Нижегородский государственный университет им. Н.И.Лобачевского

Факультет Вычислительной математики и кибернетики

Инструментарий Alchemi для распределенных вычислений

Боголепов Д.К.
Кафедра математического обеспечения ЭВМ

Содержание

- Введение и основные концепции инструментария Alchemi
  - Принцип работы инструментария
  - Возможные применения инструментария
- Обзор компонент инструментария
  - Менеджер
  - Исполнитель
  - Пользователь
  - Межплатформенный Менеджер
- Разработка приложений с использованием инструментария
  - Две модели распределенных вычислений
  - Два подхода к разработке приложений
  - Пример разработки приложения для грид
- Основные выводы
  - Близкие проекты
  - Заключение

Введение

- В настоящее время *большинство* инструментариев по разработке приложений в грид предназначены для работы в ОС Unix
- Однако сегодня наблюдается быстро растущий интерес к технологиям грид со стороны коммерческих организаций, для которых *основной ОС* во многих случаях *является Microsoft Windows*
- Организации стремятся эффективно использовать свободную вычислительную мощность настольных компьютеров и рабочих станций посредством создания виртуального вычислительного ресурса, стоимость которого значительно ниже стоимости традиционных суперкомпьютеров

Для широкого распространения грид не хватает развития грид-платформ, поддерживающих ОС Microsoft Windows. Развитие подобных платформ может во многом определить успех грид-технологии

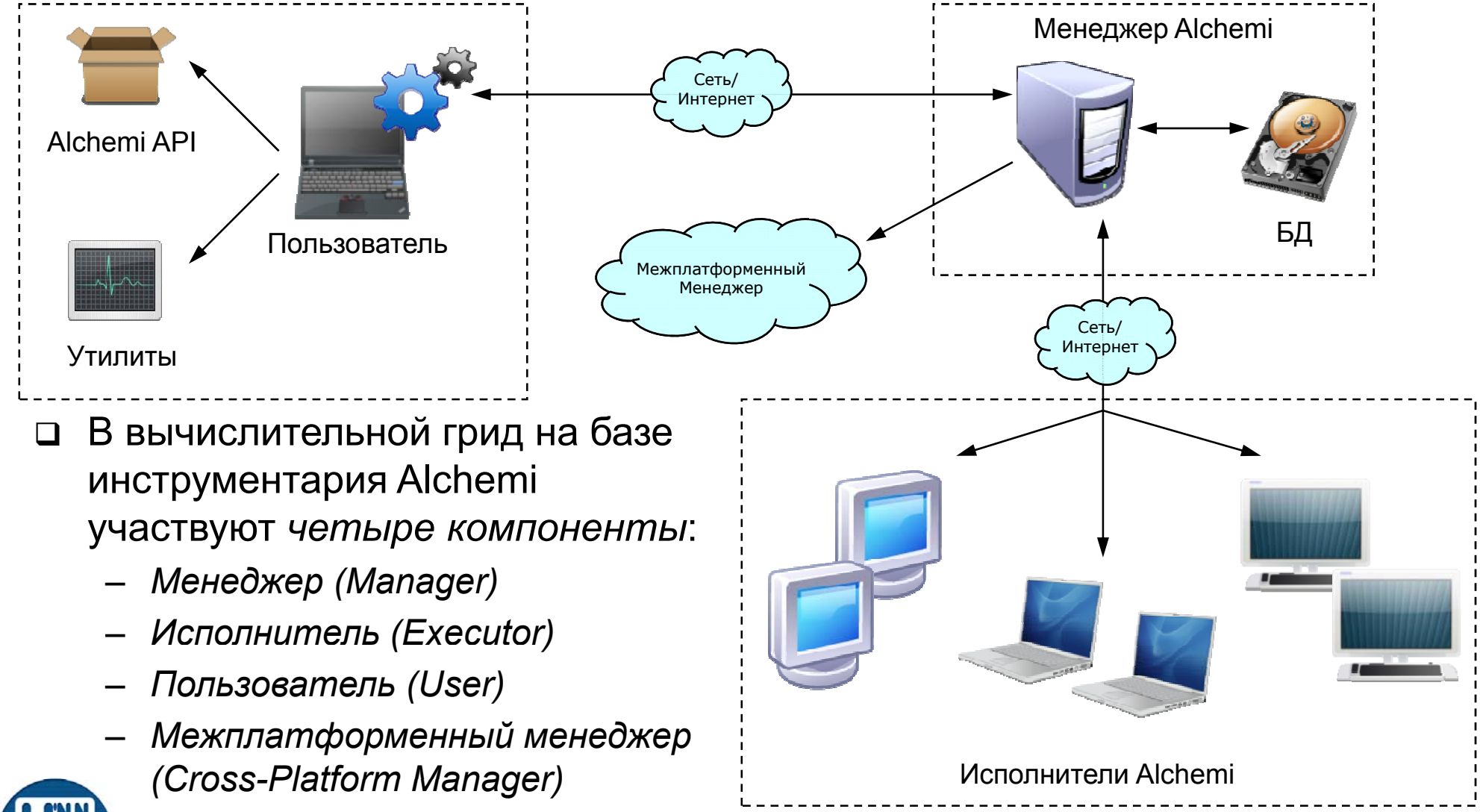
Введение

- *Практическая реализация* идеи грид *сталкивается с множеством трудностей*:
  - интеграция разнородность ресурсов,
  - низкая надежность инфраструктуры,
  - отсутствие средств разработки приложений,
  - проблема управление ресурсами,
  - обеспечение безопасности.
- Для решения перечисленных проблем можно использовать технологию Microsoft .NET Framework. Обладая встроенной поддержкой *удаленного исполнения*, *многопоточности*, *безопасности*, *удаленного доступа к данным*, *управления исполнением* и *несколькими языками программирования,* инструментарий Microsoft .NET Framework является идеальной платформой для разработки промежуточного ПО Грид

Alchemi

- Инструментарий Alchemi, разрабатываемый в университете Мельбурна, реализует ряд концепций *грид-компьютинга*. Интересен прежде всего тем, что для его реализации была выбрана платформа Microsoft .NET Framework и операционная система Microsoft Windows, тогда как большинство существующих инструментариев предназначены для работы в Unix
- Инструментарий Alchemi предназначен для организации высокопроизводительных *вычислительных грид-сетей* за счет утилизации свободных ресурсов. Основной упор сделан на *простоту* построения грид-сетей без ущемления *гибкости, надежности и масштабируемости*
- В настоящее время инструментарий Alchemi является *проектом с открытым исходным кодом* и доступен для свободного скачивания с ресурса SourceForge (www.sourceforge.net)

Схема работы Alchemi



- В вычислительной грид на базе инструментария Alchemi участвуют *четыре компоненты*:
  - *Менеджер (Manager)*
  - *Исполнитель (Executor)*
  - *Пользователь (User)*
  - *Межплатформенный менеджер (Cross-Platform Manager)*

Общая характеристика Alchemi…

- Инструментарий Alchemi *построен по клиент-серверной схеме*. Для объединения совокупности компьютеров в вычислительную грид:
  - нужно выбрать один узел, который будет играть роль сервера, и установить на него Менеджер,
  - на один или несколько компьютеров сети устанавливаются клиенты – Исполнители, которые настраиваются на работу с Менеджером
- В дистрибутив Alchemi входят *удобные программы инсталляции*, требующие минимальной настройки. Процесс создания грид является достаточно простым
- Исполнитель поддерживает два режима работы с Менеджером: *выделенный* (dedicated) и *невыделенный* (non-dedicated)
- Пользователи могут *разрабатывать*, *выполнять* и *осуществлять мониторинг* грид-приложений, используя специальный .NET API и инструментальные средства, включенные в Alchemi SDK

Общая характеристика Alchemi

- Инструментарий предлагает *две модели программирования*:
  - *модель грид-потоков*, которая позволяет очень просто разработать *новое приложение,*
  - *модель грид-заданий* для поддержки *межплатформенных программ*
- Дополнительная компонента – Межплатформенный Менеджер (Cross-Platform Manager) в виде веб-сервиса – предоставляет функциональную совместимость с другим *промежуточным программным обеспечением* грид
- Грид-сеть, построенная на базе Alchemi, может входить в состав другой, более крупной грид-сети

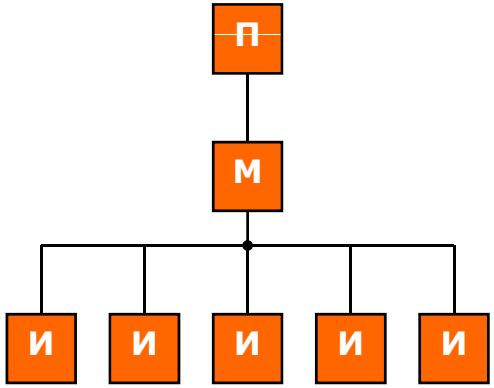
Возможные применения инструментария

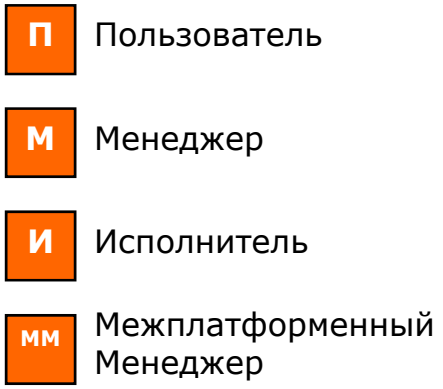
- Компоненты, входящие в состав инструментария Alchemi, могут использоваться для построения *различных конфигураций вычислительной сети*:
  - *кластер* (desktop grid),
  - *мультикластер* (multi-cluster grid),
  - *глобальная грид-сеть* (global grid)

Рассмотрим данные сценарии применения инструментария подробнее

Кластер

- Простейший сценарий применения – создание *вычислительного кластера из настольных компьютеров*. В этом случае используется *один Менеджер* и *несколько Исполнителей*, сконфигурированных на работу с Менеджером. Пользователи могут исполнять свои задания на кластере, подключаясь к единственному Менеджеру.

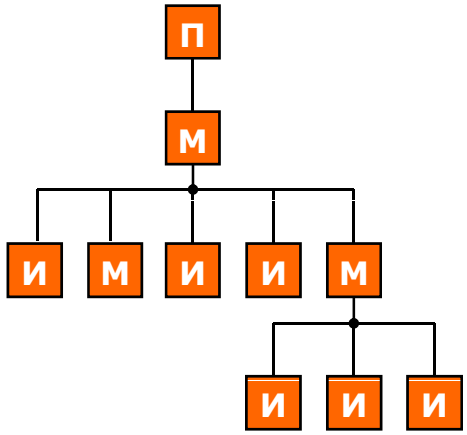




Данная схема является ненадежной и применима лишь для объединения небольшого числа компьютеров локальной сети

Мультикластер

- В этом случае *несколько Менеджеров* объединяются в *иерархическую структуру*. При этом *несколько Исполнителей* и *Пользователей* могут подключаться к Менеджерам *на любом уровне иерархии.* Менеджер на каждом уровне, за исключением самого верхнего, конфигурируется на работу с вышестоящим Менеджером как “*промежуточный*”. Для вышестоящего Менеджера он выглядит как Исполнитель



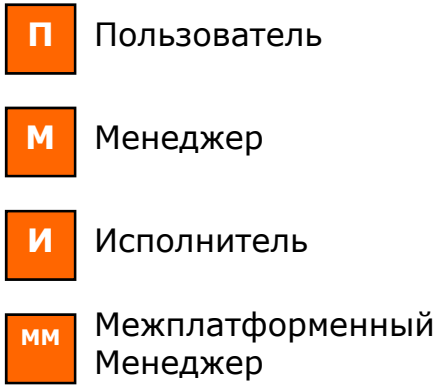


Схема лучше подходит для использования в Интернет и обладает большей надежностью по сравнению с предыдущим вариантом

Схема исполнения на мультикластере

- После того, как Пользователь запускает грид-приложение на корневом Менеджере, последний получает некоторое число грид-потоков, ожидающих своего исполнения в грид-сети
- По-умолчанию *все грид-потоки имеют наивысший приоритет* и начинают исполняться на доступных Исполнителях
- Некоторые Исполнители могут оказаться *промежуточными Менеджерами*. В этом случае приоритет потока понижается на единицу и выполнение потока планируется *на локальных Исполнителях промежуточного Менеджера*
- Понижение приоритета приводит к тому, что поток перемещается вниз по иерархии Менеджеров. Чем “ближе” поток к Исполнителю, тем с большим приоритетом он начинает выполняться

Это позволяет часть Alchemi-грид, находящуюся в пределах одного домена, разделять с другими организациями, обеспечивая совместную работу без воздействия на конечных пользователей

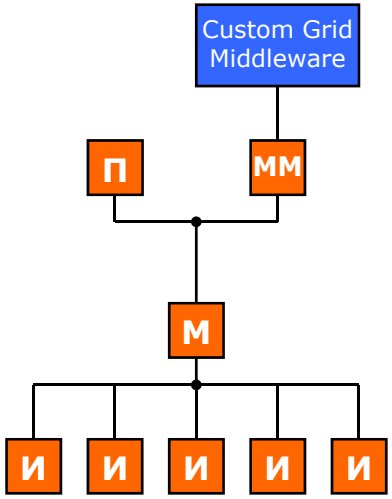
Глобальная грид

- *Межплатформенный Менеджер* может быть использован для конструирования *сегмента грид*. Компонента промежуточного ПО, такая как *брокер ресурсов*, может использовать Межплатформенный Менеджер в виде веб-сервиса для исполнения *межплатформенных приложений* (описываемые файлами задания) на сегменте Alchemi

Alchemi-грид может входить в состав более крупной грид-сети, например, в грид, построенную на базе Globus Toolkit

Обзор компонент инструментария

- Напомним, что в состав грид-сети, построенной на базе инструментария Alchemi, входят *четыре компоненты*:
  - *Менеджер,*
  - *Исполнитель,*
  - *Пользователь,*
  - *Межплатформенный Менеджер*



Рассмотрим данные компоненты инструментария подробнее

Менеджер…

- Любое приложение для инструментария Alchemi представляет собой *совокупность грид-потоков.* Менеджер *управляет выполнением* грид-приложений и входящих в их состав грид-потоков
- Исполнители *регистрируются* у Менеджера, который осуществляет непрерывный *мониторинг* их работы
- Грид-потоки, полученные от Пользователя, помещаются в пул, и составляется график их выполнения на доступных Исполнителях. Каждый поток имеет приоритет. Его можно задать явно при создании потока или отправке, а можно оставить значение по-умолчанию. Планирование осуществляется на основе *приоритетов* и *принципа* “*первым пришел – первым обслужен*”
- Исполнители возвращают Менеджеру *завершенные грид-потоки*, которые затем обрабатываются соответствующими Пользователями. В состав инструментария входит Scheduling API, позволяющее создавать свои планировщики

Менеджер

- *Стабильность* работы грид-системы играет *ключевую роль* в ее *эффективности*. Для повышения стабильности предпринимаются следующие шаги:
  - Менеджер осуществляет *непрерывный мониторинг* работы Исполнителей,
  - Грид-потоки, которые были запущены на отключившихся Исполнителях, *перепланируются и запускаются вновь* на доступных ресурсах (предполагается, что грид-потоки допускают возможность рестарта),
  - Все *данные постоянно сохраняются* на диск, так что в случае краха системы, Менеджер после повторного запуска не теряет своего состояния

Исполнитель…

- Исполнитель *получает грид-потоки* от Менеджера и *выполняет их*. Исполнитель может работать в *двух режимах*:
  - В *невыделенном* режиме Исполнитель выполняет грид-потоки во время простоя компьютера или когда пользователь явно разрешает это. В этом случае имеет место *одностороннее взаимодействие* между Исполнителем и Менеджером, так как в этом случае Исполнитель сам посылает запрос на получение грид-потока для выполнения,
  - Когда возможна *двухсторонняя связь* и используется *выделенный режим*, Менеджер взаимодействует с Исполнителем напрямую. В этом случае Менеджер явно указывает Исполнителю, какие потоки он должен выполнять

Исполнитель…

- Таким образом, модель выполнения приложений, используемая в инструментарии Alchemi, позволяет получить двойную выгоду:
  - от гибкого *управления ресурсами*, сочетая централизованное управление с принудительным исполнением и децентрализованное управление с программируемым планированием исполнения грид-потоков,
  - от гибкого *развертывания системы при ограничениях сети*, таких как proxy- и NAT-серверы, когда невозможна двусторонняя связь между Исполнителем и Менеджером

Таким образом выделенный режим лучше подходит для локальных сетей, а невыделенный для сети Интернет

Исполнитель

- Грид-потоки выполняются с *набором разрешений* **`AlchemiGridThread`**, которые настраиваются в рамках локальной политики безопасности .NET. Устанавливаемый набор разрешений определяет для грид-потоков (программный код которых является потенциально небезопасным) *среду выполнения*, *ограниченную по ресурсам* и *безопасную* для операционной системы
- Все потоки выполняются с *наименьшим приоритетом*, поэтому работа пользовательских приложений, имеющих более высокий приоритет, практически не затрагивается

Работа Исполнителей является безопасной для операционной системы и практически не отражается на пользовательских приложениях

Пользователь

- Грид-приложения выполняются на узлах Пользователей. Alchemi API *скрывает реализацию* грид от Пользователя и отвечает за выполнение необходимых служебных действий. К их числу относится:
  - отправление приложения и составляющих его грид-потоков на выполнение,
  - оповещение Пользователя о завершившихся грид-потоках и получение результатов их работы,
  - оповещение о неудачно завершившихся грид-потоках и предоставление деталей ошибки

Пользователь заботится только о бизнес-логике приложения. Все служебные действия, связанные с функционированием грид, возлагаются на Alchemi .NET API

Межплатформенный Менеджер

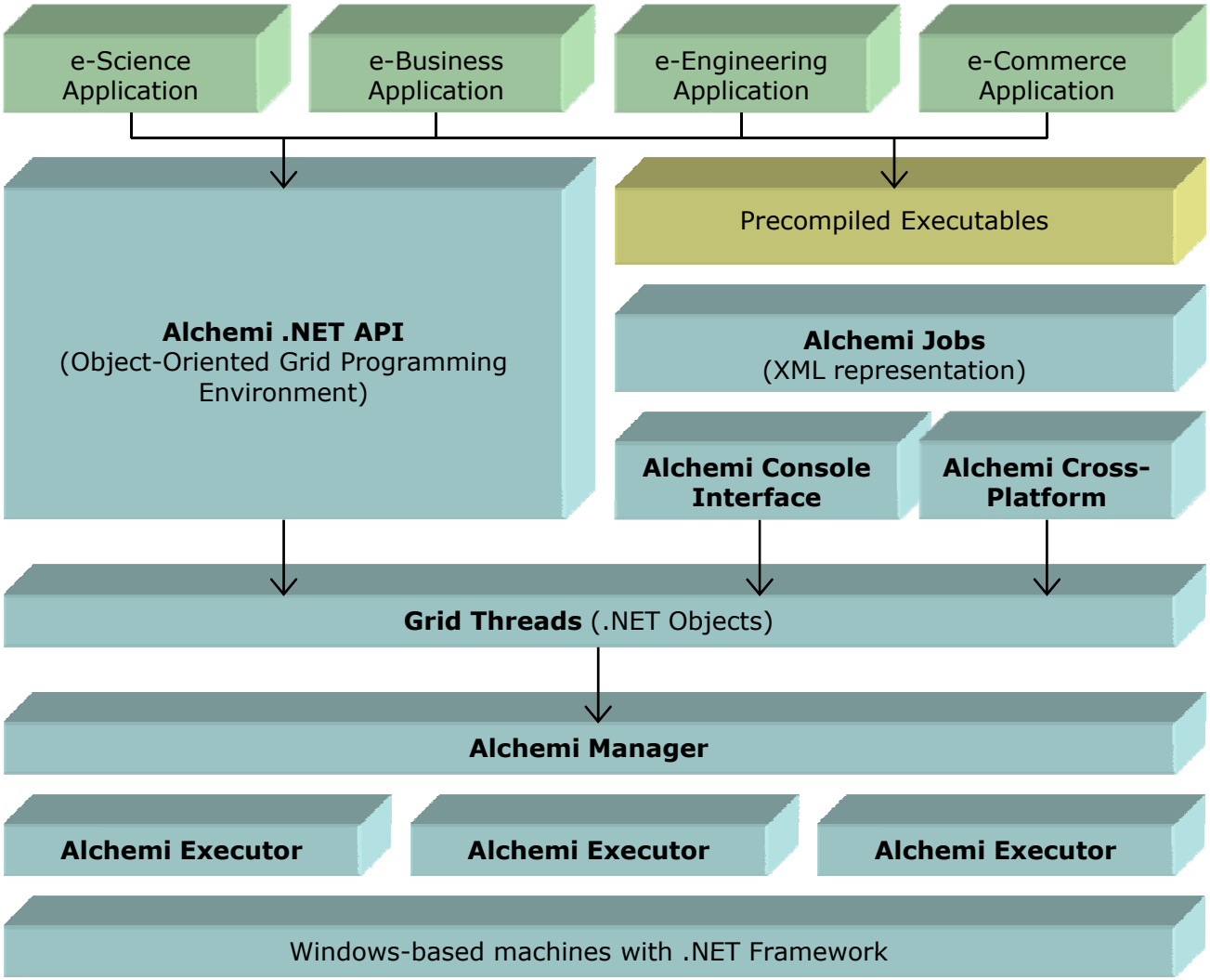
- Межплатформенный Менеджер – это *веб-сервис*, который обладает частью функциональных возможностей Менеджера и позволяет Alchemi управлять *выполнением грид-заданий* (в противоположность грид-приложениям, использующим модель грид-потоков)
- Задания, отсылаемые Межплатформенному Менеджеру, преобразуются им в представление грид-потоков, с которым работает Менеджер. Затем грид-потоки планируются и выполняются обычным способом
- Межплатформенный Менеджер обеспечивает совместимость с уже созданными грид-приложениями, позволяя использовать ресурсы Alchemi-грид на любой платформе, поддерживающей веб-сервисы

Вычислительная грид, построенная на базе Alchemi, может входить в состав “традиционной” грид-сети

Разработка грид-приложений

- Инструментарий Alchemi построен на основе парадигмы “*менеджер-исполнитель*”, в которой *центральная компонента* организует обработку *независимых единиц* параллельного вычисления
- В Alchemi этими независимыми единицами являются *грид-потоки*, содержащие инструкции, которые должны быть выполнены на одном грид-узле. Центральной компонентой является Менеджер
- *Грид-приложение состоит из множества грид-потоков*. Грид-приложения и грид-потоки создаются как обычные .NET-объекты. При запуске приложения, объекты грид-потоков отсылаются Менеджеру для исполнения в грид-сети на доступных Исполнителях
- Альтернативным вариантом являются *грид-задания на основе файлов*. Описание заданий хранится в XML-файлах специального формата. Обеспечивают совместимость с существующими приложениями. Задания могут отправляться посредством Консоли Alchemi или Межплатформенного Менеджера. XML-представление переводится в представление грид-потоков, которые затем отсылаются Менеджеру и исполняются обычным образом

Схема исполнения приложений



Модели грид-приложений

- Разработка грид-приложения может рассматриваться как написание *распределенной программы* для абстрактной “виртуальной” машины с множеством центральных процессоров. Основная задача различного промежуточного программного обеспечения состоит в том, чтобы предоставить программисту эту абстракцию. В различном программном обеспечении промежуточного уровня это делается по-разному
- Благодаря характеру инфраструктуры грид, приложения должны обладать следующими особенностями:
  - они могут быть распараллелены на множество независимых вычислительных частей,
  - эти части в основном работают над вычислениями, а не над установкой соединения и передаче данных между разными Исполнителями
- Как говорилось ранее, Alchemi поддерживает *две модели для параллельных вычислений*

Модель грид-потоков…

- Позволяет достаточно быстро разрабатывать *новые приложения* для грид. Достигается это благодаря *объектно-ориентированной модели программирования*, близкой к традиционному многопоточному программированию. Элементарной единицей параллельного исполнения в этом случае является *грид-поток*; множество грид-потоков образуют *грид-приложение*
- Разработчики могут полностью сконцентрироваться на основной логике и манипулировать такими абстракциями, как грид-приложение и грид-поток, используя все возможности современных языков программирования
- В будущем планируется поддержка грид-приложений, в которых отдельные грид-потоки взаимодействуют между собой
- Программы, использующие Alchemi .NET API, могут быть написаны на любом .NET-языке, например, C#, VB.NET, Managed C++, J#, JScript.NET

Модель грид-потоков

- Не следует забывать, что при этом *теряется совместимость* с другим ПО промежуточного уровня. Кроме этого, разработка приложения ограничивается единственной ОС (Microsoft Windows) и единственной технологией (Microsoft .NET Framework)
- Следует также отметить, что управляемый (managed) код мало подходит для *вычислительно-трудоемких задач*

Модель грид-заданий

- *Традиционные реализации* грид-инфраструктуры предлагают высокоуровневую абстракцию “виртуальной машины”, в которой минимальной сущностью параллельного выполнения является *процесс*
- *Спецификация задания*, которое должно быть выполнено в грид, состоит из списка входных файлов, выходных файлов и списка команд, которые должны быть выполнены
- Поддержка модели грид-заданий присутствует по следующим причинам:
  - для поддержки *существующих реализаций грид* или не .NET-приложений,
  - для *функциональной совместимости* с другим промежуточным программным обеспечением грид

В дальнейшем будет рассматриваться лишь модель грид-потоков, освоение модели грид-заданий выходит за рамки данного учебного материала

Сравнение грид и многопроцессорной машины

Многопроцессорный компьютер	***Грид***
Операционная система	Alchemi
Процессор	Исполнитель
Низкоуровневые сервисы ОС	Менеджер
Высокоуровневые сервисы ОС Системный API	Alchemi .NET API
Процесс	Грид-приложение
Поток	Грид-поток

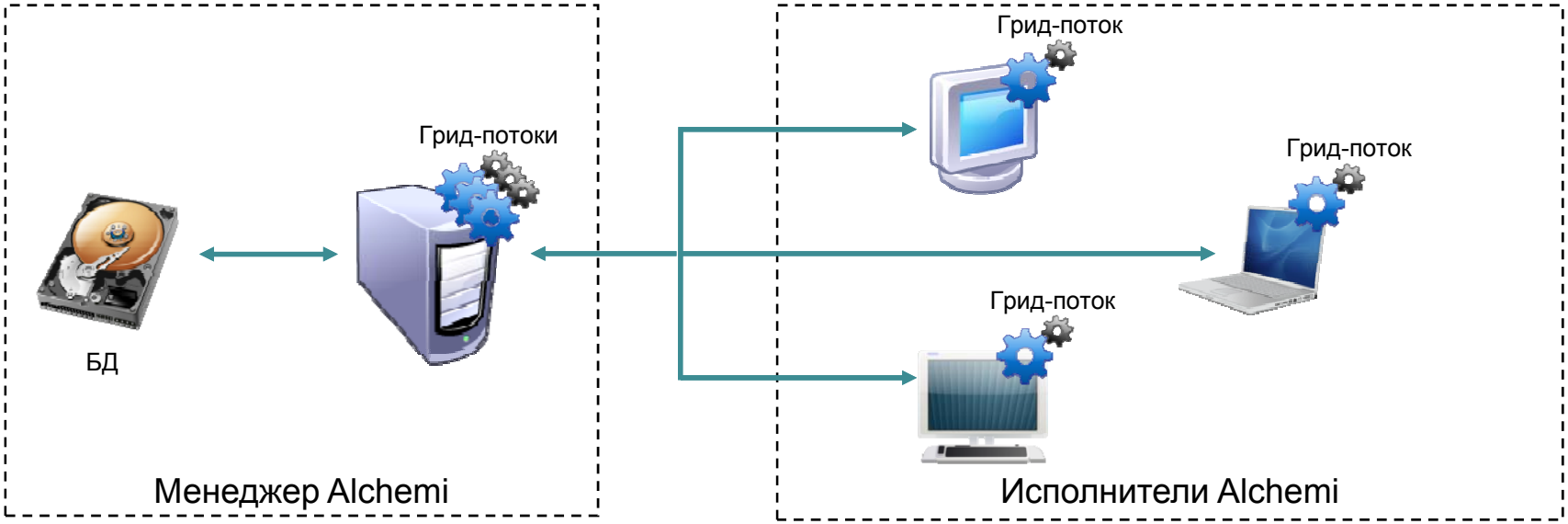
Применение модели грид-потоков

- Модель грид-потоков является достаточно универсальной, позволяя:
  - с легкостью разрабатывать *новые приложения* для грид,
  - в ряде случаев внедрять *существующие приложения* в грид

Рассмотрим оба варианта использования инструментария подробнее

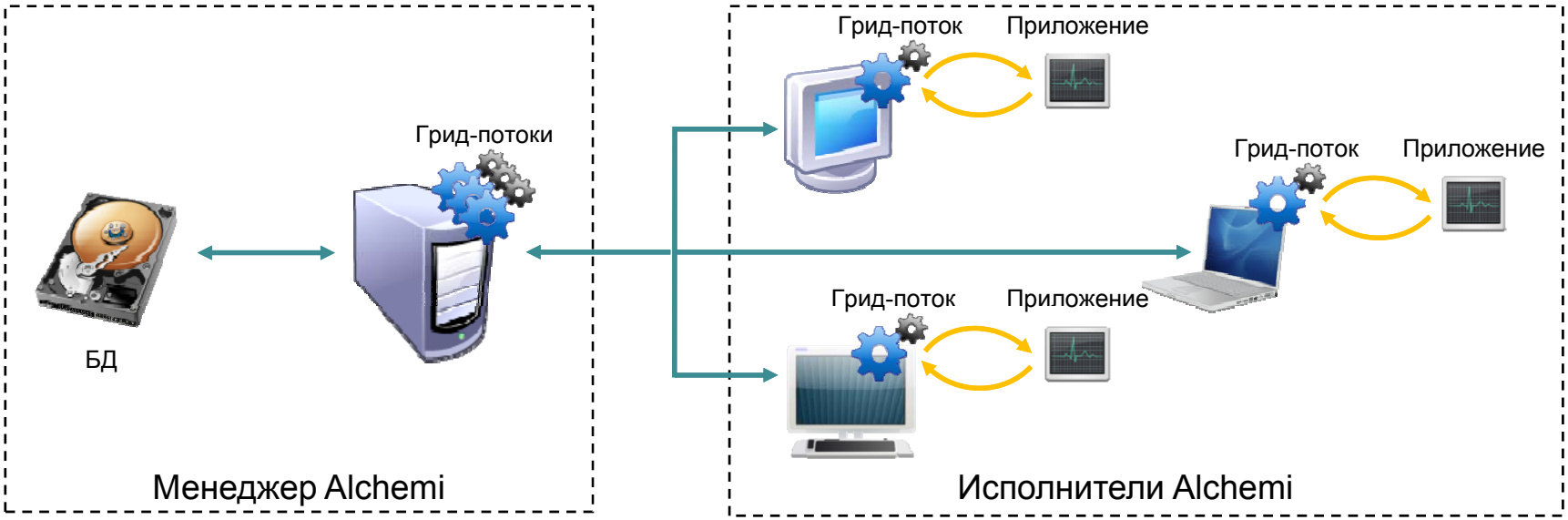
Разработка нового приложения

- Для разработки нового приложения необходимо выполнить анализ имеющейся вычислительной схемы и осуществить ее *разделение* (декомпозицию) на *части* (подзадачи), которые могут быть реализованы *независимо друг от друга*
- В инструментарии Alchemi этим “частям” соответствуют грид-потоки. *Все вычисления производятся в грид-потоках на отдельных узлах*



Внедрение существующего приложения

- Для внедрения существующего приложения необходимо выполнить анализ имеющегося набора входных данных и осуществить его *разделение на независимые части.* Предполагается, что *приложение может быть выполнено независимо для каждой порции данных*
- В этом случае грид-потоки создаются для *запуска существующего приложения* на отдельных узлах с *различными порциями данных*



Пример разработки приложения для грид

- В качестве примера будет разработано несложное грид-приложение для генерации простых чисел. Программа будет случайным образом генерировать большие числа и проверять в грид, какие из них являются простыми
- Для проверки воспользуемся самым простым методом, который перебирает делители от единицы и до заданного числа. Если делителей оказалось два (единица и само число), то, по определению, число является простым
- Рассмотрим последовательную реализацию данного алгоритма и разработку распределенного приложения с использованием инструментария Alchemi

Последовательная реализация

```
// Данная функция генерирует count целых чисел и
// последовательно проверяет, какие из них являются простыми
public void Generate(int count)
{
    Random random = new Random();
    for (int i = 0; i < count; i++)
    {
        int candidate = random.Next(1000000);
        if (CheckFactors(candidate) == 2)
        Console.WriteLine("Число {0} является простым", candidate);
    }
}

// Данный метод вычисляет количество делителей целого числа candidate
public int CheckFactors(int candidate)
{
    // Начальное число делителей полагаем равным нулю
    int factors = 0;
    // Поиск всевозможных делителей числа
    for (int i = 1; i <= candidate; i++)
    {
        if (0 == candidate % i) factors++;
    }
    return factors;
}
```

Распределенная реализация

- Прежде всего, необходимо создать класс *грид-потока (GThread)*, который содержит код для исполнения в грид-сети
- Для того, чтобы организовать параллельную работу независимых грид-потоков в грид-среде, нужно воспользоваться классом *грид-приложения (GApplication)*
- При разработке грид-приложения необходимо:
  - инициализировать грид-приложение,
  - добавить необходимое количество грид-потоков,
  - запустить грид-приложения

Класс грид-потока

```
[Serializable]
class PrimeNumberChecker : GThread
{
        public int Candidate;
        public int Factors = 0;

        // Создает новый класс грид-потока для проверки числа candidate
        public PrimeNumberChecker(int candidate)
        {
            Candidate = candidate;
        }

        // Данный метод запускает поток на выполнение
        public override void Start()
        {
            // Поиск всевозможных делителей числа
            for (int i = 1; i <= Candidate; i++)
            {
                if (0 == Candidate % i) Factors++;
            }
        }
    }
}
```

Класс грид-приложения…

```
static void Main(string[] args)
{
        Random random = new Random();
        for (int i = 0; i < 100; i++)
        {
            App.Threads.Add(new PrimeNumberChecker(random.Next(1000000)));
        }

        // Уставнока свойств соединения
        App.Connection = new GConnection("localhost", 9000, "user", "user");

        // Добавление зависимостей
        App.Manifest.Add(new ModuleDependency(typeof(PrimeNumberChecker).Module));

        // Добавление обработчика события ThreadFinish
        App.ThreadFinish += new GThreadFinish(App_ThreadFinish);

        // Запуск грид-приложения
        App.Start();

        Console.ReadLine();

        // Остановка грид-приложения
        App.Stop();
}
```

Класс грид-приложения

```
// Данный метод вызывается при завершении потока
private static void App_ThreadFinish(GThread thread)
{
    // Приведение класса GThread к классу PrimeNumberChecker
    PrimeNumberChecker pnc = (PrimeNumberChecker)thread;

    // Проверяем, является ли число простым
    bool prime = false;

    if (pnc.Factors == 2)
        prime = true;

    // Отображаем результаты
    Console.WriteLine("Число {0} простое? {1} ({2} делителей)",
                      pnc.Candidate,
                      prime,
                      pnc.Factors);
}
```

Близкие проекты

	Alchemi	Condor	SETI@home	Entropia	XtermWeb	Grid MP
Архитектура	Иерархическая		Централизованная			
Возможность интеграции в глобальную грид	Да	Нет	Нет	Нет	Нет	Нет
Технологии реализации	C# , технология веб-сервисов	C	C++, Win32	C++, Win32	Java, Linux	C++, Win32
Поддержка мультикластера	Да	Да	Нет	Нет	Нет	Да
Глобальный брокер ресурсов грид	Да (Gridbus Broker)	Да (Condor-G)	Нет	Нет	Нет	Нет
Многопоточная модель программирования	Да	Нет	Нет	Нет	Нет	Нет
Интеграции приложений	Низкоуровневая (в основном)		Отсутствует (единственное применение)	Низкоуровневая (в основном)		

Заключение…

- Инструментарий Alchemi, основанный на технологии Microsoft .NET, позволяет быстро и удобно сконструировать вычислительную грид-сеть и разработать соответствующие приложения
- Благодаря поддержке выполнения грид-заданий на основе файлов имеется возможность интеграции в глобальную кросс-платформенную грид с использованием брокера ресурсов (системы распределения заданий в грид по конкретным узлам в соответствии с некоторыми критериями)

Заключение…

- В будущем разработчиками панируется значительно расширить возможности инструментария в различных направлениях:
  - Планируется поддержка дополнительных функциональных возможностей API, включая взаимодействие между отдельными потоками,
  - Ведутся работы над поддержкой мультикластера из равноправных Менеджеров,
  - Планируется поддержка различных политик выделения ресурсов с учетом экономичности, качества обслуживания и эксплуатационных соображений,
  - Исследуется возможность перехода на стандарты OGSI посредством расширения существующего интерфейса управления заданиями. Интеграция с другими грид-системами, основанными на технологии Microsoft .NET (например, система OGSI.NET университета Виржинии) и соответствующим стандартам OGSI (Open Grid Services Infrastructure),
  - Планируется реализовать возможности грида данных, что позволит использовать ресурсы данных в дополнение к вычислительным ресурсам

Заключение

Вопросы