ALCHEMI
XÂY DỰNG LƯỚI TÍNH TOÁN

Đặt vấn đề

- Nhu cầu:
  - Công cụ phát triển Lưới tính toán hầu hết trên UNIX
  - Số lượng máy tính dùng Windows đang chiếm phần lớn
  - Tận dụng sức mạnh xử lý của các máy tính để bàn và máy trạm bằng cách tạo ra nguồn tài nguyên điện toán ảo
  - Chi phí thấp hơn rất nhiều so với việc sử dụng siêu máy tính truyền thống

Đặt vấn đề

- Thực tế đối mặt với một số khó khăn:
  - Khả năng tích hợp các nguồn tài nguyên không đồng nhất
  - Độ tin cậy của cơ sở hạ tầng thấp
  - Thiếu các công cụ để phát triển ứng dụng
  - Vấn đề quản lý tài nguyên
  - Bảo mật
- Microsoft .NET framework - lý tưởng để hỗ trợ xây dựng lưới tính toán giải quyết các vấn đề trên

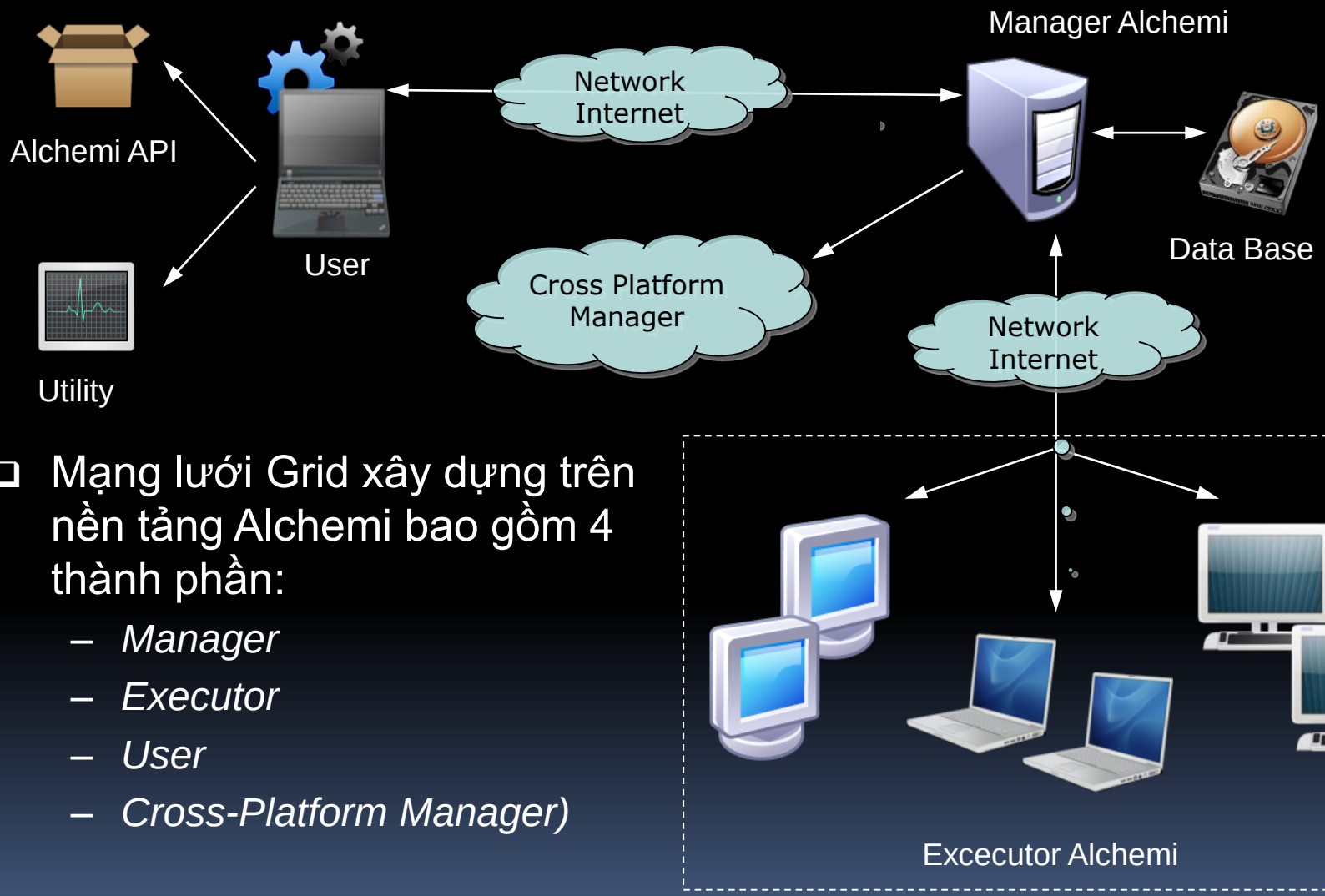
Alchemi

- Alchemi được hình thành và phát triển bởi đại học Melbourne (Úc).
- Phát triển trên nền tảng Microsoft .NET Framework và hệ điều hành Windows.
- Xây dựng hệ thống lưới tính toán gồm nhiều máy tính, mục đích tận dụng khả năng tính toán khi máy tính nhàn rỗi.
- Dễ dàng xây dựng Lưới tính toán mà không làm ảnh hưởng đến sự linh hoạt, độ tin cậy và khả năng mở rộng

Alchemi

- Sử dụng để phát triển một số ứng dung khoa học:
  - BLAST – Basic Local Alignment Search Tool – Tính tương đối giữa các chuỗi khoa học
  - Gridbus broker – sử dụng Alchemi tham gia vào dự án mạng lưới tính toán toàn cầu
  - CSIRO – mô hình hóa và mô phỏng ứng dụng thủy văn
  - Xử lý bảng tính Excel
  - Satyam của Ấn Độ - phân tích dữ liệu ứng dụng trong việc phát triển ung thư

Quy trình hoạt động Alchemi

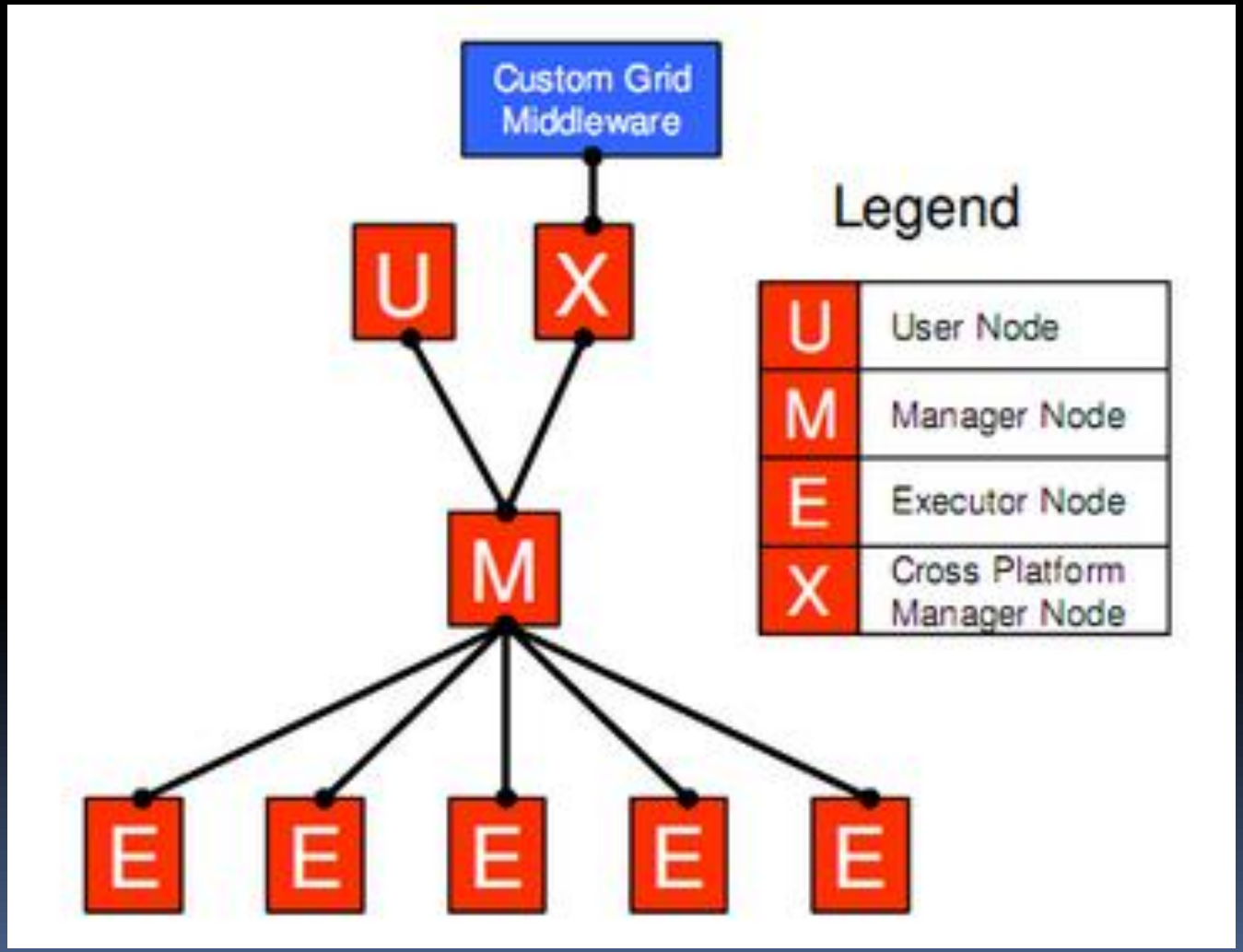


- Mạng lưới Grid xây dựng trên nền tảng Alchemi bao gồm 4 thành phần:
  - *Manager*
  - *Executor*
  - *User*
  - *Cross-Platform Manager)*

Đặc điểm cấu trúc Alchemi

- Cấu trúc hoạt động Client – Server.
- Kết hợp máy tính trong mạng tạo thành lưới tính toán:
  - Chọn một nút hoạt động như một máy chủ Server, cài đặt Manager lên đó
  - Các máy tính còn lại đóng vai trò Client – Executor lên đó, kết hợp với Manager để hoạt động.
- Quá trình cài đặt và thiết lập diễn ra đơn giản

Phân loại các thành phần Alchemi



Manager

- Grid Application được chia thành các luồng
- Điều khiển, quản lý ứng dụng, các luồng dữ liệu khi thực hiện trên hệ thống tính toán Lưới
- Tập hợp các luồng được Manager lưu lại, phân chia lịch thực hiện trên các Executor
- Nguyên tắc: ”Đến trước – thực hiện trước ”
- Xong công việc Executor gửi trả lại kết quả cho Manager để tổng hợp.

Executor

- Nhận các luồng từ Manager và thực hiện chúng.
- Bao gồm 2 trạng thái:
  - Dedicater: Manager trực tiếp quản lý Executor – chỉ cho Executor biết cần phải thực hiện luồng nào
  - Non – dedicated: Thực hiện trong suốt quá trình nhàn rỗi. Giao tiếp Manager và Executor là 1 chiều

User

- Ứng dụng Grid Application được chạy trên các nút đóng vai trò là User.
- Alchemi API tiếp nhận yêu cầu thực hiện ứng dụng từ các nút User
- Các công việc:
  - Gửi ứng dụng
  - Phân chia ứng dụng thành luồng công việc
  - Tổng hợp kết quả luồng công việc
  - Thông báo kết quả ứng dụng

Cross-Platform Manager

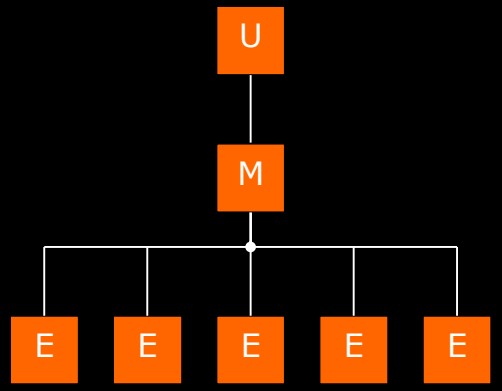
- Giao diện dịch vụ Web Service
- Cho phép thực hiện các Grid Jobs
- Grid Jobs tiếp nhận từ Cross-Platform phân chia về các Manager
- Manager tiếp nhận và phân chia thành các luồng
- Hỗ trợ tùy chỉnh tương thích với tất cả nền tảng Web

Đặc điểm cấu trúc Alchemi

- Cross Platform Manager – dịch vụ Web cung cấp khả năng tương tác với Lưới khác.
- Dễ dàng tạo một mạng Lưới rộng lớn từ Alchemi.
- Cho phép xây dựng các cấu trúc khác nhau:
  - Cluster – Desktop Grid
  - Multi-cluster Grid
  - Global Grid

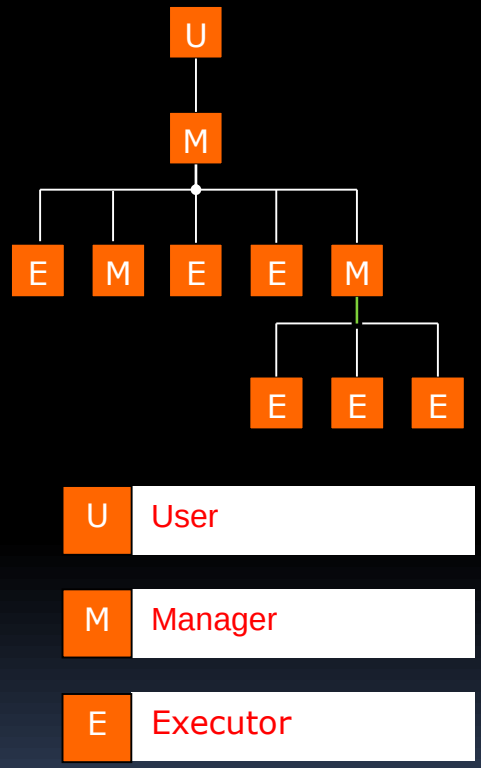
Cluster

- Mô hình đơn giản nhất tạo ra Lưới tính toán dựa trên máy tính để bàn
- Trong mô hình này bao gồm:
  - Một Manager
  - Một vài Executor kết hợp với Manager
  - User có thể kết hợp với Manager để gửi công việc lên Cluster



Multi-cluster

- Trong mô hình này các Manager kết hợp với nhau theo mô hình phân cấp.
- Các Executor và User có thể kết nối với Manager ở mọi cấp độ hệ thống phân cấp
- Các Manager trên mỗi cấp chịu sự quản lý của Manager cấp cao hơn

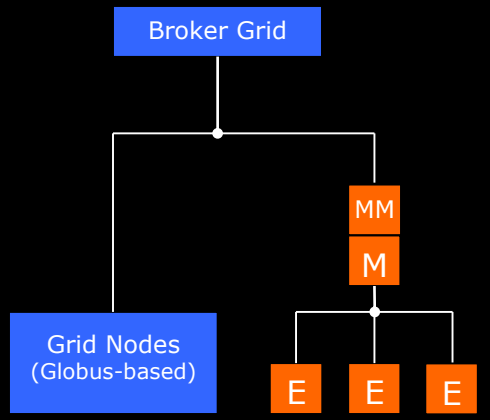


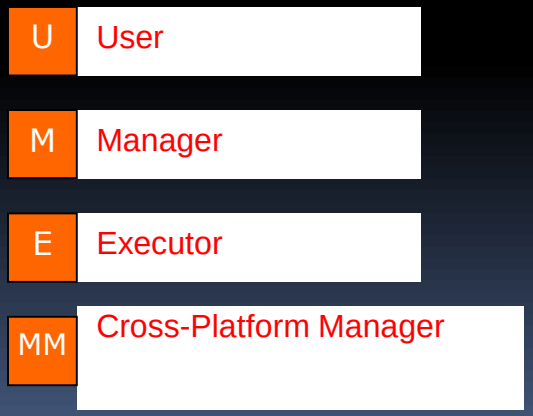
Multi-cluster

- User thực hiện Grid Application trên Manager "gốc" khi đó toàn bộ luồng sẽ được thực hiện trên toàn Lưới
- Theo mặc định các luồng có quyền ưu tiên cao nhất sẽ được xử lý trên Executor "nhàn rỗi"
- Một vài Executor có thể đóng vai trò Manager trung gian. Trong trường hợp này quyền ưu tiên của luồng sẽ bị giảm đi 1

Global Grid

- Cross-Platform Manager có thể sử dụng để xây dựng mạng Lưới "phân khúc"
- Broker sử dụng Cross-platform dưới dạng Web Service để thực hiện các ứng dụng trên nền tảng Alchermi





Yêu cầu hệ thống

- Yêu cầu:
  - Máy tính trong Lưới cài đặt .NET Framework
  - Nút tính toán – cài đặt Executor
  - Nút quản lý – cài đặt Manager
- Software Development Kit:
  - Alchemi Console: Công cụ giúp quản lý theo dõi quá trình thực hiện công việc của Executor, mức độ sử dụng tài nguyên CPU
  - Alchemi.Core.dll: Thư viễn chưa các lớp cần thiết để tạo ứng dụng chạy trên hệ thống Alchemi

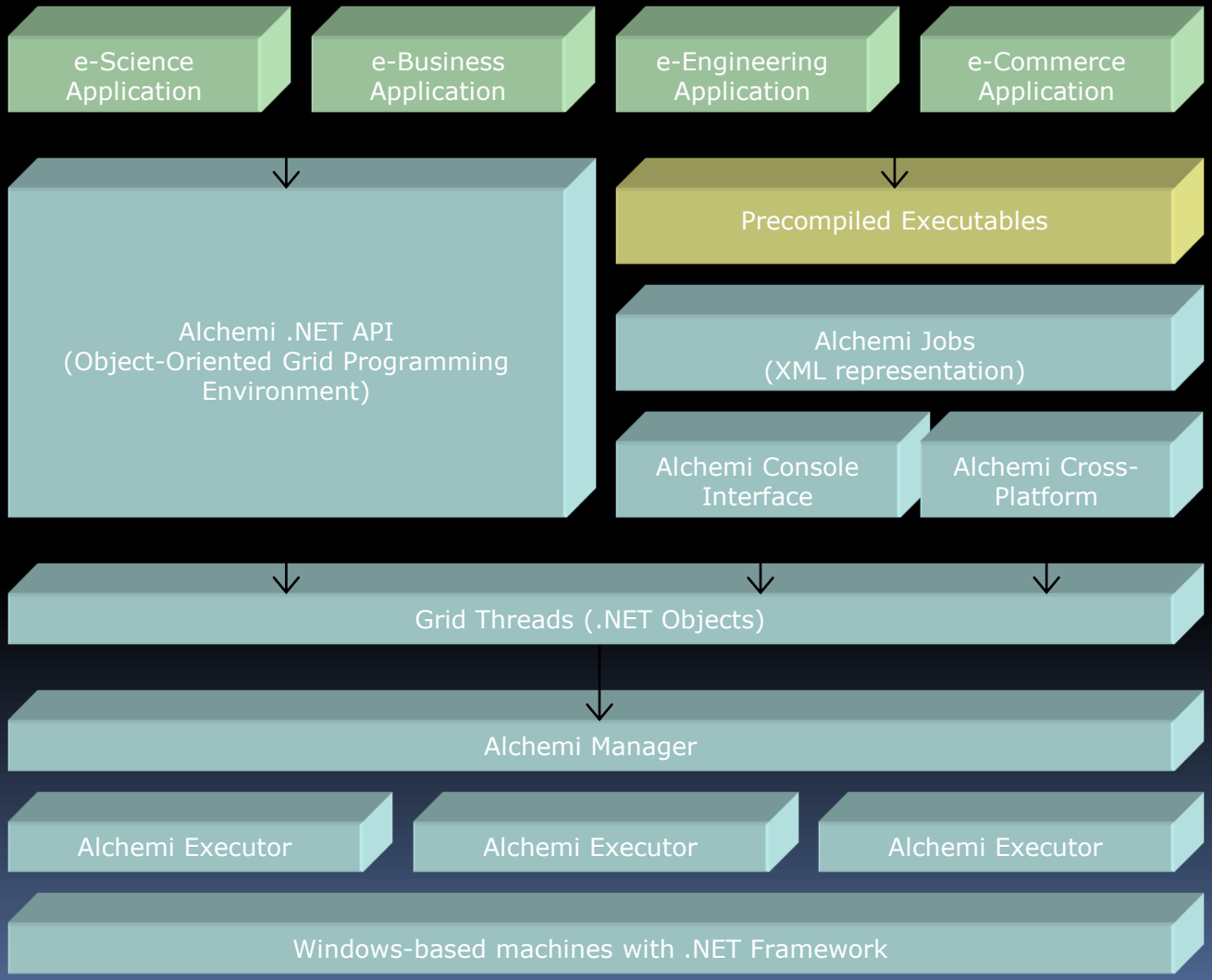
Ứng dụng trên Alchemi

- Alchemi xây dựng theo mô hình "Manager - Executor" trong đó một thành phần trung tâm đóng vai trò xử lý các đơn vị độc lập tính toán song song.
- Trong Alchemi các đơn vị độc lập với nhau được gọi là các luồng – thực hiện trên 1 nút
- Grid Application chứa tập hợp các luồng
- Grid Application và luồng được tạo lên từ các thành phần .NET.

Ứng dụng trên Alchemi

- Khi ứng dụng thực hiện, các thành phần của luồng liên hệ với Manager để thực hiện trên Lưới
- Danh sách các luồng, công việc được mô tả trên tệp tin định dạng XML – đảm bảo khả năng tương thích với các ứng dụng.

Mô hình thực hiện ứng dụng



Mô hình Grid Application

- Phát triển ứng dụng trên Lưới tương tự như việc viết một chương trình phân phối cho các một máy tính có nhiều CPU thực hiện
- Khả năng suy nghĩ "trừu tượng"
- Middlware khác nhau cung cấp nền tảng lập trình, phân chia công việc khác nhau
- Do tính chất đặc biệt của Lưới:
  - Song song trên các đơn vị tính toán độc lập
  - Thực hiện chính trong quá trình tính toán, không phải mô hình chuyển đổi dữ liệu giữa các Executor

Mô hình Grid Thread

- Alchemi hỗ trợ phát triển nhanh chóng các Grid Thread mới
- Mô hình lập trình đa luồng truyền thống
- Đơn vị thực hiện song song là luồng
- Hỗ trợ nhiều ngôn ngữ lập trình như :
  - .NET: C#, VB
  - Manager C++
  - J#
  - Jscript.NET

Mô hình Grid Thread

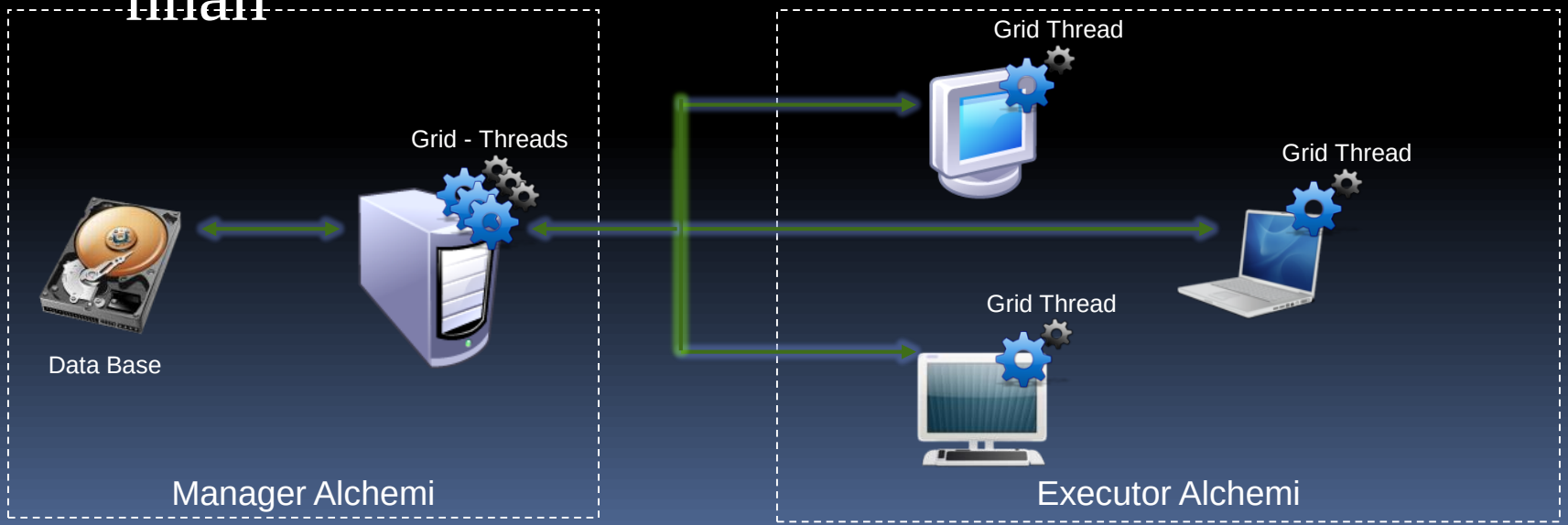
- Công việc trong Lưới tính toán bao gồm:
  - Danh sách tập tin đầu vào
  - Danh sách tập tin đầu ra
  - Danh sách các lệnh được thực thi
- Giới hạn trên nền tảng Windows và .NET Framework
- Khả năng quản lý code chưa tương thích với các lập trình chuyên sâu

Grid Computing vs Multi-processor Computer

Multi-processor Computer	*Grid Computing*
Operating System	Alchemi
Processor	Executor
Low-Level OS Service	Manager
High-Level OS Service System API	Alchemi .NET API
Process	Grid Application
Threads	Grid Threads

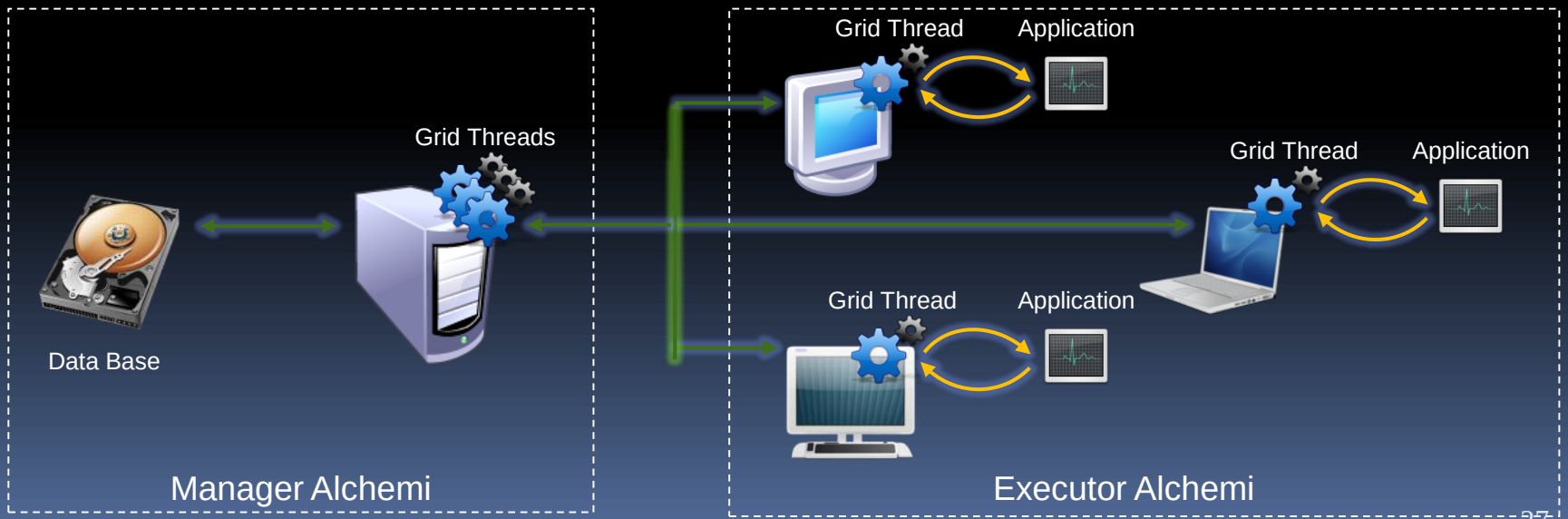
Phát triển ứng dụng

- Để phát triển ứng dụng mới cần thực hiện quá trình phân tích sơ đồ tính toán, cách phân chia tính toán thành từng phần riêng biệt (công việc không có sự liên quan với nhau)
- Tính toán luồng được thực hiện trên các nút cá nhân



Phát triển ứng dụng

- Phân tích bộ dữ liệu đầu vào và thực hiện phân chia thành các luồng riêng biệt. Ứng dụng có thể thực hiện trên các bộ dữ liệu riêng biệt không liên quan.
- Các luồng khởi tạo trên các nút riêng biệt với các thành phần dữ liệu khác nhau



Ví dụ

- Xây dựng chương trình kiểm tra tính nguyên tố của 1 số.
- Yêu cầu
  - số lượng số nguyên tố nhập vào
  - Số cần kiểm tra lấy ngẫu nhiên

Chương trình tuần tự

```
// Hàm số này kiểm tra số count có phải là số nguyên tố hay không
// Kiểm tra tuần tự xem một số có phải số nguyên tố hay không
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("Number {0} is prime", candidate);
    }
}

// Hàm này cho phép tính số lượng ước của một số candidate bất kỳ
public int CheckFactors(int candidate)
{
    // Số lượng ước ban đầu gán bằng 0
    int factors = 0;
    // Tìm kiếm tất cả các ước có thể có của số candidate
    for (int i = 1; i <= candidate; i++)
    {
        if (0 == candidate % i) factors++;
    }
    return factors;
}
```

Thực hiện song song trên Lưới

- Trước tiên tạo 1 lớp GThread – chứa câu lệnh để thực hiện trong Lưới
- Để tổ chức công việc song song các luồng không liên quan trong môi trường Lưới, cần phải thực hiện lớp ứng dụng (Grid Application)
- Khi thực hiện Grid Application thì cần:
  - Khởi tạo ứng dụng
  - Xác định số lượng các luồng
  - Chạy ứng dụng trên Lưới

Lớp GThread

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

        // Thiết lập lớp mới để kiểm tra số Candidate
        public PrimeNumberChecker(int candidate)
        {
            Candidate = candidate;
        }

        // Khởi động để thực hiện luồng
        public override void Start()
        {
            // Tìm kiếm các ước của candidate
            for (int i = 1; i <= Candidate; i++)
            {
                if (0 == Candidate % i) Factors++;
            }
        }
    }
}
```

Lớp Grid Application

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

        // Thiết lập liên kết
        App.Connection = new GConnection("localhost", 9000, "user", "user");

        //Thêm vào các mô đun cần thiết
        App.Manifest.Add(new ModuleDependency(typeof(PrimeNumberChecker).Module));

        // Thêm vào sự kiện ThreadFinish
        App.ThreadFinish += new GThreadFinish(App_ThreadFinish);

        // Khởi động ứng dụng
        App.Start();

        Console.ReadLine();

        // Kết thúc Ứn dụng
        App.Stop();
}
```

Lớp Grid Application

```
// Hàm này thực hiện ThreadFinish
private static void App_ThreadFinish(GThread thread)
{
    // Dẫn vào Gthread với lớp PrimeNumberChecker
    PrimeNumberChecker pnc = (PrimeNumberChecker)thread;

    // Kiểm tra xem số nguyên tố hay không
    bool prime = false;

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

    // Đưa ra kết quả
    Console.WriteLine("Number {0} prime? {1} ({2} convention)",
                      pnc.Candidate,
                      prime,
                      pnc.Factors);
}
```

So sánh một vài dự án

	Alchemi	Condor	SETI@home	Entropia	XtermWeb	Grid MP
Kiếnn trúc	Phân cấp		Tập trung			
Khả năng tích hợp mạng lưới Global	+	-	-	-	-	-
Kỹ thuật lập trình	C#, web service	C	C++, Win32	C++, Win32	Java, Linux	C++, Win32
Hỗ trợ Multi Cluster	+	+	-	-	-	+
Global Broker Resource	+ (Gridbus Broker)	+ (Condor-G)	-	-	-	-
Đa mô hình lập trình	+	-	-	-	-	-
Tích hợp ứng dụng	Cấp thấp		Không có	Cấp thấp		