Инструментальный комплекс для организации гетерогенных распределенных вычислительных сред

И.В. Бычков, А.С. Корсуков, Г.А. Опарин, А.Г. Феоктистов

Аннотация. Рассматривается архитектура инструментального комплекса DISCENT, обеспечивающего поддержку основных этапов технологии построения и применения гетерогенных распределенных вычислительных сред для решения ресурсоемких научно-исследовательских задач.

Ключевые слова: инструментальные средства, Grid, моделирование потоков заданий.

Введение

В последнее время для решения ресурсоемких фундаментальных и прикладных задач создаются различные распределенные и параллельные вычислительные системы и среды (например, [1-7]). В частности, активно применяется технология создания специальной вычислительной сети, получившей название Grid [8]. Эта технология базируется на интеграции географически распределенных информационно-вычислительных и коммуникационных ресурсов и их совместном использовании в процессе вычислений. Инфраструктура Grid включает объединенные телекоммуникационной средой средства вычислений и обработки информации (суперкомпьютеры, вычислительные кластеры, отдельные персональные компьютеры, системы хранения данных и др.) и системное программное обеспечение (ПО), предназначенное для управления процессом выполнения заданий пользователей в этой среде.

Grid, организованная на базе кластеров, в большинстве случаев ее использования обеспечивает возможность удаленного доступа к ресурсам (узлам) вычислительной сети и позволяет определить вычислительные возможности конкретного узла (число процессоров, объем оперативной памяти и т.п.) и степень его работоспособности, а также выполнить на этом узле некоторое независимое задание или обработать один из блоков данных при решении «большой» задачи, допускающей распараллеливание по данным [9].

Однако существуют типы научно-исследовательских задач, для решения которых необходимы дополнительные возможности:

- обеспечение вычислительных услуг нетиражируемых программных комплексов, размещенных в узлах Grid;
- выполнение ряда взаимозависимых заданий, составляющих процесс решения одной общей задачи и требующих интеграции распределенных вычислительных ресурсов на основе автоматического планирования последовательности их использования.

Анализ систем управления заданиями (СУПЗ) для кластеров показывает, что как правило в этих системах реализовано централизованное управление автономными заданиями. Между тем, решение задач с большим числом подзадач и информационно-логических связей между ними в значительной степени увеличивает нагрузку на управляющий компьютер кластера и может привести к снижению работо-

способности всей вычислительной системы в целом [10]. Кроме того, формы ведения вычислительных работ в Grid, обусловленные используемым для ее организации сложным системным ПО, во многом ориентированы на специалистов с достаточно высоким уровнем квалификации в области системного программирования. Эти обстоятельства сдерживают широкое применение Grid специалистами-прикладниками и актуализируют вопросы создания «дружественных» средств доступа к ресурсам Grid и распределенных способов управления вычислительным процессом [11].

В работе предложен способ создания гетерогенных распределенных вычислительных сред (РВС)[1], предоставляющих широкий выбор средств для решения ресурсоемких научно-исследовательских задач различных типов и обеспечивающих возможность интеграции с другими РВС. Рассмотрен программный комплекс, предназначенный для инструментальной поддержки основных этапов построения и применения РВС.

Разработанное ПО предоставляет пользователям возможность самостоятельно (без участия высококвалифицированных системных программистов) описывать исследуемую предметную область и программно-аппаратную часть РВС, а также формировать задания для решения своих задач и проводить вычислительные эксперименты в РВС. Тем самым сокращаются сроки и повышается эффективность проведения сложных научных и прикладных задач.

1. Модель РВС

Предложенная авторами модель РВС [12] обеспечивает накопление знаний о вычислительных ресурсах, а также администрирование, планирование и применение этих ресурсов при решении пользовательских задач.

В рассматриваемой модели выделим следующие основные уровни:

– пользовательский уровень, включающий в себя: способы и средства доступа к вычислительным узлам РВС; множество пользователей РВС, их классификацию по категориям и правам доступа к РВС; множество заданий, запускаемых пользователями в РВС;

– уровень ПО, отражающий свойства и характеристики программных приложений, запуск которых требуется при выполнении заданий;

– аппаратный уровень, определяющий характеристики и комплектующие вычислительных узлов и коммутирующих устройств РВС;

– уровень планирования вычислений и загрузки ресурсов, обеспечивающий выбор необходимых ресурсов РВС для запуска заданий;

– вычислительный уровень, отражающий процесс выполнения заданий на конкретных вычислительных узлах РВС.

Разные уровни модели РВС содержат свои наборы объектов. Для каждого объекта определены его атрибуты. Особенностью этой структуры является то, что она позволяет на разных ее уровнях определять и совместно использовать различные модели распределенных вычислений, такие как модели программирования приложений, модели планирования вычислительных процессов, модели планирования загрузки ресурсов.

При формировании в модели РВС множества типов задач следует учитывать характеристики этих задач, важные в плане формирования и выполнения заданий по их решению. Для определения типов задач будем использовать следующую нотацию:

Type: <имя типа> (<имя атрибута> = <значение атрибута>; <имя атрибута> = <значение атрибута>; …).

Приведем имена и значения основных атрибутов:

– subTask – характеристика алгоритма решения задачи с точки зрения наличия подзадач; фиксированными значениями для subTask являются 1 (отсутствие подзадач) и K (наличие подзадач);

– algorithmConcurrency – характеристика алгоритма решения задач с точки зрения наличия в нем параллелизма; фиксированными значениями для algorithmConcurrency являются Coarse-grained (крупноблочный параллелизм алгоритма), Fine-grained (мелкозернистый параллелизм алгоритма) и Sequential (последовательный алгоритм); недопустимыми комбинациями значений subTask и algorithmConcurrency

[1] Под РВС следует понимать как отдельные вычислительные кластеры, так и Grid.

являются (subTask = 1; algorithmConcurrency = Coarse-grained; …) и (subTask = K; algorithmConcurrency = Fine-grained; …);

– multiCalculation – характеристика процесса решения задачи с точки зрения необходимости выполнения многовариантных расчетов; фиксированными значениями для multiCalculation являются 1 (один единственный вариант данных) и N (наличие множества вариантов данных);

– taskLocation – характеристика процесса решения задачи с точки зрения места размещения выполняемого программного приложения; фиксированными значениями для taskLocation являются Local (выполнение приложения, размещенного в узлах РВС) и Remote (выполнение удаленного приложения).

Вышеперечисленные атрибуты могут принимать значение All, что обозначает любое допустимое значение.

Определим основные типы задач для РВС:

– Type: Standard (subTask = 1; algorithmConcurrency ≠ Coarse-grained; multiCalculation = All; taskLocation = Remote) – стандартные задачи (скомпилированные программы пользователей), для выполнения которых используется штатный набор функций СУПЗ;

– Type: MultiVariant (subtask = All; algorithmConcurrency = All; multiCalculation =N; taskLocation = All) – многовариантные задачи, для решения которых необходимо выполнение программы с различными наборами данных;

– Type: Located (subTask = All; algorithmConcurrency = All; multiCalculation = All; taskLocation = Local) – задачи, в которых требуется выполнение приложений, размещенных в узлах РВС;

– Type: Interrelated (subTask = K; algorithmConcurrency = Coarse-grained; multiCalculation = All; taskLocation = All) – взаимосвязанные задачи, требующие выполнения набора взаимозависимых заданий, включенных в процесс решения одной общей задачи.

Для каждого уже определенного типа могут быть образованы подтипы:

<имя типа/подтипа> SubType: <имя подтипа> (<имя атрибута> = <значение параметра>; <имя атрибута> = <значение параметра>; …), где <имя атрибута> – это имя атрибута уже определенного типа, имеющее значение All.

Тип решаемой задачи определяет соответственно тип задания, т.е. задает формат паспорта этого задания и способ его обработки (режим запуска) в РВС.

2. Схема взаимодействия пользователя с узлами РВС

Рассмотрим схему взаимодействия пользователя с узлами РВС, представленными вычислительными кластерами. В рамках приведенной схемы в качестве интеллектуальной управляющей надстройки к СУПЗ, установленных в узлах РВС, используется программная система Web-Interface Manager (WIM). Схема взаимодействия пользователя с узлами РВС представлена на Рис. 1.

Запуск пользовательского задания и его выполнение на узлах вычислительного кластера происходит поэтапно. Прежде чем приступить к работе с кластером, пользователь должен пройти процедуру регистрации, если же у него имеется учетная запись, то необходимо пройти процедуру авторизации. Следующим этапом будет отображение веб-формы, с помощью которой пользователь сможет выбрать тип исполняемой задачи и необходимый режим запуска. Каждому режиму запуска задания соответствует определенная веб-форма, содержащая набор необходимых полей, на основе которых система WIM составляет паспорт задания. Такие поля обычно содержат информацию о запускаемой пользовательской программе, ее исходные данные и другую специфическую информацию, характерную для каждого режима запуска.

Таким образом, после заполнения пользователем полей в веб-форме информация о задании передается по протоколу HTTP на главный узел кластера, где установлен веб-сервер Apache. Система WIM получает поступившие данные от веб-сервера, обрабатывает их и составляет паспорт задания в формате СУПЗ. Подготовленный паспорт заданий, программа на выполнение и ее исходные данные передаются запускающей службе СУПЗ. Главная служба СУПЗ, основываясь на поставляемых службой сбора информации данных о загруженности кластера, производит выбор подходящего вычислительного узла (узлов), на котором будет выполняться задание пользователя.

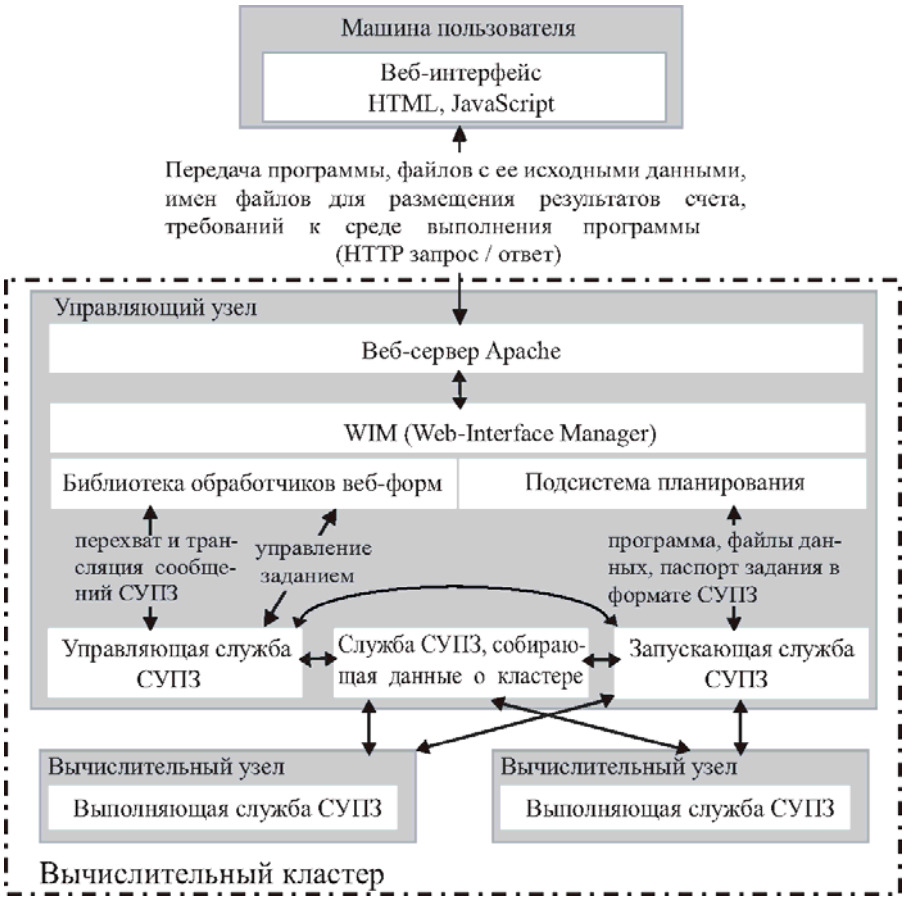

Рис. 1. Схема взаимодействия пользователя с узлами РВС

Выполнение задания обеспечивает выполняющая служба СУПЗ, установленная на каждом вычислительном узле кластера. В процессе выполнения задания пользователь имеет возможность совершать следующие операции: управлять заданием, просматривать ход его выполнения и просматривать отчет об ошибках. После того как задание выполнено, пользователь может скачивать файлы с результатами счета и просматривать историю выполнения задания. Операции, выполняемые пользователем в веб-интерфейсе, обрабатываются системой WIM и конвертируются в команды, понятные СУПЗ. Ответы на посланные команды перехватываются и транслируются системой WIM пользователю.

3. Архитектура инструментального комплекса

Инструментальный комплекс (ИК) DIStributed Computing ENvironment Toolkit (DISCENT) предназначен для организации и применения РВС. ИК DISCENT состоит из трех основных компонентов (Рис. 2): конструктора, базы данных и системы WIM.

Конструктор включает четыре подсистемы.

- Конструктор командного языка служит для создания, редактирования или удаления шаблонов команд для различных СУПЗ.
- Конструктор языка заданий используется для построения и модификации шаблонов паспортов заданий для различных СУПЗ.
- Конструктор веб-форм предназначен для создания шаблонов веб-форм, используемых для заполнения паспортов заданий. Создание веб-формы паспорта задания выполняется путем размещения на форме и задания свойств графических элементов HTML, соответствующих параметрам формируемого паспорта задания.
- Конструктор модели РВС применяется для описания и модификации данных о вычис-

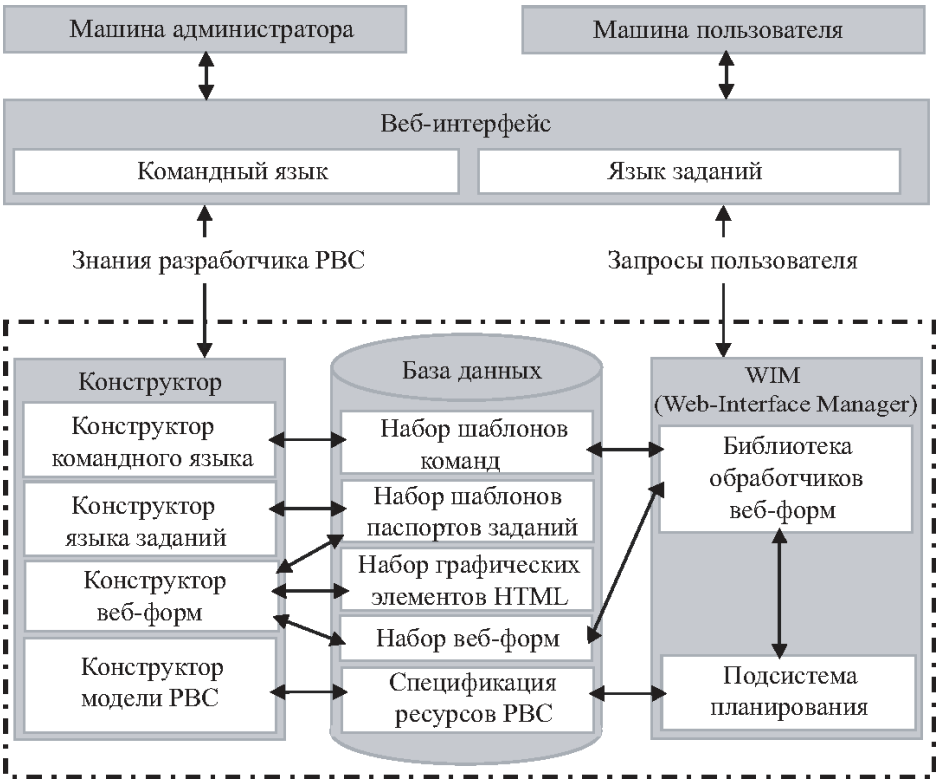

Рис. 2. Архитектура ИК DISCENT

лительных ресурсах РВС. Такая информация необходима подсистеме планирования для эффективного распределения различных типов задач по вычислительным узлам РВС.

Язык заданий определяет взаимодействие пользовательских задач с системой WIM и предназначен для формирования паспорта задания, включающего в себя уникальное имя задания, тип решаемой задачи, исполняемые программы, исходные данные, минимальные и/или желаемые требования к вычислительным ресурсам РВС и др.

Командный язык представляет собой совокупность команд, предназначенных для управления пользовательскими заданиями в узлах РВС. С помощью командного языка система WIM может быть легко и гибко настроена для взаимодействия с используемой СУПЗ за счет применения унифицированных шаблонов описания команд СУПЗ.

Основным назначением базы данных является накопление информации о вычислительных ресурсах РВС, средствах создания и обработки паспортов заданий. Эта информация используется планировщиками системы WIM при формировании потоков заданий. Содержимое базы данных включает наборы шаблонов для командного языка и языка заданий, наборы графических элементов HTML и веб-форм, спецификацию ресурсов РВС.

Система WIM обеспечивает выполнение заданий пользователя путем распределения указанных в заданиях приложений на узлы РВС, подходящие для их запуска. Основными составляющими системы являются библиотека обработчиков веб-форм и подсистема планирования.

Библиотека обработчиков веб-форм предназначена для выполнения следующих функций: вывода пользователям требуемых веб-форм; обработки поступивших из этих форм данных; формирования паспортов заданий; предоставления пользователю информации о состоянии заданий и результатах счета; добавления, удаления и приостановления заданий пользователей. Данная библиотека состоит из набора скриптов языка PHP (обработчиков веб-форм), каждый из которых реализует алгоритмы для работы с одной или несколькими веб-формами на стороне

веб-сервера. Имя обработчика веб-формы содержится на ней в специальном скрытом поле. Обработчики веб-форм имеют доступ к шаблонам команд, хранящимся в базе данных. В шаблонах команд для каждой операции управления заданием, размещенной на веб-форме, определяется команда СУПЗ, с помощью которой данная операция будет выполнена в РВС.

Подсистема планирования служит для обработки, постановки в очередь и распределения заданий, соответствующих различным типам задач. Данная подсистема включает три планировщика.

- Планировщик стандартных заданий MST (Manager of Standard Tasks) используется для распределения заданий типа Standard.
- Планировщик распределенных приложений MDA (Manager of Distributed Applications) предназначен для отправки заданий на вычислительные узлы РВС, на которых установлены приложения, необходимые для выполнения заданий типа Located. Планировщик MDA взаимодействует с планировщиком MST с целью резервирования за заданием выбранных ресурсов перед передачей этого задания обработчику веб-форм для запуска задания.
- Планировщик взаимосвязанных заданий MIT (Manager of Interrelated Tasks) ориентирован на распределение заданий типа Interrelated и контроль процесса их выполнения. Планировщик MIT, используя информационно-логические связи между объектами модели РВС, выполняет частичное упорядочение подзадач общего задания, находит необходимые для выполнения задания вычислительные ресурсы, взаимодействует с планировщиком MST с целью резервирования за заданием выбранных ресурсов, передает задание обработчику веб-форм для его запуска и осуществляет дальнейший контроль выполнения этого задания.

Как правило, резервирование ресурсов увеличивает время ожидания заданий в очереди [13]. Однако без применения такого способа время ожидания возрастает в значительно большей степени для заданий типов Located и Interrelated.

При распределении заданий по узлам РВС планировщики системы WIM используют спецификацию ресурсов РВС из базы данных.

4. Технология построения гетерогенных РВС

При организации РВС с помощью ИК DISCENT (Рис. 3) выделим три основных этапа:

– формирование программно-аппаратной части РВС;

– настройка веб-интерфейса;

– реализация процесса решения пользовательских задач.

На Рис. 3 указаны подсистемы ИК DISCENT, используемые для реализации описанных ниже подэтапов.

I. *Определение вычислительных узлов РВС*, которые будут задействованы в качестве ее ресурсов для выполнения заданий пользователей. Помимо этого необходимо выделение рабочей станции (или машины) с выходом в Интернет, которая будет выполнять роль веб-сервера.

II. *Установка и настройка ПО*. На данном подэтапе осуществляется инсталляция системного ПО, необходимого для функционирования ИК DISCENT. В частности, необходима установка и настройка следующих пакетов: веб-сервер Apache версии 2.0.43 (или выше), интерпретатор языка программирования для веб PHP версии 5 (или выше). Затем выполняется установка и конфигурация ИК DISCENT, включающая копирование исходных файлов данного ИК на веб-сервер РВС и задание значений переменных окружения (например, путь к СУПЗ).

III. *Построение модели РВС*. Включает в себя формирование множества пользователей РВС, заинтересованных в использовании вычислительных ресурсов РВС; классификацию пользователей по видам или категориям (например, администраторы, операторы, преподаватели, студенты, сторонние пользователи и др.); наделение их соответствующими правами использования ресурсов РВС; составление спецификации вычислительных ресурсов РВС; определение основных типов задач, которые будут решаться в РВС.

IV. *Параметризация и настройка шаблонов команд для каждой СУПЗ, используемой в РВС.* На данном подэтапе создаются шаблоны команд для обеспечения систем взаимодействия WIM с различными СУПЗ.

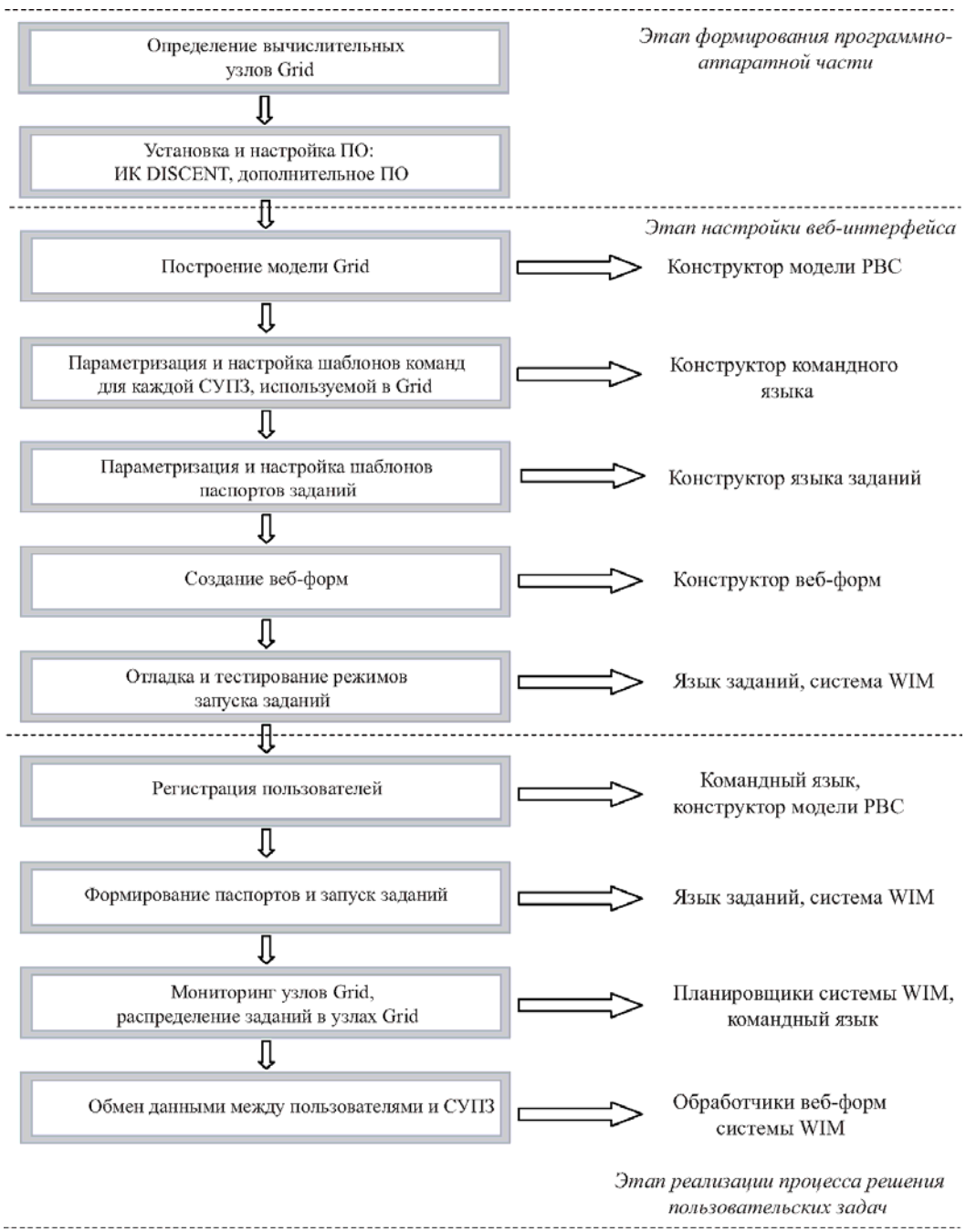

Рис. 3. Этапы организации РВС

V. *Параметризация и настройка шаблонов паспортов заданий*. Для каждого типа пользовательского задания, которое будет обрабатываться определенной СУПЗ, необходимо создать шаблон паспорта задания, который должен учитывать специфику запуска выбранного типа заданий и особенностей СУПЗ.

VI. *Создание веб-форм*. Производится администратором РВС с помощью конструкторов, входящих в состав ИК DISCENT и позволяющих в графическом интерактивном режиме формировать необходимые поля (текстовые поля, выпадающие меню и т.д.) на веб-формах, которые будут выполнять запуск определенных типов заданий.

VII. *Отладка и тестирование режимов запуска пользовательских заданий*. На этом подэтапе происходит прогон специально подготов-

ленных тестовых примеров для каждого режима запуска заданий, определяется их работоспособность и отказоустойчивость. В случае появления сбоев в работе системы WIM происходит поиск возникших ошибок, выявление и устранение их причин.

VIII. *Регистрация пользователей*. Включает создание учетных записей пользователей, которые будут использовать ресурсы РВС.

IX. *Формирование паспортов и запуск заданий.* На данном подэтапе для осуществления запуска задания пользователи должны выбрать тип запускаемого задания и заполнить все необходимые поля в веб-форме, предложенной системой WIM.

X. *Мониторинг узлов РВС, распределение заданий в узлах РВС.* Этот подэтап выполняют планировщики системы WIM. С помощью командного языка планировщики различных типов заданий могут получать информацию у СУПЗ о состоянии загруженности узлов РВС. Спецификацию вычислительных ресурсов планировщики получают из базы данных.

XI. *Обмен данными между пользователями и СУПЗ.* Осуществляется обработчиками веб-форм, которые получают данные (например, программу на выполнения, ее исходные данные и др.) с машины пользователя с помощью метода «upload». Данные же с СУПЗ (например, файл с результатами счета, файл отчета СУПЗ) обработчики получают из определенной директории, место расположения которой указывается в паспорте задания для СУПЗ. После того, как задание выполнено, обработчики веб-форм помещают результаты счета в уникальный веб-каталог и высвечивают пользователю гиперссылку, по которой эти данные могут быть сохранены на жесткий диск.

5. Вычислительный эксперимент

ИК DISCENT был использован для организации экспериментальной Grid ИДСТУ СО РАН и решения в ней целого ряда практических задач [12]. С целью более полного и детального анализа эффективности функционирования Grid, в том числе и разработанных средств управления заданиями, был проведен вычислительный эксперимент по моделированию процессов формирования и обработки непрерывных потоков заданий в течение длительного периода времени.

Grid представляет собой стохастическую динамическую вычислительную сеть, состояние которой в некоторый момент времени определяется распределением в ней потоков заданий [14]. Для получения оценки эффективности различных способов распределения потоков заданий (E, %) генерируется поток заданий, осуществляется их распределение и выполнение в Grid двумя способами (с помощью GridWay и GridWay + WIM) и сравниваются следующие показатели эффективности функционирования вычислительных ресурсов Grid: число заданий с нулевым временем ожидания (T_0, сек.), среднее время ожидания задания в очереди (T_q, сек.), среднее число заданий в очереди (C_q, ед.), среднее время пребывания задания в Grid (T_g, сек.), коэффициент полезного использования ресурсов Grid (K_g, %), количество заданий в потоке (C_s, ед.), общее время решения заданий потока (T_s, сек.).

Поток заданий в Grid характеризуется следующими свойствами: неоднородностью (задания соответствуют разным типам задач и отличаются друг от друга по своей специфике); отсутствием обратной связи (число заданий, поступивших за один промежуток времени, не зависит от числа заданий, поступивших за другой промежуток времени); неординарностью (возможно поступление двух и более заданий в один и тот же момент времени); стационарностью (число событий, поступивших за определенный промежуток времени, зависит от длины этого промежутка и не зависит от момента его начала).

Для формирования потока заданий и отправки их на выполнение в Grid ИДСТУ СО РАН разработан специальный генератор. В качестве приложений в заданиях используются программы-имитаторы, выполняющие в Grid реальную загрузку вычислительных ресурсов и обмен заданными объемами данных. Генератор заданий был запущен на удаленной машине в Институте вычислительного моделирования СО РАН, которая играла роль сторонней Grid, а также на независимых рабочих станциях в Интернет, представляющих машины пользователей веб-интерфейса к Grid ИДСТУ СО РАН.

Результаты вычислительных экспериментов

Grid	2 кластера с ОС Linux (168 выч. ядер)			2 кластера с ОС Linux, кластер с ОС Windows (184 выч. ядер)			2 кластера с ОС Linux, кластер с ОС Windows (184 выч. ядер)		
Типы задач	Все типы			Все типы			Некоторые типы*		
	GridWay	GridWay+ WIM	*E* (%)	GridWay	GridWay+ WIM	*E* (%)	GridWay	GridWay+ WIM	*E* (%)
T_0	49060	52620	7,26%	24340	26680	9,61%	17980	19920	10,79%
T_q	87260	80100	–8,21%	44020	39740	–9,72%	28240	25540	–9,56%
C_q	7360	7260	–1,36%	3760	3620	–3,72%	2380	2280	–4,20%
T_g	123920	118440	–4,42%	62580	58460	–6,58%	39220	35420	–9,69%
K_g	93,34%	94,13%	0,85%	92,41%	95,12%	2,93%	93,74%	94,97%	1,31%
C_s	18688	18688	–	9344	9344	–	157	157	–
T_s	4917080	4716820	–4,07%	2486420	2336520	–6,03%	1585160	1459720	–7,91%

* Потоки формировались из задач следующих двух подтипов: *Standard MultiVariant:* (*multiCalculation* = *N*) и *Interrelated Remote*: (*taskLocation* = *Remote*)

Результаты вычислительных экспериментов показывают устойчивое преимущество второго способа распределения заданий (GridWay+WIM) по всем показателям эффективности функционирования ресурсов Grid, что видно из приведенной таблицы.

Заключение

Разработан способ организации распределенных вычислений, отличающийся от известных обеспечением таких дополнительных возможностей, как формирование потоков заданий в зависимости от типов решаемых задач и распределенное управление этими потоками в процессе выполнения заданий. В основе этого способа лежат специализированные языковые и инструментальные средства, предназначенные для быстрой и гибкой настройки веб-ориентированного доступа к ресурсам гетерогенной РВС.

Литература

1. Бурцев В. Параллелизм вычислительных процессов и развитие архитектуры суперЭВМ / В. Бурцев. – М.: ИВВС РАН, 1997. – 152 с.
2. Воеводин В. Параллельные вычисления / В. Воеводин, Вл. Воеводин. – СПб.: БХВ-Петербург, 2002. – 608 с.
3. Емельянов С.В. Реализация Grid-вычислений в среде IARnet / С.В. Емельянов, А.П. Афанасьев, В.В. Волошинов, Я.Р. Гринберг, В.Е. Кривцов, О.В. Сухорослов // Информационные технологии и вычислительные системы. – М.: Институт микропроцессорных вычислительных систем РАН, 2005, №2, С.61-75.
4. Коваленко В.Н. Организация ресурсов ГРИД / В.Н. Коваленко, Д.А. Корягин. – М., 2004. – 25 с. – (Препринт / ИПМ им. Келдыша РАН; № 63).
5. Корнеев В. Параллельные вычислительные системы / В. Корнеев. – М.: Нолидж, 1999. – 320 с.
6. Лацис А.О. Как построить и использовать суперкомпьютер / А.О. Лацис. – М.: Бестселлер, 2003. – 274 с.
7. Опарин Г.А. Инструментальная распределенная вычислительная САТУРН-среда / Г.А. Опарин, А.Г. Феоктистов // Программные продукты и системы. – 2002. – №2. – С. 27-30.
8. Foster I. The Anatomy of the Grid: Enabling Scalable Virtual Organizations / I. Foster, C. Kesselman, S. Tuecke // Intern. J. of High Performance Computing Applications. – 2001. – Vol. 15, № 3. – P. 200-222.
9. Воеводин В.В. Решение больших задач в распределенных вычислительных средах / В.В. Воеводин // Автоматика и телемеханика. – 2007. – № 5. – С. 32-45.
10. Каляев И.А. Распределенные системы планирования действий коллективов роботов / И.А. Каляев, А.Р. Гайдук, С.Г. Капустян. – М.: Янус-К, 2002. – 292 с.
11. Durfee E.H. Distributed problem solving and planning / E.H. Durfee // Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence / Ed. by G. Weiss. – Cambridge: MIT Press, 1999. – P. 121-164.
12. Феоктистов А.Г. Разработка Grid-системы с децентрализированным управлением потоками заданий / А.Г. Феоктистов, А.С. Корсуков // Вестник НГУ. Серия: Информационные технологии. – 2008. – Т. 6, вып. 3. – С. 147-154.
13. Топорков В.В. Модели распределенных вычислений / В.В. Топорков. – М.: ФИЗМАТЛИТ, 2004. – 320 с.
14. Попков Ю.С. Макросистемы и GRID-технологии: моделирование динамических стохастических сетей / Ю.С. Попков // Проблемы управления. – 2003. – № 8. – С. 10-20.

Бычков Игорь Вячеславович. Директор Института динамики систем и теории управления СО РАН. Окончил Иркутский государственный университет в 1983 году. Член-корреспондент, доктор технических наук, профессор. Автор 100 печатных работ, 8 монографий. Область научных интересов: ГИС, Grid и веб-технологии. E-mail: idstu@icc.ru.

Корсуков Александр Сергеевич. Младший научный сотрудник Института динамики систем и теории управления СО РАН. Окончил Иркутскую государственную экономическую академию в 2004 году. Кандидат технических наук. Автор 15 печатных работ. Область научных интересов: инструментальные средства организации распределенных вычислений, Grid, веб-технологии. E-mail: alexask@mail.ru

Опарин Геннадий Анатольевич. Заместитель директора Института динамики систем и теории управления СО РАН. Окончил Казанский авиационный институт им. А.Н. Туполева в 1974 году. Доктор технических наук, профессор. Автор 180 печатных работ. Область научных интересов: модели и методы организации распределенных вычислений, Grid. E-mail: oparin@icc.ru.

Феоктистов Александр Геннадьевич. Старший научный сотрудник Института динамики систем и теории управления СО РАН. Окончил Иркутский государственный университет в 1987году.
Канд. тех. наук, доцент. Автор 62 печатных работ. Области научных интересов: методы и инструментальные средства организации распределенных вычислений, Grid. E-mail: agf@icc.ru.