УДК 681.324

М.А. Волк

Харьковский национальный университет радиоэлектроники

СТРУКТУРНАЯ ОРГАНИЗАЦИЯ ПОВЕДЕНЧЕСКОГО ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ В GRID

Целью статьи является описание состава и функциональности элементов имитационной системы моделирования, предназначенной для работы в GRID-инфраструктуре.

имитационное моделирование, распределенные системы, GRID-инфраструктура

Введение

Использование глобальных распределенных вычислителей открывает перспективы построения имитационных моделей систем, требующих больших ресурсов. Одной из наиболее перспективных инфраструктур, предоставляющих подобные возможности, является GRID.

GRID является следующим поколением параллельного и распределенного компьютинга, объединяющим различные гетерогенные ресурсы для решения задач большой размерности в науке, инженерии и коммерции [1]. Сегодня многие пакеты программ, рассчитанные на параллельные вычисления, адаптируются для работы в GRID. То же самое касается и систем моделирования, например, Bricks[2], OptorSim [3], SimGrid [4].

Адаптация существующих систем приводит к ограничению их возможностей, так как программирование симуляторов выполняется в виде переделки существующего программного обеспечения (ПО). Данный подход не дает естественной интеграции системы моделирования в GRID - пространство.

В связи с этим, необходимо пересмотреть саму структуру моделирующих программ с целью оптимального включения в инфраструктуру GRID, тем самым создав своего рода middleware систем имитационного моделирования.

Вычислительное и поведенческое моделирование

Исходя из того факта, что первоначально в истории развития теории моделирования преобладали математические модели, очень часто системы моделирования представляют собой пакеты решения сложных математических задач, которые предполагают формальные методы распараллеливания.

Для таких моделей в большинстве случаев дополнительных моделирующих программ не требуется, а достаточно наличия интерфейса параллельного программирования, в качестве которого наиболее широко используется MPI.

Когда говорят об имитационном моделировании, имеют ввиду проведение вычислительного эксперимента на ЭВМ [5]. При этом результатами проведения эксперимента могут быть статические, динамические характеристики моделируемой системы. Имитационное моделирование может быть проведено двумя способами:

1) простое решение математической задачи алгоритмическими методами (фактически мы приходим к способу, описанному выше);

2) построение алгоритмического решения, отражающего поведение системы (которое не всегда можно представить в виде математической модели).

Последнее решение чаще всего связано с поведенческим моделированием.

Поведенческое моделирование призвано отразить поведение модели во времени под воздействием внешних и внутренних факторов. В основе поведенческой имитационной модели лежит компьютерная программа, алгоритм которой соответствует алгоритму поведения некоторого объекта. Программа постоянно получает во времени вектор входных воздействий, отрабатывает логику поведения и постоянно выдает, с некоторой задержкой реального времени, выходные значения. Система моделирования следит за функционированием программы-модели, при необходимости производя согласование между частными имитационными моделями, пользователем или реальным устройством. Как частный случай, алгоритм может быть основан на математической модели, которая окружена программным интерфейсом.

Исполнение поведенческой модели аналогично исполнению процесса в многозадачной среде: процессорное время выделяется программе периодически, в точках синхронизации происходит обмен данными с другими программами. В отличии от вычислительной задачи, в которой происходит "отработка" модели от начала и до конца для получения результата, поведенческая модель постоянно находится в режиме выполнения.

В данной статье будут обсуждаться системы моделирования, поддерживающие поведенческое имитационное моделирование в GRID-инфраструктуре.

Основные понятия

Система имитационного моделирования состоит из двух основных модулей: *моделирующей программы* и непосредственно *имитационной модели* (моделей). Каждый из модулей, с точки зрения операционной системы, может являться параллельно исполняемым *процессом*.

Моделирующая программа получает от пользователя *задание* на моделирование. *Пользователем* может выступать человек, программа, программно-аппаратная система и т.п. Задание на моделирование включает в себя имитационную модель объекта моделирования и параметры проведения эксперимента.

Имитационная модель, как правило, декомпозируется на *частные* модели, которые могут исполняться параллельно.

Во время проведения вычислительного эксперимента задача моделирующей программы - организовать синхронное исполнение частных моделей и согласование результатов (промежуточных и окончательных) с пользователем.

Применение GRID для вычислительного эксперимента накладывает дополнительные требования на моделирующую программу.

Во-первых, частные модели исполняются на разных ресурсах.

Во-вторых, синхронизация частных моделей происходит через программное обеспечение GRID.

В-третьих, вычислительная структура ресурсов, на которых производиться моделирование, может изменяться от эксперимента к эксперименту.

При использовании традиционной схемы моделирования [6], моделирующая программа берет на себя функции взаимодействия с пользователем. В среде GRID эти функции должна взять на себя отдельная программа, запущенная на вычислительном ресурсе пользователя – *управляющая программа*. Объединение функций возможно в рамках одной программы, однако, в этом случае, часть функций не будут задействованы во время конкретного запуска системы.

Исходя из вышеизложенного, сформулируем требования, налагаемые на систему имитационного моделирования в GRID.

1. Система имитационного моделирования должна взаимодействовать с программным обеспечением GRID. Последнее получило название *middleware*.

2. Моделирующая программа и частные модели должны быть платформенно независимыми в том смысле, в котором неизвестен заранее набор исполнительных ресурсов.

3. Структура системы моделирования должна учитывать специфику распределенного выполнения.

4. Моделирующая программа должна следить за изменением конфигурации вычислительных ресурсов. Исполнение данного требования в большинстве случаев можно переложить на middlewar*e*.

5. Наличие двух "сервисов" системы моделирования: моделирующей и управляющей программы.

В следующем разделе статьи предложенные требования получат свою реализацию в структурах систем имитационного моделирования для GRID.

Структуры системы моделирования

В данном разделе будет предложено несколько вариантов организации структуры распределенной имитационной системы поведенческого моделирования, ориентированных на использования в GRID.

Первая структура, которую назовем *универсальной*, может использоваться для любых классов имитационных моделей и состава вычислительных ресурсов (рис. 1).

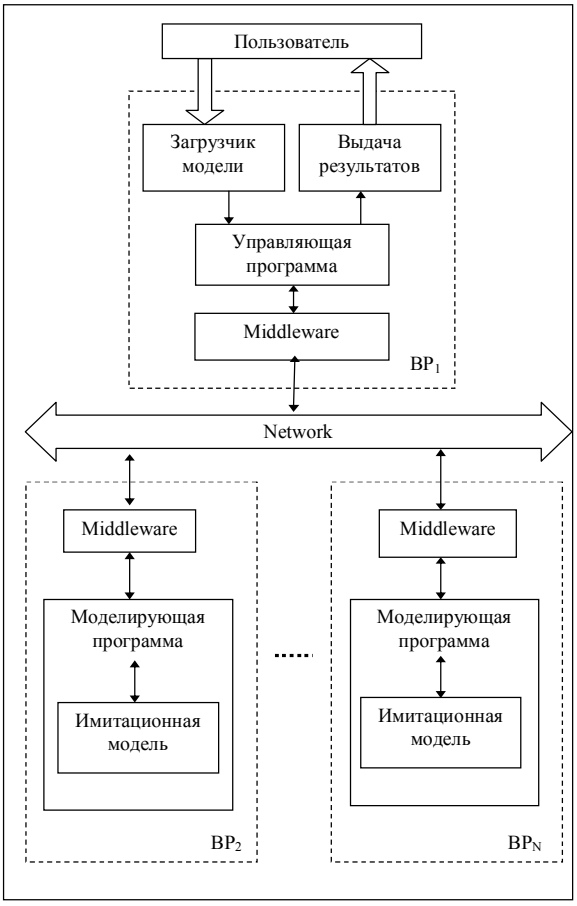

Рис. 1. Универсальная структура

В универсальной структуре явно выделены управляющая и моделирующая программы. Имитационные модели являются органически включенными в оболочку моделирующей программы. Взаимодействие между элементами системы моделирования, расположенными на разных вычислительных ресурсах (ВР), выполняется средствами middlewar*e*. Понятие middlewar*e* широко используется в GRID для обозначения сервисов, реализующих безопасное и надежное управление вычислительными ресурсами виртуальной сети. При достаточно высокой степени стандартизации программного обеспечения системы моделирования управляющие и моделирующие программы сами могут стать честью middlewar*e*. Последнее свойство подкреплено еще тем, что наиболее распространенное программное обеспечение middlewar*e* (gLite и Nurdu Grid) является открытым и исполняется в открытой операционной системе Linux.

Данная структура будет работать как в локальной сети ЭВМ и на территориально распределенных компьютерах, так и на удаленных кластерах. Еще одним достоинством такой структуры является то, что написание самих имитационных моделей при соблюдении стандартного интерфейса частных с системой моделирования не требует от разработчика знаний о распараллеливании процесса моделирования.

Самый большой недостаток в данном случае – низкая скорость обмена данными между участниками вычислительного эксперимента. И в том случае, если время проведения цикла выполнения частных моделей превышает время передачи данных в среде GRID, применение распараллеливания задачи стоит под вопросом.

Универсальная структура с наибольшей эффективностью может быть применена в тех случаях, когда для исполнения частных моделей требуются значительные временные ресурсы, а связь частных моделей между собой является слабой.

В том случае, если частные модели имеют небольшую вычислительную сложность, а их взаимосвязи значительны, необходимо стремиться к минимизации времени работы моделирующей программы и middlewar*e*.

Идеальным выходом, в этом случае, является исключение из программного обеспечения, функционирующего на удаленных ресурсах BP_1 – BP_N, моделирующей программы. Однако, это значит, что реализация функций взаимодействия с middlewar*e*, управляющей программой и другими частными моделями ложиться на саму имитационную модель.

Такой подход реализуем, но требует больших затрат со стороны разработчиков имитационной модели. Действительно, кроме своей предметной области, такой разработчик должен знать правила работы с middlewar*e* GRID, а сама модель становиться "привязанной" к GRID-платформе. Выходом из подобной ситуации может быть создание некоторого контейнера для частной модели, который является стандартным для системы моделирования и обладает стандартным интерфейсом как с частной моделью, так и с управляющей программой.

С точки зрения объектно-ориентированного программирования это можно реализовать следующим образом: создать базовый класс ("интерфейс"), от которого будут порождаться все частные модели. Данный класс будет содержать стандартные функции обмена информацией с моделирующей программой, другими частными моделями и т.п.

Другое решение этой же проблемы – внутри частной модели использовать один из стандартных интерфейсов параллельного программирования, например, MPI.

Подобное решение приводит нас к другой структуре системы моделирования (рис. 2). Наиболее эффективно применение такой архитектуры на гомогенных кластерах, поэтому такую структуру можно назвать *кластерной*.

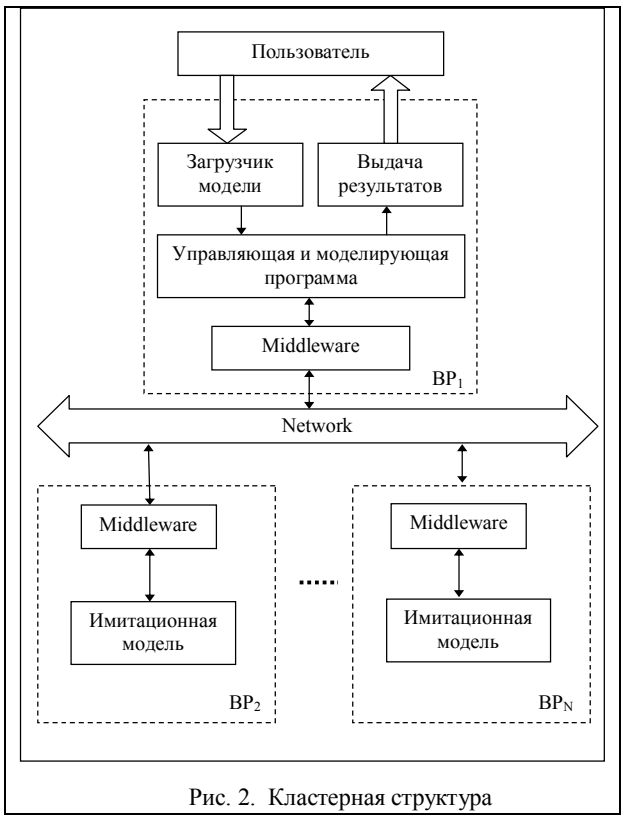

Рис. 2. Кластерная структура

В настоящее время существуют имитационные модели большой вычислительной сложности и обладающие неоднородной структурой. Такие модели относятся к моделям HLA (High Level Architecture), стандарты которых приняты IEEE (Institute of Electrical and Electronics Engineers) в 2000 году.

Распараллеливание разнородных имитационных моделей является более сложным процессом и должно учитывать множество факторов. К ним можно отнести: природу частных моделей (например, дискретные и аналоговые), способы организации моделей (процессный, событийный, транзактный, просмотра активностей), разные механизмы управления модельным временем [7].

Следующая предлагаемая структура учитывает ряд значимых составляющих имитационной модели:

1. Вычислительная сложность частных моделей может быть разной. В связи с этим на отдельном вычислительном ресурсе могут исполняться несколько частных моделей небольшой сложности и обладающих высокой степенью связности.

2. Для реализации п.1 в управляющую программу системы моделирования вводится дополнительный модуль: диспетчер частных моделей, - осуществляющий оценку взаимосвязей моделей и их вычислительную сложность.

3. Для каждого рода частных моделей необходимо наличие своей моделирующей программы. В случае, если моделирующие программы для частных моделей реализованы с использованием разных способов организации имитационного моделирования [5], необходимо дополнительное программное обеспечение, согласовывающее их взаимодействие.

4. Для взаимодействия моделирующих программ, указанных в п.3, необходимо создание middleware системы моделирования, согласовывающего работу разнородных моделирующих программ.

5. Для реализации п.4 необходимо создание интерфейсов и протоколов взаимодействия, учитывающих специфику имитационного моделирования.

6. Управление отдельными вычислительными ресурсами может осуществляться автоматически (программными или аппаратными средствами) или вручную (пользователями).

В общем случае полное программное обеспечение, установленное на одном вычислительном ресурсе может иметь структуру, приведенную на рис. 3.

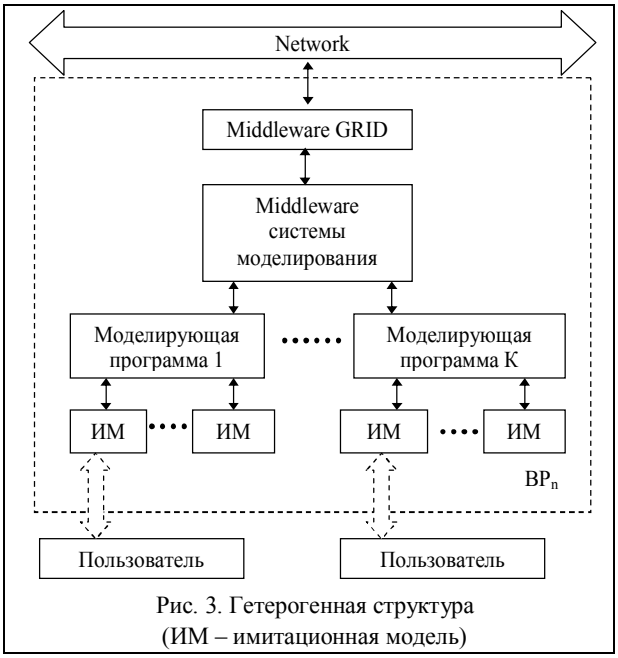

Рис. 3. Гетерогенная структура
(ИМ – имитационная модель)

В гетерогенной системе моделирования задания на моделирование могут корректироваться пользователями во время вычислительного эксперимента. Отдельные вычислительные ресурсы могут содержать одну имитационную модель и не содержать управляющих программ. Состав и характеристики ресурсов в GRID могут динамически изменяться. Организация взаимодействия подсистем моделирования в таких условиях является сложной технической задачей, работа над решением которой ведется в настоящее время.

В заключении раздела, приведем несколько примеров наиболее вероятных заказчиков гетерогенной системы моделирования в GRID.

Во-первых, разработчики сложных энергетических систем, включающих элементы производства, транспортировки и потребления энергии, системы управления, охлаждения, очистки т.д.

Во-вторых, разработчики систем нефте- и газопроизводства, транспортировки и переработки.

В-третьих, военные – для имитации военных действий с учетом взаимодействия тренажеров пер-

сонала, географических и погодных моделей, моделей противника и т.п. [8]

Кроме приведенных выше приложений, огромное число задач, способных решаться в GRID, присутствует в таких наукоемких областях, как биология, метеорология, медицина, химия, электроника и т.д.

Выводы

В настоящее время стандарты построения систем моделирования в GRID, как и сама GRID-инфраструктура, находятся в стадии формирования. В данной статье предложены различные структуры построения имитационных поведенческих систем моделирования, а также оценены области их возможного применения. В качестве дальнейших исследований необходимо рассмотрение состава и функциональности конкретных реализаций моделирующих и управляющих программ, разработка протоколов обмена данными между элементами предложенных структур.

Список литературы

1. Петренко А.І. GRID технології в науці і освіті // Матеріали 9-ої Міжнародної конференції «Системний аналіз та інформаційні технології». – К., 2007. – С. 138-140.

3. Aida K., Takefusa A., Nakada H., Matsuoka S., Sekiguchi S., Nagashima U. Performance Evaluation Model for Scheduling in a Global Computing System // Int. J. of High Performance Computing Applications. – 2000. – 14 (3). – P. 268-279.

3. Bell W.H., Cameron D.G., Capozza L., Millar A.P., Stockinger K., Zini F. OptorSim. A Grid Simulator for Studying Dynamic Data Replication Strategies // Int. J. of High Performance Computing Applications. – 2003. – 17 (4). – 403 p.

4. Casanova H. Simgrid. A Toolkit for the Simulation of Application Scheduling // Proc. of the First IEEE/ACM Int. Symposium on Cluster Computing and the Grid, Brisbane, Australia. – 2001. – P. 137-143.

5. Максимей И.В. Имитационное моделирование на ЭВМ. – М.: Радио и связь, 1988. – 232 с.

6. Горбачев В.А., Волк М.А., Бабаев А.П. Организация эффективного моделирования сложных систем // АСУ и приборы автоматики. – 1997. – Вып. 104. – С. 61-69.

7. Окольнишников В.В. Представление времени в имитационном моделировании // Вычислительные технологии. – 2005. – Т. 10, № 5. – С. 57-80.

8. Mille D.C., Thorpe J.A. SIMNET: the advent of simulator networking // Proc. of the IEEE. 1995. – Vol. 83, N 8. – P. 1114-1123.

Поступила в редколлегию 30.10.2007

Рецензент: д-р техн. наук, проф. С.Г. Удовенко, Харьковский национальный университет радиоэлектроники, Харьков.