УДК 52(15).003; 681.61

ПОДХОДЫ И СРЕДСТВА МОДЕЛИРОВАНИЯ GRID-СИСТЕМ ОБРАБОТКИ СПУТНИКОВЫХ ДАННЫХ

А.Ю. Шелестов

Институт космических исследований НАН и НКА Украины,
03680, проспект Академика Глушкова, 40.
Тел.: (+380-44) 526 2553; inform@ikd.kiev.ua

Рассматриваются вопросы построения распределенных программных систем, в частности, Grid-систем обработки данных наблюдения Земли. При этом особое внимание уделяется анализу предметной области, а также подходам к анализу таких систем. Для исследования важных свойств Grid-систем обработки спутниковых данных предлагается использовать аппарат сетей Петри. Для автоматизации этого процесса предлагается использовать специализированную систему визуального имитационного моделирования.

Issues of the building of distributed systems, particularly Earth observation Grid systems, are considered. Major attention is attending to the domain analysis and approaches to the analysis of these systems. For the investigations of the most important properties of such systems using of the Petri networks is proposed. For the automation of this process the specialized visual modeling system is suggested,

Стремительное усложнение инфраструктуры распределенных информационно-аналитических систем, которое наблюдается в настоящее время, привело к необходимости развития подходов, методов и средств моделирования таких систем. Системный подход к моделированию предложен в [1], а результаты исследований на основе его практического применения описаны в [2]. Сейчас одной из наиболее активно развиваемых технологий построения распределенных систем является Grid-технология, которая положена в основу работы виртуальных организаций, решающих вычислительно сложные задачи с привлечением распределенных хранилищ данных и мощных вычислительных ресурсов [3].

Для моделирования подобных распределенных систем используются различные подходы, ни один из которых не может претендовать на исчерпывающее описание, а позволяет описать какой-либо аспект функционирования или структуры системы. Достаточно полный обзор существующих моделей приведен в [4, 5]. Один из наиболее информативных способов моделирования программных систем связан с применением математического аппарата сетей Петри [6], поскольку Grid-система представляет собой набор взаимодействующих между собой компонентов (вычислительных узлов и хранилищ данных), которые могут функционировать параллельно и в работе которых должна быть обеспечена синхронизация. Особое значение такие модели приобретают при исследовании Grid-систем наблюдения Земли, поскольку в этих системах особое внимание уделяется синхронизации доступа к общим сегментам данных и распараллеливанию вычислений.

В данной работе описан один из подходов к моделированию динамики Grid-систем на основе сетей Петри, который позволяет исследовать ряд важных свойств распределенных приложений, в частности основанных на Grid-технологии. Кроме того, описывается визуальное средство, позволяющее существенно снизить требуемые для проведения этих исследований временны*е* затраты.

Описание предметной области и формулировка требований к модели

Распределенные системы, связанные с обработкой данных наблюдения Земли, характеризуются несколькими важными особенностями. В частности, для таких систем характерно использование как распределенных вычислительных, так и информационных ресурсов (совместно используемых баз данных или хранилищ). Как правило, прикладные задачи, решаемые в системах обработки спутниковых данных (в том числе, Grid-системах), предполагают обмен данными большого объема, использование информации из различных географически удаленных друг от друга источников, необходимость синхронизации потоков выполнения задач и доступа к общим хранилищам данных. Именно поэтому Grid-инфраструктуры, связанные с областью обработки данных наблюдения Земли, выделяют в отдельный класс Grid-систем, исследованию которых посвящен проект DEGREE [7] программы EGEE. Подобные системы включают следующие функциональные компоненты.

1. Метапланировщики (управляющие узлы) уровня всей системы в целом или ее отдельной, архитектурно значимой, части, которые обеспечивают передачу заданий (т.е. исполняемого программного кода

и данных) на требуемый вычислительный ресурс (в терминах пакета gLite [8] — Computational Element, или вычислительный элемент) или ресурс хранения (Storage Element). В состав программного кода метапланировщика входят алгоритмы планирования выполнения задач, которые могут отличаться для разных систем. Примерами метапланировщиков являются GridWay [9] или WMS (gLite, который в настоящее время не находит широкого применения в области исследования Земли). Существуют и другие примеры метапланировщиков, в частности GrAS [10]. Однако наиболее востребованным на сегодня является GridWay, включенный в состав последних версий Globus Toolkit [10] и активно применяемый в гетерогенных системах. Метапланировщики являются надстройкой над базовой Grid-инфраструктурой (работающей под управлением программного обеспечениея промежуточного уровня) и взаимодействуют не с конкретными аппаратными ресурсами, а с представляющими эти ресурсы Grid-сервисами. Для выполнения своих функций метапланировщики используют данные, предоставляемые информационными сервисами Grid-инфраструктуры, что позволяет им учитывать общее состояние системы при принятии решений о распределении задач между Grid-ресурсами. Для отправки задач на конкретные ресурсы системы используются интерфейсы, предоставляемые сервисами выполнения задач.

2. Локальные планировщики, например Torque [12], PBS Pro [13], Sun Grid Engine [14], LSF [15], которые обеспечивают управление выполнением задач на локальном ресурсе и взаимодействуют с метапланировщиком.

3. Распределенные вычислительные узлы и ресурсы хранения данных, доступ к которым и их использование должны быть синхронизированы. На узлах с этими ресурсами должны быть установлены также Grid-сервисы программной инфраструктуры (MDS, GRAM, GridFTP, RFT и т.д.), которые предоставляют на верхний уровень информацию о состоянии данного ресурса и тем самым позволяют реализовывать эффективное планирование выполнения задач.

Таким образом, выполнение задачи в Grid-среде является трехуровневым иерархическим процессом, на верхнем уровне которого находится метапланировщик, на среднем – локальные планировщики, а на нижнем – физические вычислительные ресурсы. В частности, такая иерархия поддерживается в системах на основе метапланировщика GridWay платформы Globus Toolkit или брокера ресурсов Resource Broker платформы gLite. Задачи пользователей поступают в очередь метапланировщика заданий. Этот компонент, используя данные от информационных сервисов Grid-системы и передавая команды сервисам передачи данных и управления задачами, распределяет эти задачи по ресурсам Grid-системы, реализуя некоторый алгоритм планирования с учетом статистики, предоставляемой информационным сервисом MDS.

Все вышеперечисленные компоненты и взаимосвязи между ними показаны на рис. 1.

Поскольку задачи обработки данных наблюдения Земли характеризуются высокой вычислительной сложностью и большим объемом используемых данных, то очень важным является обеспечение синхронизации как отдельных потоков выполнения, так и доступа к общим ресурсам, в частности хранилищам данных. Вопрос синхронизации доступа к общей памяти можно всесторонне исследовать с использованием аппарата сетей Петри. Данная статья посвящена построению именно такой модели, а также подходам, которые можно использовать для автоматизации этого процесса. При этом важно учитывать тот факт, что в структуре Grid-системы наблюдается фрактальность, т.е. синхронизацию доступа необходимо обеспечивать как на уровне метапланировщика и доступа к общему хранилищу (как к одному из общих ресурсов системы), так и на уровне локального планировщика и доступа к общей памяти вычислительных элементов.

Ситуация, когда группа задач или потоков выполнения одновременно использует общее сетевое хранилище или общую память, является достаточно распространенной в области обработки спутниковых данных и моделирования окружающей среды. В частности, разовый запуск региональной метеорологической модели WRF для территории Украины требует более 10 Гбайт входных данных.

Как правило, в современных сетевых системах хранения данных в качестве носителей информации используются жесткие диски. Одним из основных их свойств является нелинейное снижение производительности при увеличении числа параллельных запросов. Например, один жесткий диск позволяет выполнять однопоточное линейное чтение со скоростью около 80 МБ/с, однако при считывании двух линейных потоков скорость каждого из них будет не более 30 МБ/с. С точки зрения суммарного времени выполнения задач более предпочтительным является последовательное считывание данных. Поэтому актуальной является задача оптимизации доступа к общим ресурсам хранилища, в том числе путем блокировки параллельных процессов чтения/записи.

Grid-система должна обеспечить прозрачную для пользователя обработку его запросов на поиск и обработку данных, в том числе, синхронизацию доступа к необходимым данным, обработку этих данных и предоставление пользователю результатов обработки. Работа Grid-системы, предназначенной для обработки спутниковых данных, подробно описана в [16].

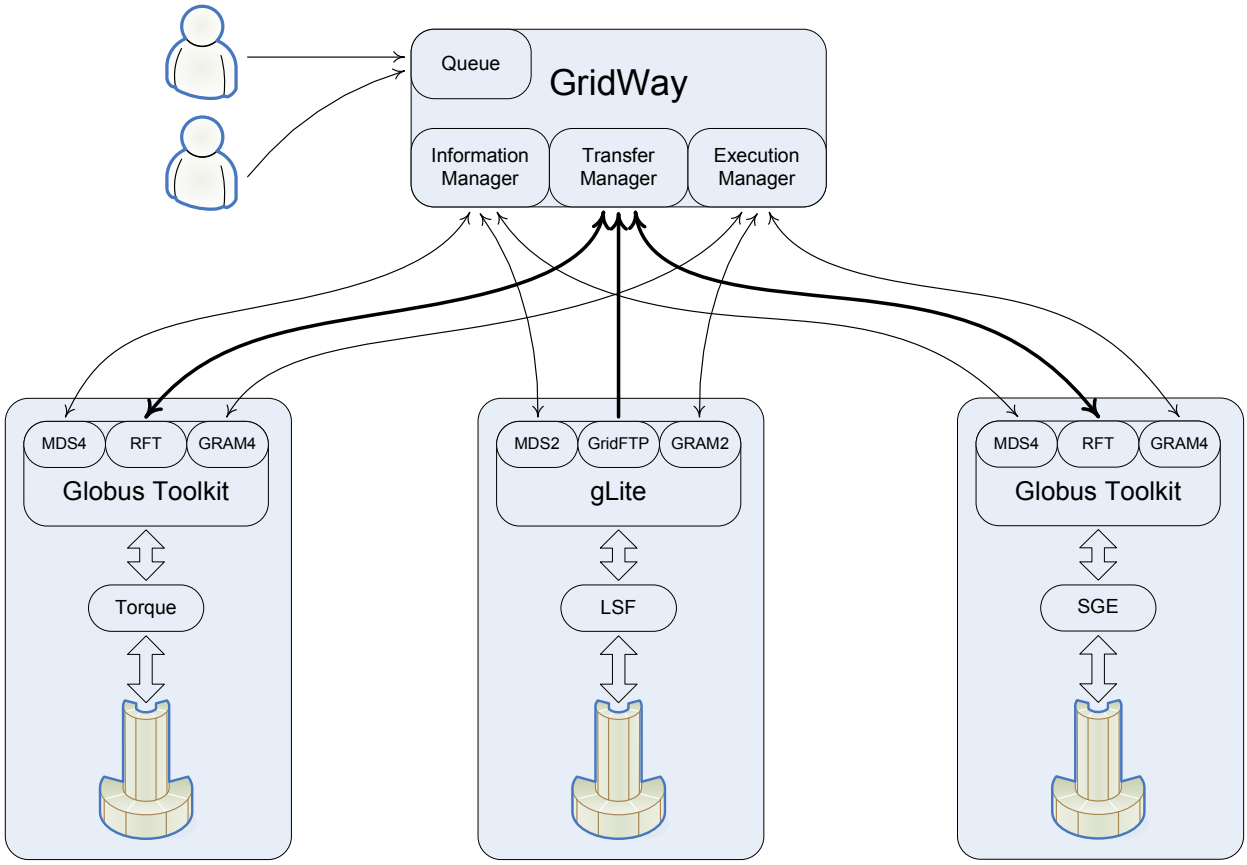

Рис. 1. Высокоуровневое представление Grid-системы обработки данных наблюдения Земли

Синхронизация доступа к данным и их корректная обработка предполагает выполнение некоторых базовых свойств, а именно:

– взаимное исключение (mutex) — синхронизация событий при обработке данных: два или более вычислительных узла не могут одновременно иметь доступ к общей области данных (общему хранилищу или общей памяти). Здесь и ниже в качестве вычислительного узла будем понимать либо отдельный вычислительный элемент, либо один из процессоров, входящих в его состав;

– равноправие или справедливость (fairness) — отсутствие дискриминации заданий. Если пользователь сформировал запрос (задание) и отправил его в систему, то обязательно наступит момент, когда это задание начнет выполняться и будет выполнено;

– отсутствие блокировок (deadlock free) — в системе не может возникнуть ситуация взаимной блокировки вычислений или доступа к данным (общим или распределенным).

Прежде чем приступать к построению системы, обладающей указанными свойствами, необходимо построить ее модель и исследовать свойства этой модели.

Модель работы узла grid-системы в виде сети петри

Модель работы вычислительного узла с 4 процессорами, обрабатывающего задания пользователя под управлением планировщика (глобального или локального уровня) в соответствии с общей схемой показана на рис. 2. Эта модель представляет собой одноцветную сеть Петри, начальное состояние которой описывается разметкой

M_0=(proc, query, want, work, wait, free, busy, 2, 3, 4, 5, 6, block, dop) = (5, 4, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0).

Позиции и переходы в данной сети Петри (СП) имеют такую семантику:

`proc` — свободные в данный момент времени процессоры;

`query` — запрос на выполнение задания от пользователя;

`want` — процессор активен и желает начать работу;

`work` — процессор получил доступ к общей памяти и выполняет вычисления;

`wait` — активный процессор ожидает доступа к общей памяти;

`block` — блокируется доступ к общей памяти и очереди ожидания, если они заняты другими процессорами;

`free` — управляющий узел сигнализирует о снятии блокировки общей памяти;
`busy` — управляющий узел сигнализирует о том, что общая память заблокирована для доступа.

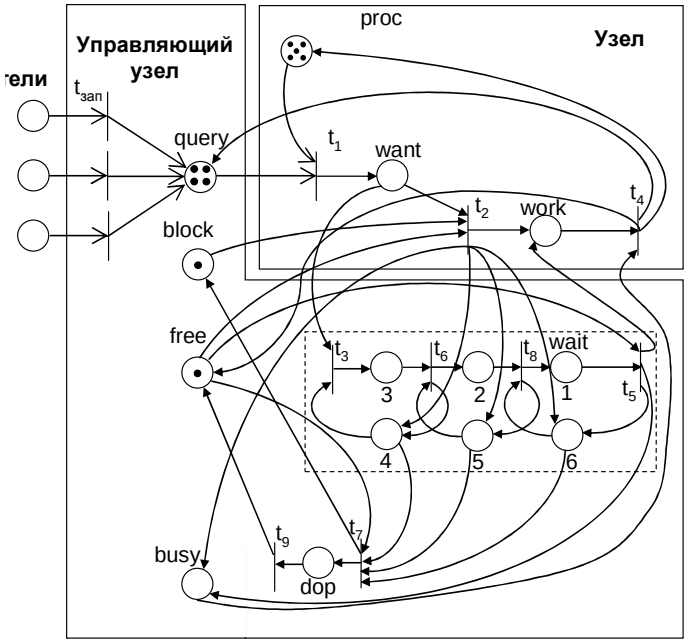

Рис. 2. Модель вычислительного узла, работающего под управлением планировщика над общей памятью с глубиной очереди 3

Показанная на рис. 3 модель достаточно точно описывает процесс выполнения задач с большими потребностями в данных в Grid-окружении. При этом элементам модели соответствуют следующие компоненты реальной системы:

– позиция `query` является представлением очереди задач метапланировщика (или планировщика) Grid-системы, а метки в ней — представлением задач в очереди (глобальной или локальной);

– метки в позиции `proc` соответствуют свободным вычислительным узлам Grid-системы (вычислительным элементам или процессорам);

При начальной разметке $M_0 = (4, 0, 0, 1, 1, 0, 0, 0)$ система находится в состоянии покоя (который иногда называют состоянием «сна»). Из семантики позиций следует и смысл переходов:

$t_{зап}$ — наличие запроса на обработку данных;
t_1 — один из свободных процессоров хочет работать;
t_2 — процессор получает доступ к общей памяти;
t_3 — активный процессор переходит в состояние ожидания;
t_4 — процессор завершил работу с общей памятью и она разблокирована;
t_5 — активный процессор из состояния ожидания переходит в состояние работы с общей памятью, при этом одновременно разблокируется переход активного процессора в состояние ожидания.

Переход t_6 и позиция без наименования между переходом t_3 и t_6 введены в результате модификации структуры сети Петри и преобразования полуцикла в цикл. Это сделано во избежание неоднозначностей в процессе последующего анализа сети.

При появлении запроса пользователя на выполнение задания срабатывает сначала один из переходов $t_{зап}$, а затем переход t_1. В результате этого один из свободных процессоров переходит в состояние активности (готовности к работе), т.е. переходит в позицию `want`. Если одновременно поступает 2 запроса, то выполнение одного из них будет задержано отсутствием маркера в позиции `block`. Дальнейшая работа модели очевидна. В [**Ошибка! Источник ссылки не найден.**] описаны результаты исследования структурных свойств модели. В частности в этой доказано следующее важное утверждение.

Утверждение 1. Представленная на рис. 2 сеть Петри, описывающая модель взаимодействия узлов Grid-системы с очередью доступа к общей памяти, является ограниченной, живой и не содержит недостижимых позиций.

Кроме того, для построенной сети выполняются свойства взаимного исключения и равноправия.

Система имитационного моделирования сетей петри

Для более наглядного представления работы описанных моделей, а также с целью автоматизации их анализа было разработано средство имитационного моделирования сетей Петри, обеспечивающее визуализацию работы СП, построение матриц инцидентности и транзиционной системы. В основу реализации системы имитационного моделирования положен объектно-ориентированный подход. В качестве средства реализации была выбрана система Visual C++ с библиотекой классов MFC. Графический пользовательский интерфейс разработанного средства моделирования показан на рис. 3.

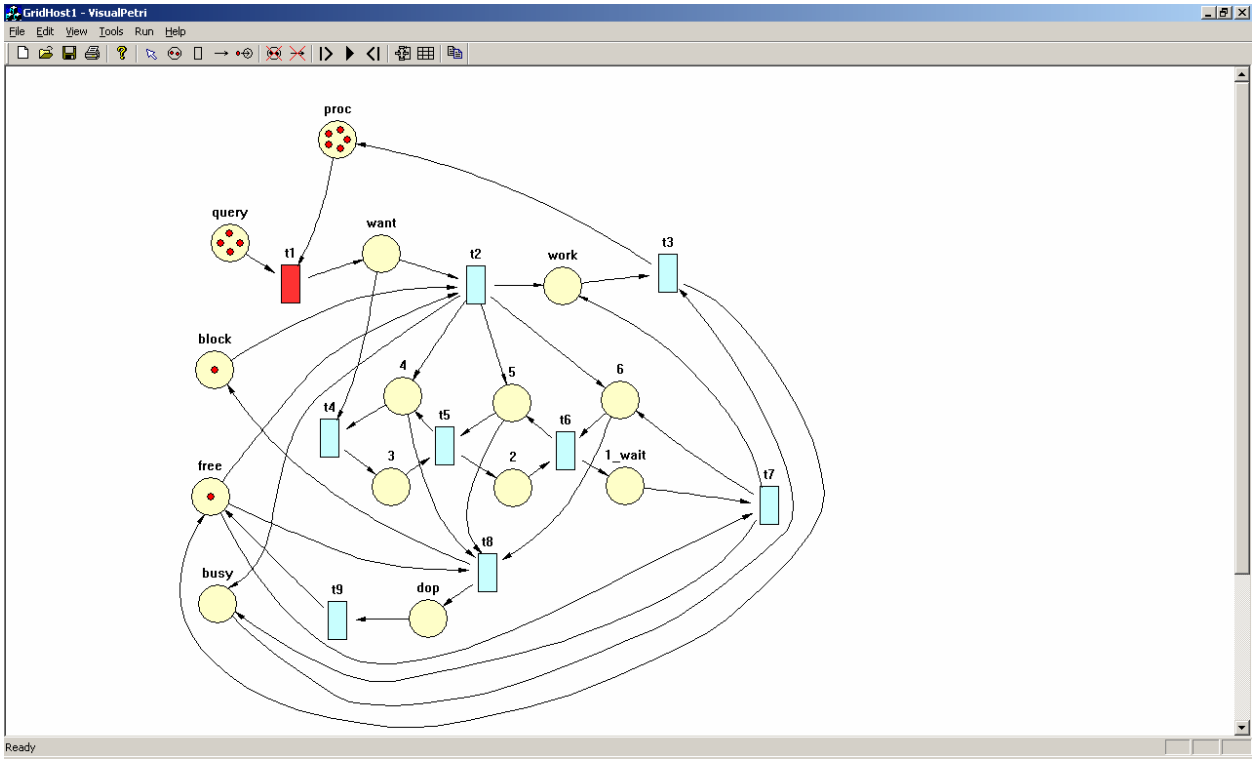

Рис. 3. Интерфейс пользователя системы имитационного моделирования сетей Петри

К основным функциям системы моделирования следует отнести следующие:

– визуальное построение сети Петри;
– построение матриц инцидентности и вычисление S- и T-инвариантов сети;
– эмуляция работы модели;
– построение транзиционной системы (графа достижимости состояний);
– поиск нужного состояния на графе достижимости.

Разработанную систему имитационного моделирования удобно применять для анализа моделей.

Структура классов (модулей) системы имитационного моделирования

Классы, в которых реализована основная функциональность системы имитационного моделирования сетей Петри, можно разбить на две основные категории.

– классы, которые описывают и реализуют работу СП и ее функциональных свойств, а также обеспечивающие проведение анализ СП, в частности, построение графа достижимости и матрицы инцидентности.

– классы библиотеки MFC, которые обеспечивают графический вывод элементов СП и предоставляют интерфейс взаимодействия пользователя с системой.

Такое разделение удовлетворяет шаблону проектирования Model-View-Controller и позволяет использовать различные графические интерфейсы для одних и тех же классов-компонентов СП.

В основу СП положен класс CPetriNet, который содержит множество узлов (объектов класса CHost) и множество соединений между узлами (объектов класса CEdge). Помимо классов, описывающих структуру СП, имеется еще класс CPetriScheme, который обеспечивает за построение графа достижимости на основе информации, инкапсулированной в классе CPetriNet.

Укрупненная диаграмма классов СП показана на рис. 4, а детализированная — на рис. 5.

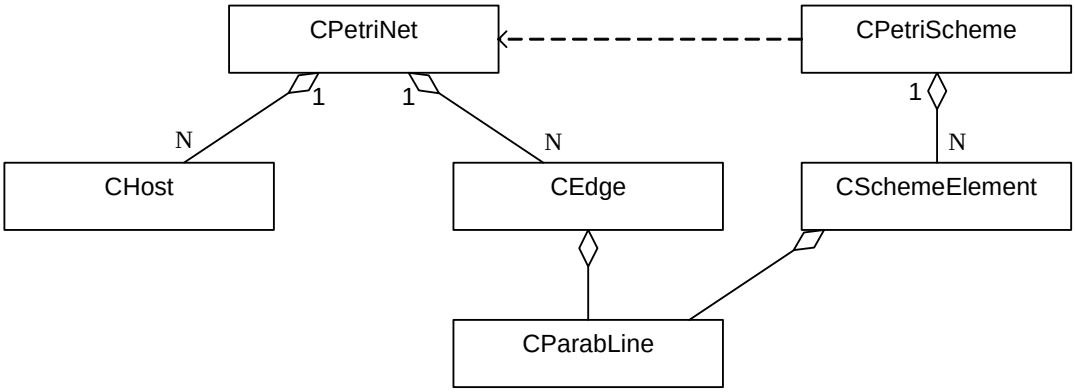

Рис. 4. Структура классов, описывающих свойства СП

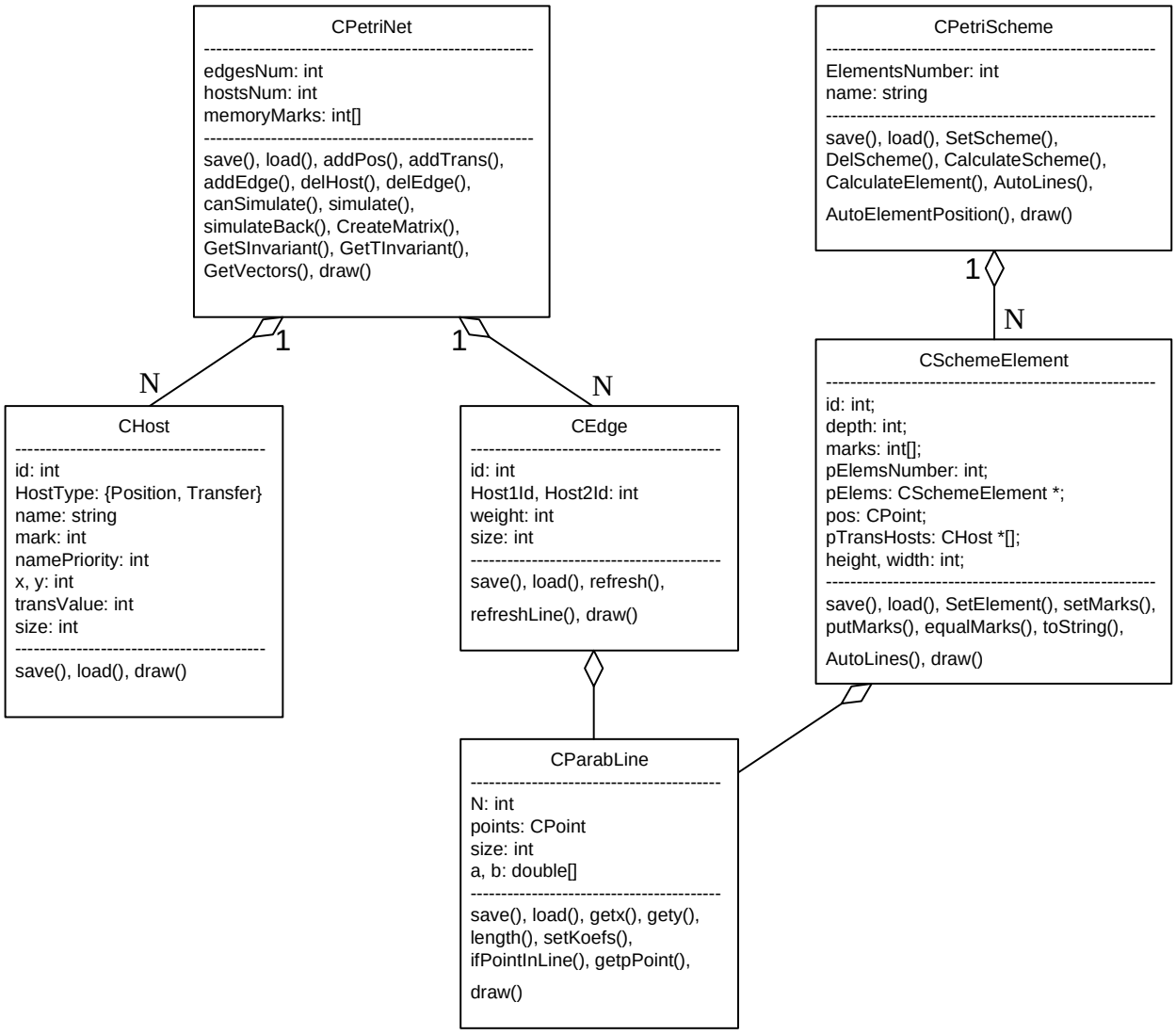

Рис. 5. Детальная структура классов описания СП

Теперь рассмотрим структуру классов (модулей) MFC. Классом всего приложения MFC является CVisualPetriApp. С его помощью выполняется инициализация класса документа CVisualPetriDoc, класса отображения CVisualPetriView и класса панели управления CMainFrame. При этом почти все основные события по обеспечению взаимодействия пользователя берет на себя класс отображения CVisualPetriView, который помимо перехвата событий обеспечивает также графическое отображение СП или графа достижимости. Обобщенная диаграмма классов MFC и их взаимосвязь с классами, представляющими сеть Петри, показана на рис. 6. На рис. 7 показаны те же классы с основными атрибутами и методами.

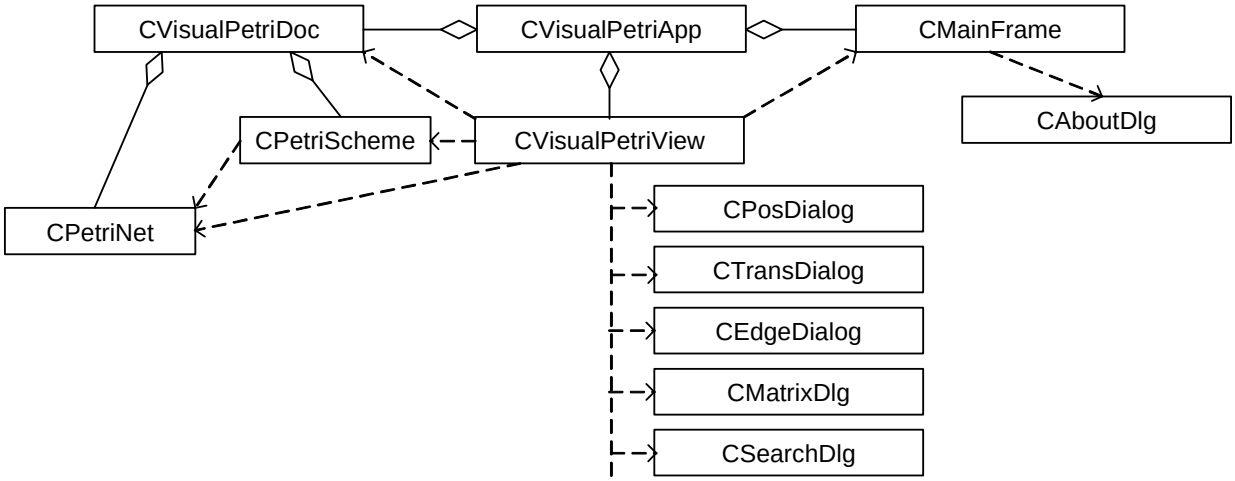

Рис. 6. Структура классов MFC

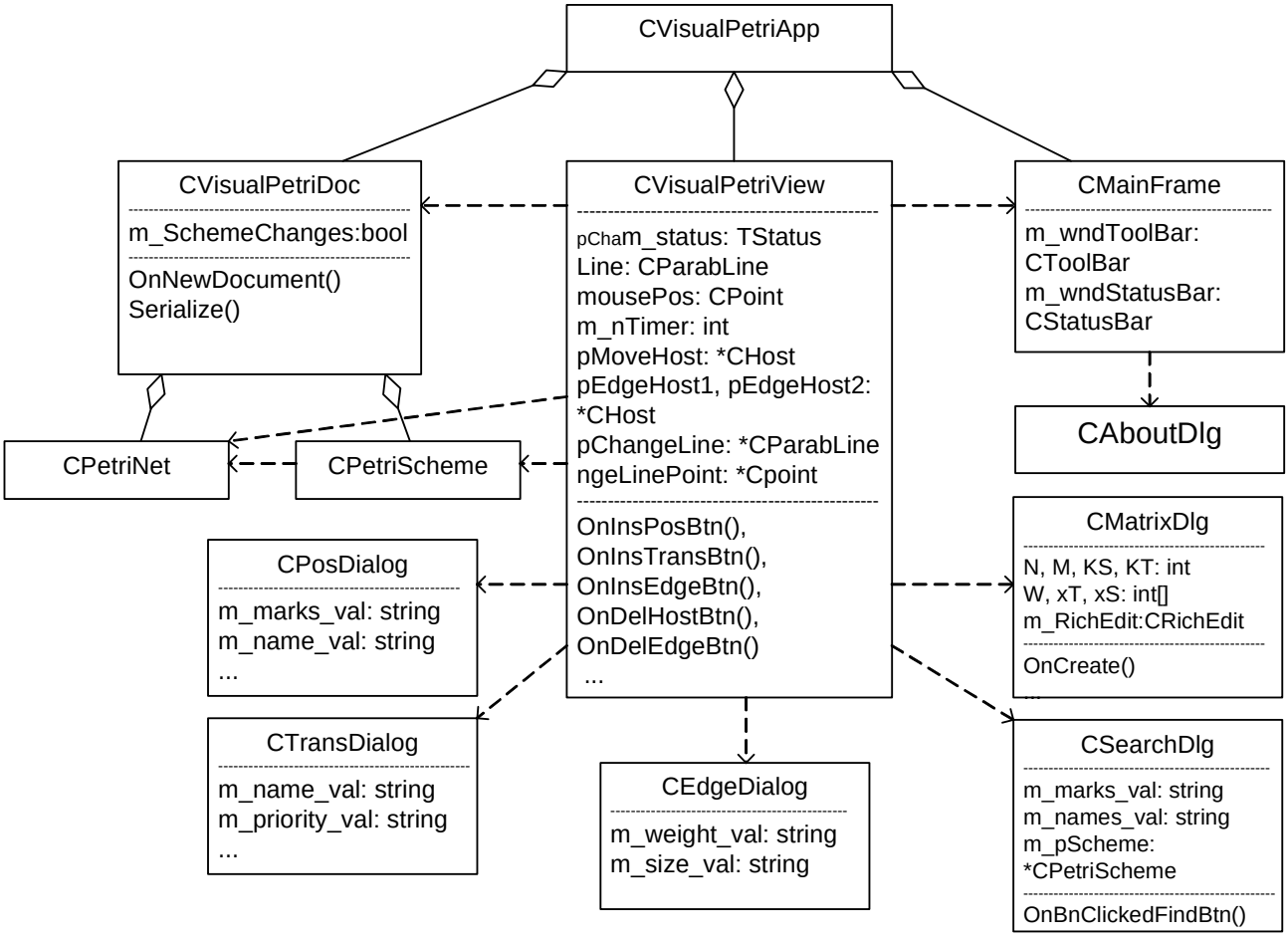

Рис. 7. Детальная структура классов MFC

Как видно из структуры классов MFC основная задача, связанная с построением СП и управлением ее работой, в том числе и построением графа достижимости, решается классом CVisualPetriView. Кроме того, как уже упоминалось, этот класс предоставляет пользовательский интерфейс и обрабатывает поступающие от пользователя события.

Выбранная внутренняя архитектура системы моделирования позволяет легко модифицировать ее функциональность и добавлять новые программные компоненты. С ее помощью можно выполнить анализ выполнимости ряда важных свойств сети Петри, а также получить графическое представление полученных результатов, в частности граф достижимости.

Выводы

Таким образом, в настоящее время распределенные системы обработки данных, в частности данных наблюдения Земли, активно развиваются. С учетом того, что подобные системы являются чрезвычайно сложными, в последнее время многие специалисты уделяют много внимания вопросам моделирования и всестороннего исследования таких систем. Сейчас известно множество подходов, которые позволяют выполнить необходимые исследования. В то же время следует отметить, что ни один из подходов не обеспечивает комплексного решения задачи моделирования.

В данной статье рассматривается предметная область построения Grid-систем обработки данных наблюдения Земли. При этом для исследования систем этого класса предлагается использоваться аппарат сетей Петри, который позволяет учитывать специфику таких систем, в частности необходимость синхронизации происходящих в них процессов. Для облегчения исследований и повышения эффективности работы предлагается использовать систему имитационного моделирования сетей Петри, которая является легко расширяемой и позволяет проверить выполнение наиболее важных свойств построенной сети.

1. *Згуровский М.З., Панкратова Н.Д.* Системный анализ: проблемы, методология, приложения. – К.: Наук. думка, 2005. – 744 с.
2. *Шелестов А.Ю.* Структурно-функциональный анализ компонентов Grid-систем // Проблемы управления и информатики. – 2007. – № 5. – С. 25 –32.
3. *Krauter K., Buyya R., Maheswaran M. A* Taxonomy and Survey of GRID Resource Management Systems and Distributed Computing // Software-Practice and Experience, John Wiley & Sons, Ltd. – 2001. – P. 1–10.
4. *Менаске Д., Алмейда В.* Производительность Web-служб. Анализ, оценка и планирование. – ДиаСофт, 2003. – 480 с.
5. *Куссуль Н.Н., Шелестов А.Ю., Лобунец А.Г.* Применение методов операционного анализа для оценки производительности GRID-систем// Кибернетика и вычислительная техника. – 2004. – Вып. 144. – С. 3 – 20.
6. *Дж. Питерсон* Теория сетей Петри и моделирование систем. – М.: Мир, 1984. – 264 с.
7. *Dissemination* and Exploitation of GRids in Earth sciencE, http://www.eu-degree.eu/.
8. *EGEE* gLite middleware, http://glite.web.cern.ch/.
9. *GridWay* Metascheduler, http://www.gridway.org/.
10. *Коваленко В.Н., Коваленко Е.И., Шорин О.Н.* Разработка диспетчера заданий грид, основанного на опережающем планировании. – М., ИПМ РАН – 2005. – 28 с.
11. *Globus* Toolkit, http://globus.org/.
12. *TORQUE* resource manager, http://www.clusterresources.com/pages/products/torque-resource-manager.php.
13. *PBS* Professional, http://www.pbsgridworks.com/PBSTemp1.3.aspx?top_nav_name=Products&item_name=PBS%20Professional&-top_nav_str=1.
14. *Sun* Grid Engine, http://www.sun.com/software/gridware/.
15. *Platform* LSF, http://www.platform.com/Products/platform-lsf-family/platform-lsf.
16. *Шелестов А.Ю.* Моделирование Grid-узла на основе сетей Петри // Проблемы управления и информатики. – 2008. – № 1. – С 104–113.