

СПОСОБ ПОСТРОЕНИЯ ГРИД ИЗ НЕКЛАСТЕРИЗОВАННЫХ РЕСУРСОВ*

Березовский П.С., Коваленко В.Н.

Институт прикладной математики им. М.В.Келдыша РАН;
Россия, 125047, Москва, Миусская пл. 4; тел. (495)250-79-01,
bps@keldysh.ru, kvn@keldysh.ru

1. Введение

Концепция распределенной среды грид становится все более популярной. В рамках известных типов грида (вычислительный грид, грид данных, информационный грид и т.д.) появляются новые формы. Наиболее активно развивается вычислительный грид, для которого можно указать формы, различающиеся: по типу использования ресурсов (отчуждаемые и неотчуждаемые ресурсы); по типу объединения ресурсов (кластеризованные и некластеризованные ресурсы); а также по классу заданий (однопроцессорные, параллельные и сериализуемые). Наибольшее распространение получил метод построения грид-инфраструктур из находящихся в разных точках сети кластерных узлов, в которые объединяется множество компьютеров, принадлежащих одному административному домену – двухуровневая организация грид [1]. Такой способ применен, например, в проекте gLite [2], реализованном на базе инструментария Globus Toolkit [3]. В нем предполагается, что компьютеры должны полностью выделяться в грид и не могут использоваться сотрудниками тех организаций, в которых они установлены. Все ресурсы объединены в кластеры и находятся под управлением локального менеджера ресурсов. Эта форма грида наиболее развита, так как сама концепция зародилась и развивалась в научных организациях, обладающих развитой кластерной инфраструктурой. Тем не менее, для такой формы организации грида разрабатывается программное обеспечение, делающее возможным использовать эти ресурсы, не отчуждая их от владельцев [9].

Однако помимо кластеров имеются вычислительные ресурсы – рабочие станции пользователей, домашние компьютеры, серверы приложений и т.д., суммарная производительность которых во много раз превосходит производительность самого мощного суперкомпьютера. Следует отметить, что их средняя загрузка составляет по имеющимся оценкам 10%, а в ночные часы они вообще простаивают. Такие компьютеры обладают небольшой производительностью и непредсказуемым временем доступности ресурсов, однако, их очень много. Поэтому они представляют большой интерес для пользователей грида, которые могли бы решать свои вычислительные задачи, используя эти ресурсы, но не имеют возможности организовывать и поддерживать кластерную инфраструктуру.

* Работа выполнена при поддержке Российского фонда фундаментальных исследований (проект 06-07-89111-а), программы Президента РФ для ведущих научных школ (проект 2006-РП-112.0/001/417) и программы фундаментальных исследований Президиума РАН «Разработка фундаментальных основ создания научной распределённой информационно-вычислительной среды на основе технологий GRID».

2. Постановка задачи

Целью настоящей статьи является разработка принципов построения системы диспетчеризации, осуществляющей распределение заданий в гриде с некластеризованными ресурсами, которые используются совместно с их владельцами, т.е. не отчуждаются целиком в грид. Для достижения поставленной цели необходимо решить следующие задачи:

- разработать программную архитектуру системы диспетчеризации заданий;
- определить функциональность каждого компонента программной архитектуры;
- разработать механизмы, необходимые для взаимодействия компонентов грид-инфраструктуры из некластеризованных ресурсов;
- обеспечить интероперабельность разрабатываемой системы с подобными системами, уже функционирующими в гриде.

Задачи решаются в следующих условиях:

- на вычислительные узлы поступает два типа заданий – глобальные задания от пользователей грида, а также приложения владельца этого узла, причем приоритет процессов владельца выше;
- компьютеры, составляющие ресурсную базу грид, пространственно распределены;
- должна обеспечиваться автономность ресурсов – объем ресурсов, получаемых заданиями грида, должен быть лимитирован;
- динамичность среды – непрогнозируемые выключения и включения отдельных ресурсов.

Помимо этого, предъявляются требования к программной составляющей системы, которая будет устанавливаться на компьютер владельца: она должна быть максимально компактной, расходовать минимум системных ресурсов, а ее установка и администрирование не должны требовать от пользователя дополнительных знаний и навыков.

3. Существующие программные решения

На сегодняшний день идея объединения персональных компьютеров для решения больших задач становится все более популярной. Такие задачи решаются как на глобальном уровне, так и в рамках одного предприятия (корпоративные системы). В качестве примера систем первого класса можно отметить разработку, которая послужила началом применения одноуровневой схемы организации пространственно распределенных ресурсов. Это проект Калифорнийского университета SETI@Home – поиск внеземного разума. Участникам проекта предлагалось скачать с веб-сайта университета небольшую программу, работающую как хранитель экрана. Во время простоя компьютера эта программа получала данные радиотелескопических наблюдений с сервера, анализировала их и отправляла результат обратно. Радиотелескоп генерировал большое количество информации, которая могла быть разбита на небольшие порции, которые можно обрабатывать независимо. Позднее появилось множество проектов из серии @Home, предназначенных для решения

конкретной задачи. Разработки университета вылились в создание программной системы BOINC [4], позволяющей организовывать новые проекты для новых задач.

Второй класс систем – это корпоративные решения, нацеленные на повышение эффективности работы машинного парка предприятия. Так, в системе Entropia [<http://www.entropia.com>] предложен новый подход к объединению персональных компьютеров для обработки сериализуемых заданий. Программная архитектура состоит из сервера и множества клиентов, на которых установлена агентская часть системы. Компонента сервера делит задание на множество подзадач и запускает их на клиентских машинах. После завершения вычислений результаты выполнения подзадач собираются вместе и возвращаются пользователю. Здесь уже нет ограничения на одно приложение, список доступных приложений может легко расширяться. В данном случае разрабатывается специальное программное обеспечение, протоколы взаимодействия, программная и аппаратная инфраструктура для того, чтобы использовать компьютеры организации для решения ресурсоемких задач (например, анализ финансовых рисков) в те часы, когда они слабо загружены, либо простаивают.

4. Способы построения грид-сегментов

Текущее состояние в области грид

Среди основных компонентов программной архитектуры грид мы выделяем программные средства для запуска заданий, хранения и доставки данных, средства обеспечения безопасности, а также средства информационной поддержки. Перечисленные программные средства образуют набор функциональных компонентов, необходимых для построения грид-системы. В настоящее время, как правило, запуск заданий на исполнительных узлах и управление ресурсами осуществляется локальными системами управления пакетной обработкой. С помощью таких средств, как GRAM, осуществляется запуск задания через локальный менеджер ресурсов, который сам определяет, на каком исполнительном узле будет выполняться задание. В качестве наиболее известных систем можно отметить PBS [6], LSF [7] и Condor [8].

Для передачи данных в гриде используются протокол RFIO (Remote File I/O) и протокол GridFTP. При создании протокола GridFTP за основу был взят уже существующий и популярный протокол передачи данных FTP, который был расширен необходимыми для грида функциями. Основными особенностями протокола GridFTP являются: безопасность каналов передачи данных (использование при аутентификации GSI – Grid Security Infrastructure), использование различных каналов при параллельных передачах данных, передача частей файлов, передача от сервера к серверу по инициации клиента (third-party transfer).

Безопасность в любой распределенной среде с коллективным доступом играет важную роль. В существующих грид-системах эта проблема решается применением стандартных механизмов и способов обеспечения безопасности: использованием сертификатов, применением политик распределения ресурсов, разделением доступа к ресурсам между участниками различных виртуальных организаций, делегированием полномочий.

Информационная система осуществляет сбор информации о ресурсах грид-сегмента, а информационные поставщики предоставляют эту информацию в объемлющую грид-инфраструктуру. Поскольку направление грид активно развивается, вместе с ним развивается и изменяется программное обеспечение. В частности, информационные системы на примере инструментария Globus Toolkit прошли путь от статического хранения информации, когда информационные поставщики (GRIS) передавали данные по протоколу LDAP на вышележащие уровни агрегации (GIIS), до применения реляционных баз данных (R-GMA).

Объединение некластеризованных ресурсов

Рассмотрим, какие существуют способы включения отдельных компьютеров в грид-инфраструктуру. В предыдущем разделе были описаны варианты решения задачи объединения некластеризованных ресурсов, предлагаемые различными разработчиками программного обеспечения грид. Учитывая эти варианты, можно предложить два возможных способа создания инфраструктуры грид из некластеризованных вычислительных ресурсов:

- объединить географически распределенные ресурсы с помощью специального ПО для решения конкретной задачи – общественный компьютеринг [5];
- объединить ресурсы в рамках предприятия для решения набора заранее определенных задач.

В предыдущем разделе было рассказано о некоторых системах, в которых используются некластеризованные ресурсы, но предлагаемые в них решения являются ограниченными и не удовлетворяют стандартам грид. Во-первых, эти системы не имеют средств запуска заданий. Для каждой новой задачи необходимо создавать свой «экземпляр» ресурсной и серверной инфраструктуры. Во-вторых, не удовлетворяя стандартам грид, такие системы не являются интероперабельными. В-третьих, отсутствует информационная служба, предоставляющая информацию о ресурсах в грид. Наличие такой информационной службы необходимо для включения сегмента из некластеризованных ресурсов в грид-инфраструктуру более высокого уровня.

Предлагаемое решение

Для объединения некластеризованных ресурсов и включения их в состав грид-инфраструктуры мы предлагаем систему диспетчеризации, осуществляющую распределение заданий в гриде с некластеризованными ресурсами, которые используются совместно с их владельцами. Архитектура программного обеспечения состоит из трех компонентов: управляющего узла (УУ), исполнительного узла (ИУ) и пользовательского интерфейса (ПИ). Управляющий узел – это выделенный сервер в инфраструктуре сегмента, к которому подключается множество исполнительных узлов. Для того чтобы сделать компонент, устанавливаемый на исполнительный узел, компактным почти все функции как управления сегментом, так и взаимодействия компонентов необходимо реализовать на управляющем узле.

Чтобы сделать грид-сегмент из некластеризованных ресурсов доступным для пользователей грид, необходимо выполнение двух условий. Во-первых, информация о количестве доступных ресурсов должна публиковаться в информационной системе

объемлющей грид-инфраструктуры, а во-вторых, пользователи должны иметь возможность запуска своих заданий. Первая задача решается наличием информационной службы, которая предоставляет такую информацию, а вторая – предоставлением пользовательского интерфейса.

Мы выделяем следующие основные события, происходящие в сегменте грида с некластеризованными ресурсами:

- подключение нового исполнительного узла;
- выключение исполнительного узла;
- поступление очередного глобального задания;
- освобождение ресурса;
- запуск глобального задания.

Далее рассмотрим все три программные компоненты и поясним, какие механизмы используются при обработке того или иного события. Особое внимание уделим циклу обработки поступившего задания.

Управляющий узел

Проводя аналогию с кластерами, входящими в состав грида, управляющий узел можно назвать шлюзом сегмента. В частности, на нем реализованы необходимые базовые компоненты программного обеспечения грида, а также некоторые специальные механизмы, необходимые для работы с некластеризованными ресурсами. В состав управляющего узла входят следующие службы:

- информационная служба;
- служба взаимодействия с исполнительными узлами;
- служба поддержки пользователей;
- служба диспетчеризации (распределения заданий);
- служба мониторинга;
- служба передачи данных;
- служба по работе с базой данных.

Информационная служба (ИС) выполняет функции по хранению информации о пользователях грида, глобальных заданиях и свободных ресурсах исполнительных узлов. Компоненты ИС – поставщики информации – собирают информацию о доступных ресурсах исполнительных узлов, заносят ее в информационную базу, а затем преобразуют эту информацию для публикации ее в гриде. Для хранения информации используется реляционная база данных.

В информационной базе хранится как первичная информация по каждому узлу и агрегированная информация для всего сегмента, так и накопленная статистика о доступности ресурсов каждого узла. Владелец ресурса может как частично, так и полностью отдавать свой компьютер для пользователей грида. Если отчуждаемые ресурсы управляются сервером, то неотчуждаемые могут быть внезапно заняты их владельцем. Предсказать, когда это произойдет, невозможно, поэтому нельзя ориентироваться на моментальные показатели доступных ресурсов и необходимо отдельно рассматривать отчуждаемые и неотчуждаемые ресурсы.

В случае с отчуждаемыми ресурсами информация публикуется отдельно для каждого ресурса, для неотчуждаемых информация интегрируется на основе статистических данных. Полученная информация затем анализируется, группы ресурсов объединяются по какому-то общему признаку (например, по архитектуре, производительности процессора или операционной системе), и в рамках каждой группы накапливается статистика использования этих ресурсов. На основе собранной статистики можно делать предположения о количестве ресурсов той или иной группы узлов, которые будут доступны в течение некоторого промежутка времени. Наличие такого рода прогноза может повысить качество распределения заданий по ресурсам.

Агрегированная информация публикуется также во внешних информационных системах грида для того, чтобы ресурсы сегмента были видны всем остальным участникам грида – пользователям и брокерам ресурсов. Если внутри системы информация может храниться и представляться в произвольном виде, то для публикации информации в объемлющей инфраструктуре необходимо использовать стандартные и общепринятые протоколы. Соответствующим преобразованием также занимаются поставщики информации. Обновление агрегированной информации происходит в зависимости от интенсивности изменения ситуации, а также по запросу внешних систем на получение информации о ресурсах сегмента.

Служба взаимодействия с исполнительными узлами осуществляет связь центрального сервера – управляющего узла – с компьютерами владельцев, на которых будут выполняться задания. Принимая во внимание тот факт, что некластеризованных ресурсов очень много, данная служба должна предоставлять механизм автоматической регистрации исполнительных узлов. При регистрации нового ресурса агент, расположенный на исполнительном узле, посылает файл с описанием компьютера и количеством предоставляемых ресурсов, который обрабатывается службой. Таким образом, служба взаимодействия с ИУ выполняет следующие функции:

- предоставляет механизм автоматической регистрации исполнительных узлов;
- разбирает файл с описанием исполнительного узла и заносит полученную информацию в базу данных;

Служба поддержки пользователей необходима для приема глобальных заданий и обработки запросов пользователя. К ней обращаются пользователи грид и брокеры ресурсов, чтобы запустить свое задание и управлять им. В ответ пользователю возвращается идентификатор задания, используя который он может управлять своим заданием, посылая соответствующие команды (снять или приостановить выполнение задания) службе. Также служба поддержки пользователей осуществляет разбор пользовательских запросов и добавляет информацию о новом задании в локальную информационную систему.

Служба планирования (планировщик) периодически запускает процесс поиска «подходящего» узла для каждого находящегося в очереди глобального задания и отправляет его на выполнение, если такой узел найден. Под «подходящим» узлом мы понимаем узел, который удовлетворяет статическим требованиям задания (архитектура, операционная система, дисковое пространство, оперативная память и пр.), свободен на данный момент, его производительность гарантирует выполнение задания в срок, а пользователь обладает полномочиями запуска заданий на этом узле.

Для ограничения доступа пользователей к конкретным ресурсам используется подход, применяющийся в инструментарии Globus Toolkit, который основан на использовании сертификатов.

Учитывая динамичность среды (частое изменение состояния ресурсов), а также то, что характер прогноза не является точным, можно повысить скорость исполнения задания за счет запуска нескольких его копий на различных узлах. Например, запускать задание на все подходящие ресурсы, а с приходом новых заданий, претендующих на те же ресурсы, анализировать прогресс исполнения задания и снимать со счета часть его копий. При этом решаются две задачи: 1) вероятность успешного завершения задачи увеличивается и 2) наличие нескольких результатов, позволяет сравнивать их и предотвращать подделку результата недобросовестными владельцами исполнительных узлов.

Служба мониторинга контролирует выполнение заданий, предоставляет пользователю информацию о задании, данные о количестве потребленных заданием ресурсов и т.д. После того, как задание было отправлено на исполнительный узел, система мониторинга осуществляет слежение за ним, периодически обновляя информацию о задании (в каком состоянии оно находится) и исполнительном узле (сколько доступно дискового пространства, насколько загружен процессор и т.д.), хранящуюся в базе данных. Если задание слишком долго находилось в очереди и не успевает выполниться в указанный пользователем срок, планировщик осуществляет повторную диспетчеризацию задания.

Служба передачи и хранения данных осуществляет доставку файлов по безопасному протоколу, удовлетворяющему стандартам грид. Применение стандартных протоколов и служб грид позволяет использовать такие механизмы как «third-party transfer», а также передачу части файла. Такая возможность может быть полезна при миграции (перезапуске) задания на другой исполнительный узел, т.к. при передаче не будет нагрузки на управляющий узел, а весь поток данных будет передаваться напрямую между исполнительными узлами. Система передачи необходима также для временного хранения файлов заданий и входных данных до завершения выполнения задания.

Если доставка задания и необходимых данных осуществляется через сервер, т.е. не используются внешние ресурсы хранения, то само задание и необходимые ему данные нельзя сразу после запуска удалять из временного хранилища. Если задание будет снято со счета по причине какой-либо ошибки или из-за того, что исполнительный узел стал недоступен (владелец целиком занял его своими заданиями или просто выключил), то необходим весь набор информации для перезапуска задания на другом узле. Для этих целей можно использовать специализированный узел хранения (storage). Таким образом, служба передачи данных реализует следующие функции:

- осуществляет передачу входных данных и исполняемых файлов задания на ИУ;
- осуществляет доставку результата по адресу, который указал пользователь в свойствах своего задания;
- временно хранит все необходимые заданию данные до его завершения.

Для обеспечения независимости системы от платформы функциональные компоненты на управляющем узле реализуются с помощью технологии веб-служб. Это стандарт, который, во-первых, широко применяется при построении распределенных систем, а во-вторых, уже используется при разработке грид-приложений. Кроме того, реализация подсистем в виде веб-служб позволяет получить модульную систему, в которой в любое время один блок может быть заменен другим (например, планировщик или служба приема заданий). Для этого достаточно сохранить интерфейсы взаимодействия с остальными модулями системы.

Исполнительный узел

Агент устанавливается на компьютерах владельцев, которые хотят предоставить свои ресурсы для нужд грид.

Все ресурсы исполнительного узла делятся между процессами владельца этого узла и глобальными заданиями. При регистрации исполнительного узла его владелец указывает, какое количество ресурсов каждого типа (процессорное время, дисковое пространство, оперативная память и т.д.) он отчуждает в грид-инфраструктуру. При этом задание не может потребить больше ресурсов, чем было заявлено в его запросе, несмотря на то, что свободные ресурсы на исполнительном узле фактически еще могут оставаться. Как только задание выходит за границы потребления ресурсов, указанных в запросе (например, занимает больше дискового пространства или выполняется дольше, чем было заявлено), его выполнение должно прекращаться.

На исполнительном узле реализуются непосредственный запуск задания и управление им, осуществляется разделение ресурсов между процессами локальных пользователей и глобальными заданиями, обеспечивается передача данных, а также осуществляется необходимый уровень безопасности. Безопасность рассматривается в трех аспектах:

- защита владельца, то есть обеспечение полнофункциональности машины путем контроля уровня потребления ресурсов внешним приложением;
- защита конфигурации машины, находящихся на ней программ и данных;
- защита приложения грид, включая программу, данные и результаты.

В отличие от традиционной схемы работы в гриде, когда диспетчер сам распределяет задания по исполнительным узлам и запускает их в очередь локального менеджера ресурсов, здесь иной подход. Поскольку исполнительные узлы (компьютеры владельцев) не отчуждаются в грид, то для того, чтобы отправить на выполнение очередное задание, диспетчеру пришлось бы постоянно опрашивать ресурс и узнавать, готов ли он к приему нового задания. Такой подход, во-первых, труден для масштабирования, т.к. изначально сервер не имеет информации о вновь подключенных узлах, а, во-вторых, увеличивает загрузку диспетчера (управляющего узла сегмента). Поэтому был использован подход, при котором инициация приема задания исходит от агента на исполнительном узле.

На исполнительном узле реализованы следующие системы:

- система администрирования;
- система передачи данных;

- система запуска задания;
- система разделения ресурсов;
- система управления заданиями.

Система администрирования позволяет владельцу настроить политику использования его ресурсов заданиями грид. Он может указать, в каких пропорциях потребляют ресурсы задания грид и локальные задания пользователей, установить порог использования ресурсов, при котором компьютер считается свободным и готовым для приема заданий грид. Например, если владельцу необходимо, чтобы глобальные задания не загружали процессор более чем на 80%, то ему необходимо указать это в конфигурационном файле, и как только это значение будет превышено, вычисление глобального задания приостановится и возобновится только при снижении нагрузки. Как только выполнение глобального задания на ИУ завершается, система сообщает серверу о том, что узел готов к приему очередного задания (обращается к службе на УУ), и «получает задание». Такой способ получения заданий позволяет минимизировать на исполнительном узле количество серверных компонентов.

Система передачи данных на исполнительном узле осуществляет доставку результата на сервер или передачу задания и входной информации на другой узел в случае перепланирования. Серверная часть системы передачи данных расположена на управляющем узле, доступ к ней осуществляется через клиентские компоненты с исполнительного узла.

Система запуска заданий получает с сервера идентификатор задания, по которому она определяет, где находится файл задания и необходимые заданию входные данные. С помощью системы передачи данных они доставляются на исполнительный узел, после чего система запуска заданий формирует на компьютере исполнительную среду, в которой будет выполняться задание. Затем осуществляется непосредственный запуск задания, и система контролирует его выполнение. В целях безопасности и корректности выполнения заданий на ИУ, задание грид изолируется от процессов владельца ресурса. Само задание, а также входные и выходные данные могут шифроваться, чтобы исключить доступ к ним со стороны владельца ресурса.

Система разделения ресурсов осуществляет контроль за потребленными заданием ресурсами и обеспечивает их эффективное использование, учитывая неотчуждаемость от владельца. Периодически данные об использовании ресурсов заносятся в информационную базу. Если задание превысило выделенный ему лимит ресурсов, то оно снимается. Если пользователь активизировал свою деятельность, и загрузка компьютера превысила определенный уровень, то задание пользователя приостанавливается. После снижения нагрузки выполнение задания продолжается.

Система управления заданием позволяет приостановить, возобновить, а также прекратить выполнение задания.

Таким образом, на исполнительном узле задание обрабатывается по следующей схеме:

1. По запросу агента сервер передает идентификатор задания.
2. Используя систему передачи данных, по идентификатору задания доставляются само задание и необходимые данные.

3. Система запуска заданий формирует исполнительную среду, запускает задание и передает идентификатор системного процесса, соответствующего глобальному заданию, системе разделения ресурсов.
4. Система разделения ресурсов контролирует потребление ресурсов глобальным заданием. Если задание начинает потреблять больше ресурсов, чем было заявлено в запросе или увеличивается загрузка узла за счет локальных процессов владельца, то посылается запрос системе управления заданием о приостановлении выполнения.
5. После завершения выполнения файлы с выходными данными задания передаются туда, куда указал пользователь в описании задания, затем все данные, связанные с заданием, удаляются с исполнительного узла.
6. Агент информирует сервер о том, что он свободен (в базе данных меняется статус узла) и готов к приему очередного задания.

Пользовательский интерфейс

Предоставляя доступ к вычислительным ресурсам грид-сегмента, пользовательский интерфейс дает пользователям грид механизм запуска заданий с возможностью описания самого задания и требуемых заданию ресурсов. Пользователю предоставляется возможность получать информацию о выполнении задания и количестве потребленных заданием ресурсов, а также управлять заданием. С помощью этого компонента пользователи грид (под пользователями здесь понимаются как обычные пользователи, так и различные приложения грид, осуществляющие диспетчеризацию и запуск заданий – брокеры ресурсов) по стандартным протоколам осуществляют взаимодействие с грид-сегментом. Таким образом, пользовательский интерфейс обеспечивает связь между существующими грид-инфраструктурами и сегментом грид с некластеризованными ресурсами.

В состав пользовательского интерфейса входят клиентские компоненты служб управляющего узла, которые реализуют необходимую функциональность. Пользователю предоставляется возможность описания своего задания (например, с помощью языка JDL – Job Description Language [10]), а также интерфейс мониторинга и управления заданием. В любой момент выполнения задания пользователь может узнать статус задания (запущено, или задание по каким-то причинам было сброшено, о чем агент сообщает на управляющий узел), узнать о количестве потребленных заданием ресурсов, а также снять свое задание. Пользователю предоставляется механизм безопасной передачи данных, с помощью которого он может отправить задание и необходимые для выполнения задания данные, расположенные как на его рабочем месте, так и указать другое место расположения необходимой информации.

5. Заключение

В работе рассмотрен способ построения грид из некластеризованных ресурсов. Произведено сравнение предлагаемого подхода с существующими решениями и предложены методы организации грид-сегментов из такого рода ресурсов, опирающиеся при взаимодействии компонентов на стандарты грид, что дает

возможность включать сегменты из некластеризованных ресурсов в общую грид-инфраструктуру.

Основные результаты работы состоят в следующем:

- предложен способ организации инфраструктуры сегмента грид с некластеризованными ресурсами;
- разработана архитектура ПО грид для некластеризованных ресурсов:
 - выделены ключевые компоненты и определена их функциональность;
 - сведены к минимуму серверные компоненты на исполнительном узле, что делает менеджер ресурсов небольшим по размеру и легким в настройке;
- разработаны методы взаимодействия компонентов системы:
 - функциональность управляющего узла реализована набором веб-служб.
 - задание передается на исполнительный компьютер по запросу менеджера ресурсов;

Литература

- [1]. В.Н. Коваленко, Д.А.Корягин Организация ресурсов в грид // Препринт № 63. Москва: ИПМ им. М.В.Келдыша РАН, 2004. 25 с.
- [2]. Globus Toolkit (<http://globus.org>)
- [3]. Платформа грид gLite (<http://www.glite.org>)
- [4]. Программная система VOINC (<http://boiunc.berkeley.edu>)
- [5]. Дэвид П. Андерсон Общественный компьютеринг: вовлечение людей в науку, <http://gridclub.ru/library/publication.2004-12-08.5823513687>
- [6]. Система пакетной обработки заданий PBS (<http://www.openpbs.org>)
- [7]. <http://www.platform.com/products/LSF>
- [8]. <http://www.cs.wisc.edu/condor/>
- [9]. В.Н. Коваленко, Е.И. Коваленко, Д.А.Корягин, Э.З. Любимский Метод опережающего планирования для грид // Препринт № 112. Москва: ИПМ им. М.В.Келдыша РАН, 2005. 33 с.
- [10]. Job Description Language HowTo, http://server11.infn.it/workload-grid/docs/DataGrid-01-TEN-0102-0_2-Document.doc