных ежегодно (15 миллионов гигабайт). Открытие новых фундаментальных частиц и выявление их свойств на ускорителе LHC невозможно без статистического анализа больших объемов данных, собираемых с детекторов LHC, и детального сравнения с данными теоретического моделирования. Для решения этой грандиозной задачи в рамках проекта LCG (LHC Computing GRID) [25] разрабатывается новейшая распределенная модель хранения и обработки данных, основное место в которой занимают Grid-технологии. Задача проекта LCG — создать инфраструктуру для хранения и анализа данных, которой будут пользоваться все сообщество ученых, участвующих в LHC.

Суть распределенной модели состоит в том, что весь объем информации с детекторов LHC после обработки в реальном времени и первичной реконструкции на вычислительных мощностях ЦЕРН должен направляться для дальнейшей обработки и анализа в региональные центры. Иерархический принцип организации информационно-вычислительной системы LCG, предполагает создание центров разных уровней (Tier's):

Tier0(ЦЕРН) -> Tier1 -> Tier2 -> Tier3 -> компьютеры пользователей.

Уровни различаются как по масштабу вычислительных и архивных ресурсов, так и по выполняемым функциям [2]:

Таблица 1. Уровни иерархической модели LCG и их функции

Tier0 (ЦЕРН)	первичная реконструкция событий, калибровка, хранение копий полных баз данных
Tier1	полная реконструкция событий, хранение актуальных баз данных по событиям, создание и хранение наборов анализируемых событий, моделирование, анализ
Tier2	репликация и хранение наборов анализируемых событий, моделирование, анализ
Tier3	кластеры отдельных исследовательских групп

Конечно, создание такой распределенной системы имеет свои трудности. Прежде всего, необходимо обеспечить достаточную пропускную способность, реализовать поддержку различного оборудования, проблемы сохранности данных (устойчивость к повреждениям и удалениям) на протяжении всего жизненного цикла проекта LHC, обеспечить распределение ресурсов между различными груп-

пами пользователей и т.д. Для решения этих задач на этапе проектирования могут использоваться пробные системы. Возможности тестовых Grid, как правило, ограничены, а создание испытательного Grid с размером близким к реальным является дорогим и трудоёмким. Так же невозможно проверить множество различных сценариев использования DataGrid на реальных испытательных стендах. Под сценарием использования DataGrid понимается набор правил для пользователей и алгоритмы работы ресурсов. Поэтому возникает задача моделирования Grid для изучения сложных сценариев [18].

В процессе использования уже существующих Grid, возникают вопросы, связанные с внесением изменений в Grid и тем как эти изменения отразятся на работоспособности Grid и ее отдельных сегментов. За длительное время мониторинга передач данных из ЦЕРНа на Tier1-сайты, выяснилось, что одна из основных проблем связана с ошибками, возникающими из-за ограничения времени передачи данных — таймаутами. Такие ограничения необходимы для контроля состояния элементов системы и как механизм ограничения очередей. Возникает вопрос, как изменится ситуация при увеличении таймаутов, при увеличении размеров передаваемых файлов и что надо сделать для увеличения пропускной способности канала. Эти задачи можно решить с помощью симуляторов.

В настоящее время существует несколько пакетов моделирования Grid систем: Brick [4], SimGrid [15], OptorSim [17] и GridSim [12]. В большинстве своем эти программы моделируют вычислительные Grid и дают возможность получить статистические данные о наиболее важных характеристиках моделируемой системы: использование дисковых ресурсов, пропускной способности каналов, вероятности потерь данных и т.п. Системы имеют разные особенности и возможности для моделирования Grid. Возможность моделировать DataGrid реализована в Brick, OptorSim и GridSim.

Кроме симуляторов Grid существуют системы эмулирующие Grid. Эмуляция - воспроизведение аппаратными или программными средствами либо их комбинацией работы других устройств или программ [1]. Grid eXplorer [11] и MicroGrid [14] — эмуляторы Grid. Эмуляторы работают в реальном времени и позволяют тестировать Grid-приложения и алгоритмы, не имея реальной Grid-системы. Создание и настройка больших испытательных стендов в таких системах требует много времени и является затруднительной. Поэтому эмуляторы не приняты во внимание в наших исследованиях систем моделирования Grid.

Системы моделирования DataGrid могут быть использованы для решения трех основных типов задач:

- выявление «слабых мест» системы (перегруженные каналы, места возникновения больших очередей и прочие «узкие места»);
- тестирование различных сценариев использования DataGrid;
- поиск оптимальной конфигурации ресурсов.

Моделирование позволяет проводить разнообразные эксперименты с исследуемым объектом, не прибегая к физической реализации, что позволяет предсказать и предотвратить большое число неожиданных ситуаций в процессе эксплуатации, которые могли бы привести к неоправданным затратам, потере информации, а, возможно, и к повреждению дорогостоящего оборудования. В процессе моделирования можно определить минимально необходимое оборудование, обеспечивающее потребности передачи, обработки и хранения информации, оценить необходимый запас производительности оборудования, обеспечивающего возможное увеличение производственных потребностей, выбрать несколько вариантов оборудования с учетом текущих потребностей и перспективы развития в будущем, провести проверку работы системы, выявить ее «узкие» места и т.д.

1. Пакеты моделирования DataGrid

1.1. Bricks

Система моделирования Bricks [4] предназначена для моделирования клиент-серверной архитектуры как глобальной вычислительной системы. В ней предполагается централизованное глобальное планирование.

Bricks первый Grid симулятор, предназначенный для исследования алгоритмов планировщиков заданий и пропускной способности каналов. Создатели системы руководствовались идеей, что Grid состоит из пользователей, запускающих по сети задания на общие ресурсы [20]. Симулятор был

реализован в Java 1.1.8. Используя немногочисленные доступные публикации, мы оценили функциональные возможности Bricks.

Архитектура

Bricks реализована как взаимодействие двух главных компонент — модуля моделирования вычислительной среды Grid и модуля планирования [19].

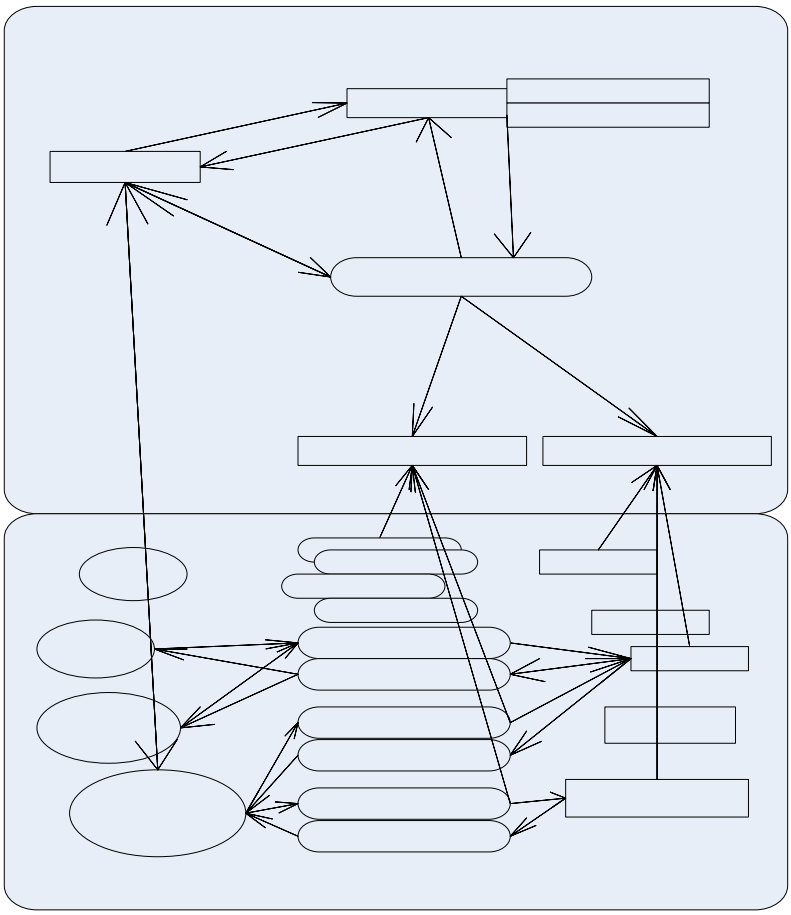

Рис.1. Архитектура Bricks

Предполагается, что вычислительная среда — сеть, состоит из однородных элементов (Host). На рис. 2 показан пример возможной сетевой архитектуры.

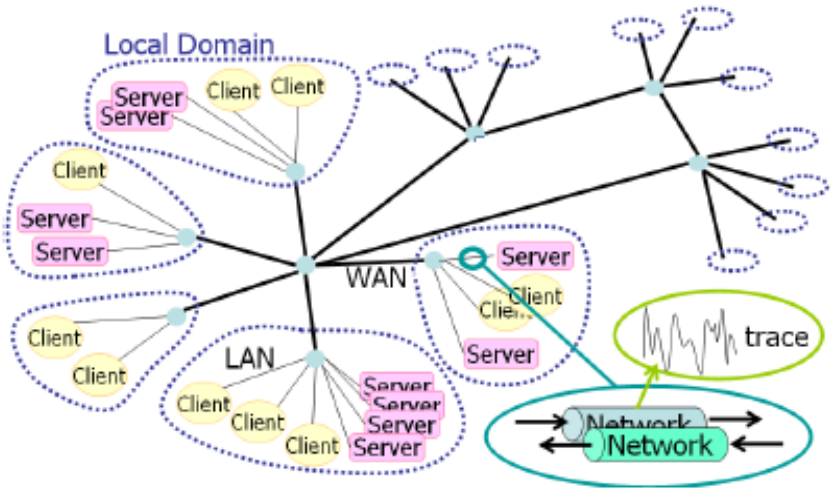

Рис.2. Пример топологии сети

Host включает: Клиента (Client), Сервер (Server) и объект типа Диск (Disk). Объект Client представляет пользовательскую машину, которая генерирует задачи для выполнения на ресурсе. Объект Server характеризуется производительностью, рабочей нагрузкой и их распределением в течение времени. Объект Network объединяет клиентов и сервера и характеризуется пропусканной способностью, нагрузкой и фоновым трафиком [21]. Различные сценарии могут быть применены к объектам. Задания характеризуются требуемым числом операций. Задание по умолчанию выполняется на одном ресурсе.

Сервер и сетевые ресурсы работают с очередями заданий. Сервер обрабатывает запланированные задачи способом FCFS (First-come, First-served). Два способа обработки очередей реализованы в Bricks: QueueFCFS и QueueTimeSharing.

Топология сети должна быть определена пользователем в конфигурационном файле.

Bricks включает в себя средства мониторинга, прогнозирования и планирования работы системы [21]. Средствами мониторинга проводится контроль работоспособности ресурсов и сохранение этих результатов в базе данных (ResourceDB). В частности, NetworkMonitor фиксирует пропускную способность и время ожидания, ServerMonitor — производительность, загруженность и работоспособность серверов. Модуль прогнозирования определяет использование и работоспособность сетевых ресурсов и серверов. Планировщик размещает новую задачу на подходящий сервер, принимая во внимание информацию о состоянии ресурса, полученную из ResourceDB и модуля прогнозирования.

Функционирование системы

Перед запуском модели пользователь должен определить частоту поступления задач от клиента и объемы задач. Работа системы начинается с анализа конфигурационных файлов.

Когда новая задача поступает от Клиента, она обрабатывается Планировщиком, ответственным за размещение задачи с учетом ее особенностей. Планировщик запрашивает информацию в модуле Прогнозирования о возможности использования того или иного ресурса и размещает задачу на подходящий Сервер. Сервер обрабатывает задания по очереди. Когда Сервер завершил обработку задачи, результаты пересылается Клиенту.

Статистика эксперимента

Вся информация о состоянии ресурсов и процессе моделирования доступна. Она включает: полосу пропускания сети и время ожидания, пропускную способность сервера и размеры очередей, текущее использование центрального процессора и среднюю загрузку, размер заданий, размер их ответа, время, в течении которого задание было начато и выполнено и многое другое.

Копирование данных

Структура Bricks позволяет моделировать множественное копирование данных (репликации) [22]. Репликация данных и автоматическое распределение задач по ресурсам гарантирует баланс нагрузки и облегчает доступ к данным всем пользователям. Охват всех часовых поясов также облегчает круглосуточный мониторинг и поддержку.

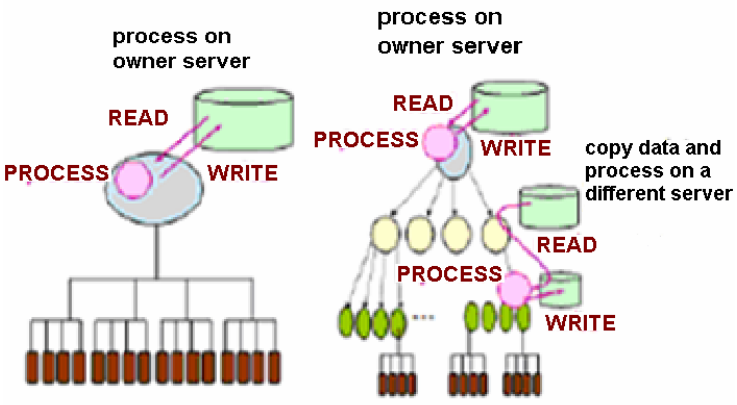

Рис.3. Централизованная и иерархическая модель обработки заданий

Копирование данных может осуществляться согласно схемам, представленным на рис. 3. В централизованной модели все данные, и все задания, имеющие дело с этими данными, хранятся и обрабатываются на единственном сайте, у которого, как предполагается, есть достаточные ресурсы для обработки и соответствующая емкость запоминающих устройств. Иерархическая модель предполагает обработку заданий, когда при увеличении загрузки сервера, часть данных копируется на ресурсы нижнего уровня.

ReplicaManager выполняет алгоритм копирования файлов и сбора информации о дисковых ресурсах.

Чтобы избежать нехватки дискового пространства, Bricks следит за свободным пространством на дисковых ресурсах и удаляет данные, когда это необходимо [22]. Критерием удаления данных является отношение использованного дискового пространства ко всему объему. Этот порог определяет пользователь в конфигурационном файле. Когда пороговое значение превышено, происходит удаление файлов.

1.2. OptorSim

OptorSim создавался в рамках европейского проекта European DataGrid (EDG) [23], как инструмент для моделирования DataGrid. В приложениях, работающих с большими объемами данных, важно не только избежать потери данных, но и организовать оптимальный доступ к ним. OptorSim — пакет моделирования DataGrid, реализованный на языке Java, позволяет оценивать различные алгоритмы оптимизации и стратегии копирования. Исходный код широко доступен.

В OptorSim, каждый сайт может содержать несколько элементов хранения (StorageElement (SE)) и/или вычислительных элементов (ComputingElement (CE)). В пакете несколько конфигурационных файлов используются для задания параметров модели. Подробное описание файлов и характеристик представлено в руководстве пользователя [9]. Их создание и доработка модели не вызывает особых трудностей. Существует возможность моделирования фонового трафика. Открытый доступ к исходному коду OptorSim дает возможность доработки системы самостоятельно.

Архитектура

Система состоит из двух частей: первая предназначена для моделирования Grid-ресурсов, а вторая часть — для моделирования запуска заданий пользователей. Концептуальная модель архитектуры OptorSim показана на рис. 4.

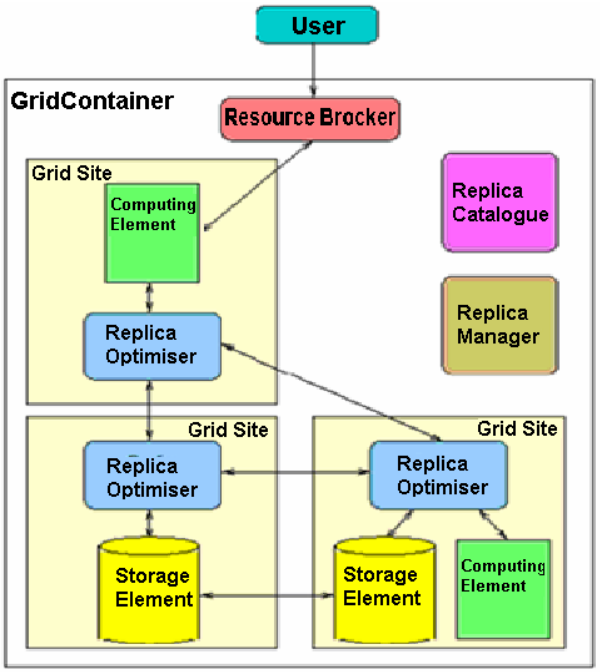

Рис.4. Архитектура OptorSim

Сеть состоит из Сайтов (GridSites), связанных между собой. Топология сети определяется пользователем в конфигурационном файле. Основной идеей является то, что каждая задача состоит из множества файлов данных (набор данных связанных с этой задачей). Прежде чем копировать файл на ресурс определяется «наилучшее» местоположение копии файла, что позволяет сайтам копировать файл с разных источников во избежание чрезмерной загруженности ресурсов. Предполагается, что для выполнения задачи нет необходимости в вычислительных ресурсах.

StorageElements позволяет выполнить различные манипуляции с файлами: добавление и получение доступа к файлу, удаление файла. В случае, когда необходимо удаление файлов, возможно использование различных сценариев удаления файлов. Например, можно удалять наименее используемые файлы или самые старые файлы.

ReplicaCatalogue (RC) — список соответствия логических имен файлов их физическим именам. ReplicaManager (RM) руководит копированием данных, управлением файлами и регистрирует их в ReplicaCatalogue. Пересылка данных между сайтами осуществляется RM и контролируется Replica-Optimiser (RO). RO содержит алгоритмы копирования данных, которые обеспечивают обмен данных, создание и удаление копий. Цель моделирования — добиться оптимизации работы сети за счет объединения усилий по оптимизации отдельных RO на сайтах. ResourceBroker (RB) решает какой Сайт предоставить для выполнения задачи. В OptorSim реализовано несколько алгоритмов работы брокера:

- *RandomCEResourceBroker* посылает GridJob случайному CE;
- *AccessCostResourceBroker* выбирает CE, которому понадобится наименьшее время для получения всех требуемых в задаче файлов;
- *QueueLengthResourceBrokers* выбирает CE с наименьшей очередью;
- *CombinedCostResourceBroker*: для выполнения задачи подбирает CE, которому понадобится наименьшее время для получения всех требуемых файлов во всех задачах, находящихся в очереди на этом CE.

Моделирование проекта GridPP [6] и испытательного стенда CMS DataChallange [7] показали, что при использовании алгоритма RandomCEResourceBroker время выполнения задачи наибольшее, алгоритм QueueLengthResourceBroker выполняет те же задачи примерно в два раза быстрее. Access-CostResourceBroker требует меньше времени, но характеризуется низким использованием CE, поскольку выбираются CE с большим количеством связей. К тому же, сценарии RandomCEResource-Broker и QueueLengthResourceBroker быстро приводят к заполнению дискового пространства SE (до 90 %), в то время как AccessCostResourceBroker использует SE приблизительно на 37%. Наилучшее использование CE, хорошие показатели использования SE (приблизительно 75 %) и наименьшее время выполнения задач показал алгоритм CombinedCostResourceBroker, так как оценивает и коммуникативные возможности сайта, и длину очередей задач, что позволяет достичь наилучшего баланса между планированием заданий на ближайшие к данным сайта и загруженностью сайтов. Стоит отметить, что выбор соответствующего алгоритма работы брокера указывается в конфигурационном файле.

Поведение Пользователя (User) определяется в конфигурационном файле и различается распределением или частотой, с которой они запускают задачи на RB. Так, по сценарию SimpleUsers, пользователи запускают задачи согласно равномерному распределению (время простоя между запусками определяется в конфигурационном файле). RandomWaitUsers моделирует ситуацию, когда пользователи «отдыхают» между запусками случайный промежуток времени. Класс CMSDC04Users использует гауссовское распределение.

Файл данных — объект, содержащий информацию об имени файла, размере, является ли файл копией или мастер-файлом. Мастер-файл — оригинальная копия данных, которая не может быть удалена. Набор данных (Dataset) — несколько файлов, которые обладают общими свойствами. Каждая задача может содержать один или более наборов данных. Параметры доступа (access patterns) определяют порядок обработки файлов в задаче.

Функционирование системы

На рис. 5 показан алгоритм работы ResourceBroker. Пользователь запускает новые задачи, которые RB обрабатывает по очереди. Задача должна быть отправлена для выполнения на сайт, у которо-

го есть доступ ко всем файлам в задаче. Когда RB находит невыполненную работу, то выполняется поиск сайта с подходящим CE. Если найденный CE свободен, задача будет направлена на сайт и добавлена в очередь на обработку на CE (рис. 5).

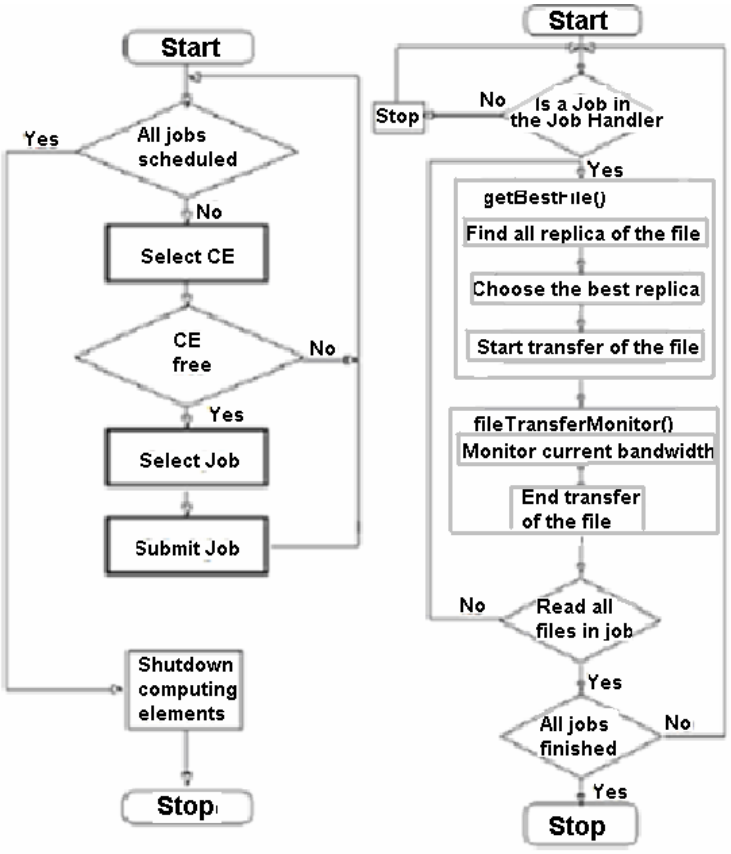

Рис.5. Алгоритмы работы RB и CE

Далее CE моделирует запуск задачи (рис. 5). Для каждого файла в задаче определяется «наилучшее» расположение файла для копирования, согласно алгоритму оптимизации. Как только этот «лучший» файл был найден, FileTransfer моделирует передачу данных по сети. Время выполнения моделируется как:

$$\frac{job.getLatency() + job.getLinearFactor() \cdot size(allfiles)}{(numberofWNs) \cdot WNcapacity},$$

где латентность (Latency) выражается в секундах, а *LinearFactor* показывает количество секунд на обработку мегабайта данных. Когда все файлы получены, задача считается выполненной. На рис. 6 представлена диаграмма последовательности действий при выполнении задачи.

Файлы копируются один за другим. Если несколько вычислительных элементов копируют файлы по одной сети, то пропускная способность делиться между CE. Таким образом, каждый последующий файл должен ждать, пока предыдущий файл будет передан.

Статистика эксперимента

Статистика в OptorSim представлена как по каждому элементу в отдельности, так и по работе модели. Отчетная информация доступна в виде таблиц, графиков и диаграмм. Для SE можно получить информацию об объеме дисков и их загрузке. Есть возможность получить статистику по времени выполнения задач, а также процентное соотношение времени использования CE ко времени простоя, число удаленных и локальных запросов к файлу, а также количество переданных файлов. Интересен показатель эффективности использования сети. Эффективность использования сети рассчитывается как отношение количества переданных файлов (удаленный доступ или репликация файлов) к числу запрашиваемых файлов. Этот показатель помогает решить вопрос — следует ли

менять стратегию оптимизации и применить алгоритм, который помещает файлы на “правильные” Сайты, в случае низкой эффективности?

Интерфейс

Одним из достоинств пакета является наличие графического интерфейса для визуального отображения модели (рис. 6). Графический интерфейс позволяет пользователю отслеживать состояние модели, даже во время выполнения моделирования. Практический интерес представляет закладка Statistics, где текущая информация представлена в табличной форме. Закладка Logical View показывает граф, на котором отображаются текущие передачи файлов между сайтами. Подробная информация представлена в руководстве пользователя [9].

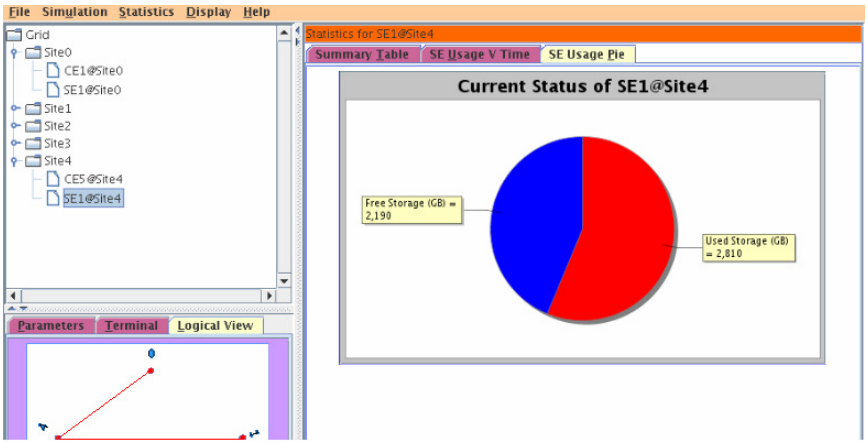

Рис.6. Графический интерфейс OptorSim

1.3. GridSim

GridSim позволяет моделировать различные классы гетерогенных ресурсов, пользователей, приложений, брокеров ресурсов, и планировщиков. Моделирование DataGrid — одна из возможных областей применения GridSim. Она предоставляет возможность определять ресурсы с гетерогенными компонентами хранения. Система обладает гибкостью для осуществления различных стратегий управления данными [18]. На сайте разработчиков много примеров по работе с программой [12]. На начальном этапе создания модели может использоватся VisualModeler (автоматическое создание пользователей и ресурсов, генерация кода и т.п.). Однако завершающий этап — создание связей, необходимо выполнять вручную, что довольно трудоемко и может быть причиной возникновения ошибок. Стоит отметить возможность моделирования фонового трафика и моделирование отказов ресурсов [8]. GridSim успешно использовался для моделирования брокера ресурсов Grid-Nimrod-G [5].

Архитектура

GridSim имеет многоуровневую архитектуру, что позволяет легко добавлять новые компоненты (рис. 7). Нижний уровень — Java Virtual Machine (JVM). Виртуальная машина Java — основная часть исполняющей системы Java, так называемой Java Runtime Environment (JRE). Виртуальная машина Java интерпретирует и исполняет байт-код Java, предварительно созданный из исходного текста Java-программы компилятором Java.

Второй уровень представляет собой SimJava [16] библиотеку, которая используется для управления моделированием и взаимодействием объектов GridSim.

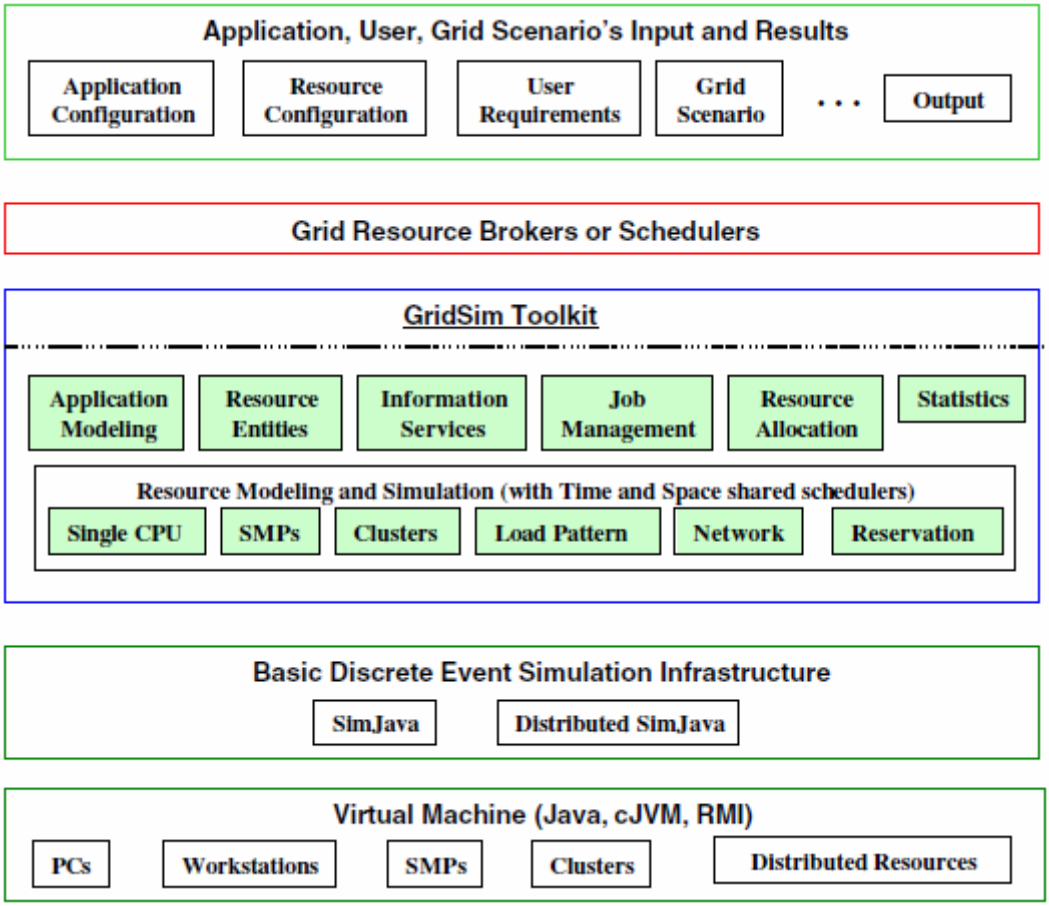

Рис. 7. Многоуровневая архитектура GridSim

Третий уровень — непосредственно GridSim. Она моделирует основные объекты Grid такие, как ресурсы и информационные сервисы. Этот уровень содержит подуровень моделирования ресурсов и определения топологии сети. Статистические данные автоматически собираются и доступны через объект GridStatistics. Также на этом уровне находятся объекты для моделирования DataGrid-ReplicaCatalogue и ReplicaManager.

Следующий уровень определяет функциональные возможности для планировщиков и брокеров ресурсов. Верхний уровень предназначен для конфигурирования параметров моделирования.

Открытый исходный код и описание классов позволяют легко осуществить реализацию собственных алгоритмов планирования, передачи данных и т.д.

Функционирование системы

Выполнение задачи в GridSim происходит следующим образом (рис. 8). Объект GridUser представляет пользователя. Задача от пользователя направляется на соответствующий Брокер Ресурсов и в виде объекта Gridlet. Gridlet является пакетом, который содержит всю информацию, связанную с задачей — длину задачи (выраженную в миллионах операций, MI), размер входных и выходных файлов, характеристики пользователя. Эти основные параметры помогают определить время выполнения, время, требуемое на передачу файлов и возвращение обработанного Gridlets создателю вместе с результатами. Брокер ресурсов запрашивает у GridInformationService (GIS) список доступных GridResource. GridInformationService регистрирует ресурсы, что дает возможность отслеживать список доступных ресурсов Grid. После чего, Брокер ресурсов выполняет выбор ресурса и направляет Gridlet на обработку, согласно установленному сценарию.

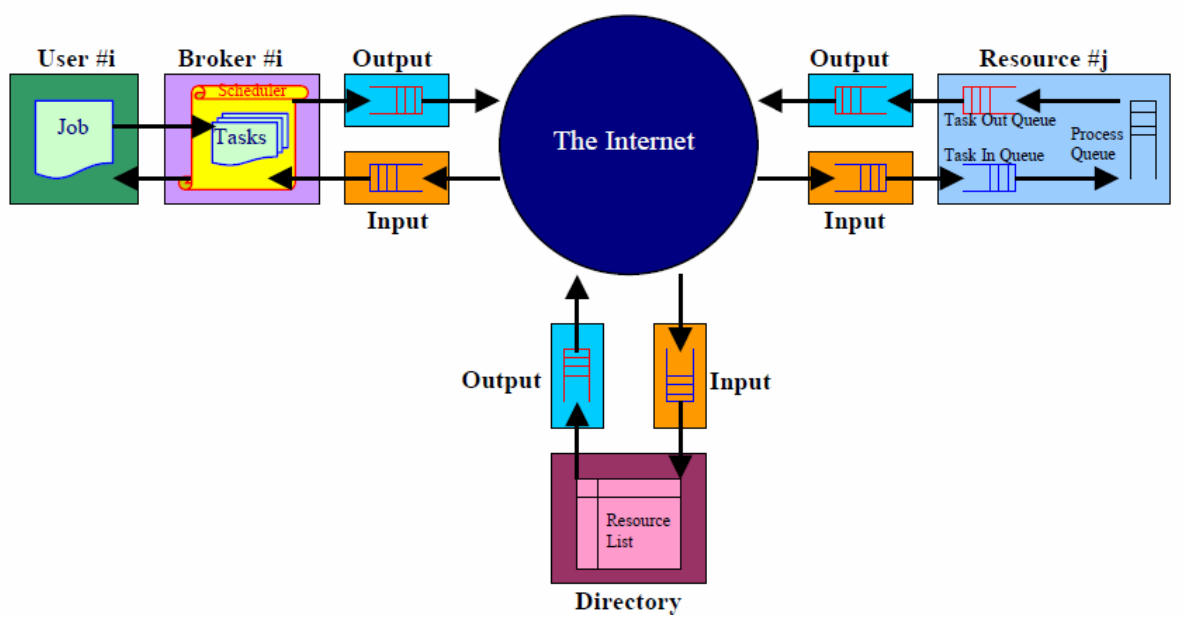

Рис.8. Диаграмма потоков данных в GridSim

GridResource — объект, составленный из одной или более машин. Машина содержит один или более CPU (processing elements, PE), для каждого из которых мощность задается количеством операций в секунду (millions of instructions per second (MIPS)). После выполнения Gridlet, GridResource сообщает Брокеру о завершении задачи. Следует отметить, что ресурс может выполнять несколько задач одновременно.

Статистика эксперимента

GridSim позволяет собрать статистику по всем или отобранным операциям, что зависит от того, что конкретно интересует пользователя. Это может быть и мониторинг дискового пространства, и размеры очередей, и загруженность сети и многое другое. В самом простом случае, можно выводить всю интересующую информацию в файл и после моделирования использовать утилиты работы с текстом (например, grep или awk), чтобы отобрать данные необходимые для дальнейшего анализа. Для графического представления результатов можно использовать Gnuplot.

Копирование данных

Начиная с версии 4.0, GridSim позволяет моделировать репликацию данных в Grid. У каждого объекта Файл есть связанный объект FileAttribute, который содержит информацию о размере файла, время создания и изменения, мастер-файл или копия и т.д. Replica Catalogue (RC) — отслеживает местоположение (копии) файла. RC обеспечивает соответствие между именем файла и его физическим расположением.

DataGridResource позволяет пользователям как запускать задачи, так и получать доступ к данным. DataGridResource предполагает использование следующих носителей данных: HarddriveStorage и TapeStorage. Объект DataGridlet представляет собой задачу, для выполнения которой требуется один или больше файлов.

Абстрактный класс ReplicaManager и класс SimpleReplicaManger реализуют основные функциональные возможности копирования данных. Эти классы отвечают за все действия с данными на DataGridResources, что включает:

- добавление файла или его копии,
- регистрация файла на RC,
- удаление файла или его копии, пересылка требуемого файла к соответствующему DataGridResource.

2. Сравнение систем

Очевидно, что DataGrid симулятор должен обладать следующим функционалом и возможностями: моделированием основных элементов DataGrid (ресурсов хранения данных (SE), брокеров ресурсов (RB), каталога реплик (RC), сети, пользователей (User), сайтов); скорость работы модели должна значительно превосходить скорость работы реальной DataGrid; необходима статистика по отдельным элементам (например, использование дисковых ресурсов) и по работе модели в целом (время выполнения, количество переданных файлов, загрузка сети и др.); необходимо моделирование сбоев оборудования; результаты моделирования должны быть сопоставимы с реальной ситуацией.

Мы обозначили основные возможности и особенности систем моделирования DataGrid. Сравним системы по вышеперечисленным требованиям.

2.1. Моделирование основных элементов DataGrid

Пользователи (users) могут быть смоделированы различными способами. В OptorSim все пользователи Grid представлены одним объектом User, который, в соответствии с распределением, создает задачи и направляет их на Брокер ресурсов, который направляет задачу на сайт. В системе Bricks объект *Client* представляет пользователя, который последовательно запускает задачи, в соответствии с установленной частотой. Совершенно другой подход реализован в GridSim, где для пользователя можно определить какие задачи он будет запускать и время запуска этих задач. Последний вариант выглядит наиболее привлекательно, но предполагает дополнительную доработку программы.

OptorSim не рассматривает необходимость в вычислительных ресурсах для выполнения задачи. Предполагается, что время выполнения задачи складывается из времени поиска подходящего файла и его копирования. Bricks и GridSim позволяют моделировать гибридные ресурсы, которые могут содержать и вычислительные элементы и ресурсы хранения данных.

Брокеры ресурсов используются для назначения заданий на ресурсы. Для определения необходимых для решения задачи ресурсов могут использоваться различные алгоритмы планирования. В OptorSim и Bricks используется Брокер ресурсов, который просто выбирает лучший ресурс согласно выбранному алгоритму. Несколько алгоритмов работы Брокеров предложено в этих системах. В GridSim каждый пользователь, может иметь свой брокер ресурсов, который будет выбирать оптимальные ресурсы согласно пожеланиям пользователя. GridSim не предлагает готовых алгоритмов работы Брокеров ресурсов.

OptorSim позволяет довольно упрощенно описать сеть — несколько сайтов соединяются друг с другом. Каждый сайт служит маршрутизатором, для пересылки данных соседям. Bricks позволяет реализовать сложные топологии сетей, но эта процедура достаточно трудоемка. GridSim предлагает инструменты для наиболее полного описания топологии Grid-сети: связи между элементами, роутеры, пакетная передача данных, MTU и т. д.

2.2. Статистика

Статистическая информация, собранная во время моделирования необходима для оценки качества построенной модели и последующего анализа результатов моделирования.

Информация об объектах Grid и их использовании в GridSim хранится в GridStatistics. Статистические данные о работе пользователей, ресурсов или роутеров могут быть получены. При желании могут быть собраны данные по всем интересующим параметрам модели, но это требует дополнительных усилий со стороны пользователя. Подобная статистика может быть доступна в Bricks. OptorSim предоставляет необходимую статистическую информацию без дополнительной доработки программы со стороны пользователя. Используя графический интерфейс, пользователь может наблюдать состояние модели даже в процессе ее выполнения, а после выполнения пользователь может получить полный отчет с графиками, диаграммами и таблицами о результатах моделирования.

2.3. Моделирование сбоев оборудования

Моделирование различного рода сбоев и отказов не представлено ни в Bricks, ни в OptorSim. GridSim дает возможность моделировать отказы ресурсов (рис. 9).

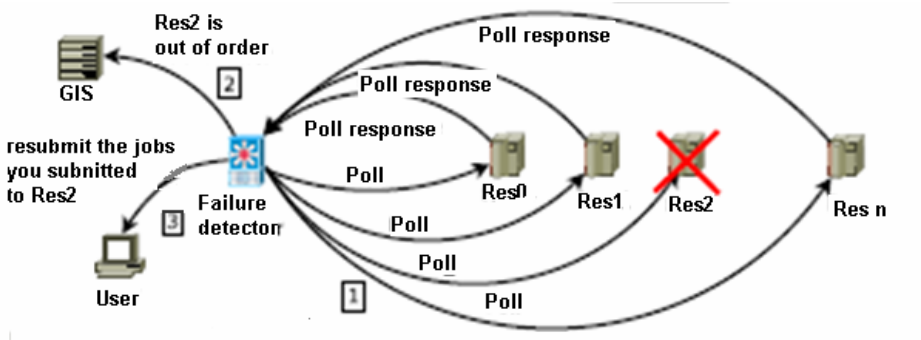

Рис.9. Возникновение неисправности ресурса

Детектор ошибок регулярно опрашивает элементы системы. При возникновении ошибки, информация о неисправном ресурсе направляется в информационный центр, который содержит информацию о состоянии всех элементов системы. Запуск задач на неисправном ресурсе будет приостановлен, до восстановления ресурса. Задачи, которые выполнялись на ресурсе, в случае неисправности, будут перенаправлены на другие ресурсы.

Обнаружение ошибок и восстановление работы ресурсов неотъемлемая часть DataGrid сетей. Возможность моделирования подобных ситуаций позволит исследовать их и улучшить работу реальных Grid. Однако, отказы ресурсов не единственная проблема DataGrid. При передаче данных могут возникать ошибки — по таймаутам, программные ошибки, ошибки пользователей и приложений [3]. Логичным путем уменьшения числа таких ошибок представляется своевременное реагирования на их появление, повышение квалификации пользователей, постоянные эксперименты и отслеживание изменений в глобальной инфраструктуре. Моделирование таких ошибок необходимо для адекватного представления процессов в DataGrid.

Для дальнейшей работы в области моделирования ошибок и сбоев в DataGrid наиболее подходит GridSim, поскольку этот пакет имеет определенные наработки в этой области.

Выводы

Несомненно, для моделирования сложных Grid-сетей наиболее подходит GridSim. Этот пакет предлагает широкие возможности для моделирования DataGrid и наиболее полного описания модели. OptorSim позволяет моделировать передачу данных, но не предполагает пакетированную передачу данных. В OptorSim реализовано разделение канала связи между несколькими пользователями, при которой пропускная способность канала делится между всеми пользователями одинаково и не зависит от потребностей этих пользователей. Пакет моделирования Bricks нацелен на исследование очередей и пропускных способностей каналов связи. Каталог реплик в Bricks моделируется как центральный компонент и не предполагает какой-либо иерархии. Использование конфигурационных файлов для задания параметров модели в Bricks и в OptorSim накладывает существенные ограничения при создании модели.

Заключение

Рассмотрены пакеты моделирования Bricks, OptorSim и GridSim. Bricks позволяет изучить очереди и пропускные способности сетей. OptorSim предполагает симуляцию алгоритмов доступа к данным и стратегий репликации. GridSim дает возможность достаточно полно описать исследуемую модель, различные классы гетерогенных ресурсов, пользователей, приложений, брокеров ресурса, и планировщиков.

OptorSim и Bricks не предусматривают моделирование различного рода ошибок. В GridSim возможно моделирование отказа ресурсов, но этого явно не достаточно. Например, наиболее распространенные ошибки, возникающие при передаче данных в EGEE/WLCG при использовании File Transfer Service (FTS) [24, 10] являются ошибки по таймаутам, программные ошибки, специфические ошибки приложений и ошибки пользователей, моделирование которых необходимо для правильного и адекватного отражения реальных процессов в Grid.

Представленные системы нуждаются в доработке для корректного моделирования процессов передачи данных в DataGrid.

Список литературы

1. Англо-русский словарь основных терминов и сокращений по телекоммуникациям [Электронный ресурс]. — Режим доступа: http://www.terms.com.ua/pages/telecom_dict.html (дата обращения: 22.01.2009).
2. Ильин, В. А. Российский сегмент глобальной инфраструктуры LCG [Текст] / В. А. Ильин, В. В. Кореньков, А. А Солдатов // Открытые системы. — 2003. — № 1.
3. Кореньков, В. В. Архитектура сервиса передачи данных в grid / В. В. Кореньков, А. В. Ужинский // Открытые системы. — 2008. — № 2.
4. Bricks: A Performance Evaluation System for Grid Computing Scheduling Algorithms [Electronic resource]. — URL: http://ninf.apgrid.org/bricks/ (дата обращения: 16.10.2008).
5. Buyya, R. An architecture for a resource management and scheduling system in a global computational grid / R. Buyya, D. Abramson, J. Giddy, G. Nimrod // Proc. of the 4th International Conference and Exhibition on High Performance Computing in Asia-Pacific Region, Beijing, China, 2000.
6. Cameron, D. UK grid simulation with optorsim / D. Cameron, R. Carvajal-Schia, A. Millar, C. Nicholson, K. Stockinger, F. Zini // e-Science All-Hands Meeting, Nottingham, September, 2003.
7. Cameron, D. Evaluating scheduling and replica optimisation strategies in optorsim / D. Cameron, R. Carvajal-Schia, A. Millar, C. Nicholson, K. Stockinger, F. Zini // 4th International Workshop on Grid Computing (Grid2003) // IEEE Computer Society Press, Phoenix, Arizona, November 17, 2003.
8. Caminero, A. Extending GridSim with an Architecture for Failure Detection / A. Caminero, A. Sulistio, B. Caminero, C. Carrion, R. Buyya // Proceedings of the 13th International Conference on Parallel and Distributed Systems (ICPADS'07), Hsinchu, Taiwan 2007.
9. David, G. OptorSim v2.1: Installation and User Guide / G. David, D. Cameron, Oct, 2006.
10. File Transfer Service [Electronic resource]. — URL: https://twiki.cern.ch/twiki/bin/view/EGEE/FTS (дата обращения: 10.09.2008).
11. Grid eXplorer [Electronic resource]. — URL: http://www.lri.fr/~fci/GdX/ (дата обращения: 18.10.2008).
12. GridSim: A Grid Simulation Toolkit for Resource Modelling and Application Scheduling for Parallel and Distributed Computing [Electronic resource]. — URL: http://www.gridbus.org/gridsim/ (дата обращения: 03.12.2009).
13. LHC Homepage [Electronic resource]. — URL: http://lhc.web.cern.ch/lhc/ (дата обращения: 11.10.2008).
14. MicroGrid: Online Simulation Tools for Grids, Distributed Systems and the Internet [Electronic resource]. — URL: http://www-csag.ucsd.edu/projects/grid/microgrid.html (дата обращения: 10.10.2008).
15. SimGrid [Electronic resource]. — URL: http://simgrid.gforge.inria.fr/ (дата обращения: 12.11.2008).
16. SimJava // Institute for Computing Systems Architecture (ICSA), School of Informatics, University of Edinburgh. [Electronic resource]. — URL: http://www.icsa.informatics.ed.ac.uk/research/groups/hase/simjava/ (дата обращения: 15.12.2008).

17. Simulating data access optimization algorithms // OptorSim [Electronic resource]. — URL: http://edg-wp2.web.cern.ch/edg-wp2/optimization/optorsim.html (дата обращения: 01.12.2008).
18. Sulistio, A. A toolkit for modelling and simulation of data Grids with integration of data storage, replication and analysis / A. Sulistio , U. Cibej, R. Buyya, B. Robic // Technical Report, GridS-TR-2005-13, GridS Lab, University of Melbourne, Australia, November 8, 2005.
19. Sulistio, A. A taxonomy of computer-based simuylations and its mapping to parallel and distributed systems simulation tools / A. Sulistio, C. Yeo, R. Buyya // Softw., Pract. Exper. — 2004. — vol. 34, №7. — P. 653-673.
20. Takefusa, A. A study of deadline scheduling for client-server systems on the computational grid / A. Takefusa , S. Matsuoka, H. Casanova, F. Berman // High Performance Distributed Computing, IEEE Computer Society. — 2001. — P. 406-405.
21. Takefusa, A. Overview of a performance evaluation system for global computing scheduling algorithms / A. Takefusa, S. Matsuoka, H. Nakada, U. Nagashima // High Performance Distributed Computing, IEEE Computer Society, 1999. — P. 97-104.
22. Takefusa, A. Performance analysis of scheduling and replication algorithms on grid datafarm architecture for highenergy physics applications / A. Takefusa, O. Tatebe, S. Matsuoka, Y. Morita // High Performance Distributed Computing, IEEE Computer Society, 2003. — P. 34-47.
23. The DataGrid Project [Electronic resource]. — URL: http://eu-datagrid.web.cern.ch/eu-datagrid/ (дата обращения: 12.12.2008).
24. The gLite File Transfer Service// Enabling Grids for E-science [Electronic resource]. — URL: http://egee-jra1-dm.web.cern.ch/egee-jra1-dm/FTS/default.htm (дата обращения: 02.02.2009).
25. Worldwide LHC Computing Grid [Electronic resource]. — URL: http://www.cern.ch/lcg (дата обращения: 12.01.2009).