【论文摘要】

校园计算网格建设与管理研究

导　师：罗省贤 教授
研究生：段林涛
专　业：计算机应用技术
方　向：分布式系统与网络并行计算技术

摘要

网格的发展和广泛应用使得各种计算资源得到了更大程度的利用，结合成都理工大学计算资源的现状以及学校科研、教学对高性能计算的迫切需要，本文依据建设校园网格及其管理的需求，通过对现有校园网格模型的研究以及计算资源的特性分析，设计与实现了基于 Globus 中间件的校园计算网格架构。

校园计算网格架构的设计和实现，不仅可以使得校园内的计算资源得到充分的利用，扩大一些昂贵且不可复制资源的用户群，还能够消除资源孤岛和信息孤岛，为用户提供一个安全、可靠并可扩展的 Web 应用环境。任何通过身份验证的用户都可以在任何一台接入 Internet 的计算机上由浏览器登录到校园计算网格应用平台，提交串行或并行作业，并可通过作业监控器和资源发现器及时监测和管理作业的执行。

为了达到既有效地整合计算资源，又方便用户的安全使用，校园网格设计为三层模型，资源层、网格服务层和用户门户层。本文通过使用 Sun 网格引擎（SGE）整合资源层，设计作业调度中心实现网格服务层，采用 Jsp/Servelt 实现用户门户层，提出了一套校园计算网格建设的实施方案，并通过应用实例给出了一个实用校园计算网格环境的实现。

关键词：校园计算网格；WSRF；SGE；作业调度；网格门户

【Abstract】

Research on Constructing and Management of University Computing Grid

Advisor: prof. Luo Shengxian

Candidate: Duan Lintao

Speciality: Distributing System and Network Computing

Major: Computer Application and Technology

ABSTRACT

The development and wider application of grid makes kinds of computing resources utilized in a deep degree. Considering status quo of computing resources in ChengDu University of Technology and the urgent needs of high-performance computing on school research, teaching in this campus, we put forward the demand of building university computing grid. In this paper, the architecture of university computing grid based on globus middleware is designed and implemented through the research on the existing university grid model and analysis of the characteristices of CDUT's resources.

University computing grid in the design and realization of the architecture, not only makes campus computing resources are fully utilized, makes user group who use expensive resources which can not be replicated expanded, but also to eliminiate information and resources isolated island and to provide users with a safe, reliable, extensible web application environment. Any user through the identification can access the Internet at any one computer can submit their serial or parallel jobs through our university computing grid application platform. Through job monitoring module and resource management and discovery module, grid user can inspect and manage their jobs duly.

To achieve both effective integration of resources and the safe use, we designed university computing grid into three-tier model, resource layer, grid services layer and user portal layer. This paper uses SGE to integrate resources, designs job scheduler to implement grid services layer, use Jsp/Servelt to achieve user portal. We give the university grid-building programme and through the use of example given a grid envirionment to achieve.

Keywords: University Computing Grid; WSRF; SGE; Job Scheduler; User Portal

第1章 引言

1.1 选题依据

网格的出现为人们充分利用资源提供了更好更新的解决方案。网格的核心就是突破了以往人们在使用资源上的种种限制，这些限制包括：

（1）计算能力大小的限制。以前大部分的用户无法得到足够的计算能力，因此许多问题的解决不能通过计算或者不能完全靠计算来实现，对模型以及算法的化简是最常见的近似方法。而网格所提供的计算能力足以满足其计算需求，在这种计算能力的支持下，人们可以做许多以前无法想象和无法完成的工作。利用网格人们可以将复杂计算交给网格计算环境，让网格整合的计算资源来处理。

（2）地理位置的限制。计算资源分布在各处，有些资源是昂贵且不可复制的，有些资源甚至无法和特定的地理位置分开，因此要使用这些资源，在以前许多情况下必须到相应的地方去，这在很大程度上限制了这些资源的使用。而网格把“到资源所在地”对资源进行使用的限制打破，对资源的使用与使用者所在的位置以及资源所在的位置无关，突破了在使用资源时对位置的限制。

（3）传统的共享和协作方面的限制。以前对资源的共享仅限于数据文件传输的层次，而网格资源的共享允许直接控制其他资源，并且共享资源的各方在协作时可以以多种方式更广泛地交流信息，充分利用网格提供的各种功能。网格的共享、协作方式和方法更广泛了，而且为这种合作提供了各种控制策略与手段，可以根据需要动态地与不同的组织或个人建立各种级别的工作关系。

（4）网格还打破了人们使用一些计算资源门槛过高的限制。我们有一些复杂的计算任务需要用到大型的计算资源以便获得足够的计算能力，但是使用这类资源往往难度比较大，门槛较高，而且这类资源价格昂贵，一般用户很难能够直接接触，但是通过网格使用者不需要直接接触高性能计算机就能得到它的高性能服务，网格应用向使用者提供了一个友好的界面，用户通过这个接口直接向网格环境递交计算任务，而不需要了解底层资源的构成和分布。

上述优越性清楚地体现了网格应用的重要作用和地位。从资源管理的角度来看，网格应用为用户有效管理了底层资源，使得底层资源能够作为一个统一体向外部世界提供计算服务。从网格用户的角度来看，网格应用将底层资源进行封装后透明地提供给网格用户使用，使得用户在不需要了解具体资源细节的情况下就可以获得资源提供的服务，为用户使用资源提供了一个良好的平台和接口。

目前成都理工大学存在着这样的一个现实，学校发展和建设了地学类、工程类、能源类等不少需求高性能计算的专业，这些相关专业存在着大量的复杂性高、计算量大的教学及科研计算任务，而单台独立的微型机甚至工作站无法提供这样的计算能力，为满足计算需求学校购进了曙光等高性能计算机，但是这类设备价格昂贵且不可复制，使用门槛高，所以利用率一直不高。因此开发校园计算网格应用来帮助师生更好的使用资源，提高这些资源的利用率势在必行。本文致力于探索和研究校园计算网格模型，希望通过对校园计算网格的研究能够有效地整合各类底层的计算资源，将这些资源利用校园网格平台透明地提供给校园内广大师生使用，为用户提供一个安全、可靠、高效、易用的校园计算网格平台。

1.2 论文研究内容及创新点

网格技术的研究开始于美国，目前关于网格的研究已经从美国和欧洲推广到了世界很多国家。英国政府已经投入超过 1 亿英镑的资金来建设英国国家网格；亚洲的日本、泰国、韩国、马来西亚等国也开始了网格的研发工作。在我国，网格的研究已经列入“863 计划”，中国科学院计算技术研究所从 1996 年开始了网格技术的研究工作。2000年开发了连接国内 8 个曙光计算中心的网格，并且在 2001 年提出了织女星网格计划。

网格计算的这种迅猛发展势头都应该归功于它优势，网格技术能够更容易的结合各种计算资源，透明地提供给用户使用，特别是 Globus Toolkit 4 的推出更让网格计算得到了快速发展，GT4 实现了 WSRF 规范，使得网格技术和 Web Service 真正地走在了一起。程序设计人员如果曾经参与过 Web Service 项目的开发，那么就可以很容易地参与到网格计算的开发工作中来。因此在短短的几年间，国内外就出现了很多网格研究项目，同时网格的名词也出现在很多的领域中，例如校园网格、电子商务网格、视频网格等等。在高等教育这个领域，目前我国已经有一些大学组建了自己的校园网格环境，如清华大学、山东大学、华中科技大学和河海大学等等。本文在总结前人研究成果的基础上，利用 GT4 开发环境设计了一个基于 Web 的校园计算网格模型，并使用现有的 Web 技术实现了这个网格模型。

1.2.1 论文的主要研究内容

本文通过对校园计算网格体系结构的研究和具体实施，实现了一个通用的校园计算网格应用平台，通过这个应用平台，管理员可以方便、高效地管理用户、作业和资源，而用户则可以安全、可靠地向校园计算网格平台提交串行和

并行作业。论文研究的主要内容如下：

（1）研究异构资源环境的整合方法以及向用户提供一个统一访问方式的接口技术；

（2）中间件环境的搭建、配置和测试方法；

（3）校园计算网格层次模型的分析和设计；

（4）校园计算网格体系中网格服务层的设计和实现方法，包括作业调度中心的设计、等待处理作业队列的设计等；

（5）校园计算网格体系中用户门户的展现形式，包括用户管理、作业管理、资源管理、作业状态监控、系统服务管理以及作业提交的方法；

（6）网格服务层中作业调度算法的初步研究和实现方案；

（7）网格安全体系的研究；

1.2.2 论文创新点

从分层网格模型的研究到安全校园计算网格应用平台的实现，本文对网格体系结构、网格系统组成以及相关技术进行了深入研究，论文主要的创新点如下：

（1）考虑资源环境的差异性，通过相关资源管理工具整合异构资源，为用户提供统一的应用接口；

（2）使用 Myproxy 在线证书仓库委托用户证书，真正实现用户的单点登录，并保证用户能够方便、安全与高效地使用网格资源；

（3）设计与实现了作业调度中心，根据优先级高者优先调度的原则协调网格层作业调度；

（4）设计的网格管理员门户可为系统管理员提供方便操作的管理平台，使管理员摆脱了受地理位置限制的资源服务操作，管理员可以通过浏览器对资源和各种服务实施在线控制；

（5）终端网格用户门户可为用户提供方便实用的资源使用平台。

第 2 章 网格及其关键技术

2.1 网格计算

网络是把地理上分散的计算机系统通过网络设备连接在一起，相互独立的计算机系统之间在遵循一定的通信协议的基础上实现资源共享。现代网络大致经历了三次发展过程，从基于简单设备的相连，到基于 WEB 的资源共享，再到更大范围内实现资源的共享（即网格技术）。通过网格这种基础设施，用户不需要了解网格环境的具体资源细节，就可以在网格环境中使用各种资源提供的计算能力，完成相关的计算任务。

网格的目标是要让加入到网格中的用户能够容易地访问网格资源。在这种网格平台上，用户不需要使用远程登录（Telnet）、文件传输协议（FTP）等工具就可以使用远程节点计算资源。现在这些计算资源主要是指一些 PC 的资源节点、计算机集群环境、高性能计算机节点和各种高性能的服务器。

用户在向网格环境提交作业或请求的时候，由网格管理系统来分配资源并控制其作业运行和资源使用，包括把这些作业分配到哪些计算机上运行，作业提交之后，状态如何收集，结果返回到哪台计算机，用户如何获得返回结果等等。这些工作对用户来说是透明的，用户使用网格中的各种资源时，不需要关心是分布在哪个地域的哪台计算机为自己提供了服务。总之，网格的目的就是不分地理位置的远近、不管用户提交作业的类型和复杂程度，为用户提供一个统一且简单的共享网格资源的接口。随着网格技术和 Web 服务的结合，网格技术具有以下的一些特点：

（1）以成熟的网络技术、计算机技术、通信技术、电器技术为基础；

（2）资源接入简单，不管是超级计算机、集群系统还是单个的 PC 计算池都可以很容易地被接入到网格中，使得网格的扩展性很强；

（3）网格为用户提供一个比它现有资源更强的计算力，但资源层的细节对用户透明。

根据网格客体对象的不同，可以把网格分为数据网格、计算网格和服务网格。数据网格中共享的基本单位是数据，主要解决数据的共享问题；计算网格中共享的基本单位是计算资源，计算网格为用户提供共享资源的良好接口和机制；服务网格中共享的对象是服务，以服务的形式提供共享的手段。校园计算网格以给用户提供计算资源的计算力为目的，网格用户可利用这些计算资源为自己解决大型计算任务问题。

2.2 网格体系结构

网格可以被划分为三个基本的层次：资源层、网格层和用户应用层（如图 2-1 所示）。资源层是网格资源的集合，也就是我们建立网格的基础。顶层是网格的应用层，各种各样的应用都在这一层实现。网格层处在资源层和用户应用层之间，主要作用是把用户和资源连接起来，为用户提供使用资源的统一接口。

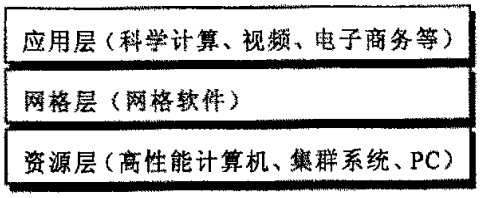

图 2-1 网格层次结构图

2.3 开放网格服务结构 OGSA

2.3.1 有状态服务模型

Globus 小组和 IBM 于 2002 年初提出了一个新的结构，该结构是要将当时网格领域最热门的两大技术，计算网格和万维网服务结合起来，把原来按照两条路线进行的研究活动归纳到一条主线上来，这就是面向服务的开放网格服务结构。

开放网格服务结构（Open Grid Services Architecture，OGSA）把 Globus 标准与面向商业应用的万维网服务结合起来，把网格计算从科学与工程计算应用扩展到了更广泛的以分布式系统服务集成为主要特征的商业应用领域，建立网格服务的基本概念。OGSA 采用万维网服务的 WSDL 和 SOAP 规范。遵循 OGSA 标准的系统都可以连在一起，用户可以很容易地集成、共享各种系统提供的功能，可以节省用户的开发成本，提高开发效率。

OGSA 以服务为中心，把一切都抽象为服务，服务既包括计算机设备、应用程序、数据，也包括仪器、设备等。将一切都抽象为服务有利于通过统一的标准接口来管理和共享网格上功能各异的资源。Web 服务一般面对的都是永久性的服务，在 Internet 这种比较松散联合的环境中，Web 服务无疑是可行的。在网格应用环境中，资源之间的联系更加紧密，但会存在大量的临时性短暂服务。鉴于网格具有不同于万维网环境的特点，OGSA 在原来 Web 服务的基础上，提出了网格服务的概念，用于解决服务发现、动态服务创建、服务生命周期管理等与临时服务有关的问题。

OGSA 把整个网格看成是一个网格服务的集合，这个集合的动态性很强，

是可以扩展的，体现了网格的动态特性。OGSA 把网格中的所有的资源都包装成服务，把网格中的各种资源的异构性隐藏起来，用服务这种统一的实体提供共享的接口。网格服务是一种特殊的万维网服务，它提供一组遵循网格服务规范的接口。通过接口提供服务发现、动态服务创建、服务生命周期管理、消息订阅、通知发送等功能。网格服务的标准接口不依赖于具体的实现和运行环境。一个网格服务可以部署在不同的运行环境中，人们常常把网格的运行环境叫做容器。不同的容器包含不同的软件环境、不同的机器型号、不同的操作系统等，甚至也可以是不同的设备。OGSA 还提供了一种网格安全机制来确保服务间所有的通信都是安全的。网格服务可以创建服务实例，人们可以通过访问特定的服务实例得到服务。

网格服务实例具有生命周期，服务工厂创建网格服务实例时可以指定服务实例的生命周期，有效生命周期中的服务实例可以被显式地终止，从而提前结束服务实例的生命周期。如果服务实例存活期超过了生命周期，而又没有得到任何用户发送的延长截止期限的确认消息或者保持存活的消息，系统将自动撤消该服务实例。如果某个用户在服务实例的生命周期结束前发送了重新确认的消息，或者发送了保持存活的消息，服务实例在预定的生命周期结束后还能够继续存活。

2.3.2 OGSA 的支撑技术

一个网格应用程序将经常包含一个不同的组件。比如，一个典型的网格应用程序包括：

①虚拟组织管理服务(VO Management Service)：管理每一个虚拟组织的节点和用户。

②资源发现和管理服务(Resource Discovery and Management Service)：通过MDS，网格上的应用程序就可以发现它们需要的资源，并且管理这些资源。

③作业管理服务(Job Management Service)：通过这个服务，网格用户可以通过 Job 的形式来提交任务到网格。

除了上面三部分，还有作为整个环境的其他分支，如网格安全和数据管理等。所有的这些服务是相互作用的，例如作业管理服务依赖于资源发现与作业相匹配的计算资源。为了解决这些服务之间因存在关联和相互作用而可能会产生的一些混乱，于是标准化就成为了必然，网格为每一种服务类型定义一个通用的接口。而被 GGF（The Global Grid Forum）开发的 OGSA(Open Grid Services Architecture)，目的就是为了基于网格应用程序定义一个通用的标准的和开放的架构。

OGSA 的目标是标准化所有的服务，为这些服务指定一组标准接口。而实现 OGSA 这个标准的技术支持是 WSRF（Web Service Resource Frame）和 Web Service。其中 WSRF 规范实现了有状态的服务，扩展了无状态的 Web Service，而有状态的 Web Service 是 OGSA 的实现根基。OGSA、WSRF 和 Web Service 之间的关系如图 2-2 所示。

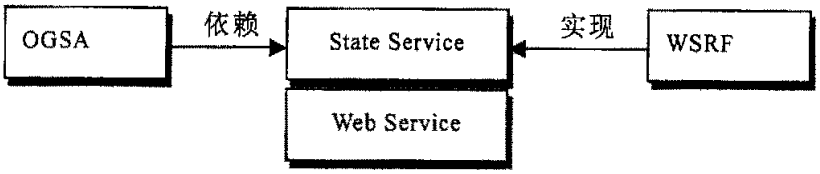

图 2-2 OGSA、WSRF 和 Web Service 之间的关系图

第 3 章 网格技术与 Web Service 的结合

现在网格技术被越来越多的人所掌握，并且在许多领域得到了应用，其中一个重要原因就是网格技术继承了原来被广泛接受的技术，而和 Web Service 的融合就是这个原因的重要体现。Web Service 是由服务组件通过 HTTP、SOAP、WSDL 等网络协议提供的远程调用接口，它所有的机制都完全基于现有的技术。

3.1 Web Service 体系结构

尽管 CORBA，RMI 和 DCOM 等技术早已得到了应用，但是这些系统都存在一些本质的问题，它们没有完全实现和平台无关的目标，也不能自由地穿越防火墙，不能实现真正的互连互通。为了达到和平台无关的目标，实现真正的互连互通，国际上一些大的计算机厂家和公司推出了 Web Service 框架。Web Service 是面向商业应用的一种框架，有关组织已经发布了一些实现 Web Service 所需要的协议和标准，如 SOAP，WSDL，UDDI 等。世界范围内的多家一流计算机公司都相继提出了自己的方案，推出了支持 Web Service 开发和应用的产品，主要产品有 IBM 公司的 WebSphere，微软公司的.Net，Sun 公司的 Sun ONE 等。

Web Service 的结构如图 3-1 所示，其中服务方法模块中一般包括多于一个的 Web 服务。例如，其中的发现服务允许程序设计人员从 Web 服务的集合中定位一个具体的服务。服务描述模块是通过 WSDL 来描述服务，并且告诉外部世界它能提供什么操作，以及外部世界如何调用它。服务调用模块主要是负责使用 SOAP 协议调用一个 Web 服务，包括在客户、服务器之间传输信息，SOAP（Simple Object Access Protocol）说明客户将如何请求服务器，并且如何做出响应。传输模块是通过 HTTP（HyperText Transfer Protocol）协议在服务器和客户程序之间传递信息。

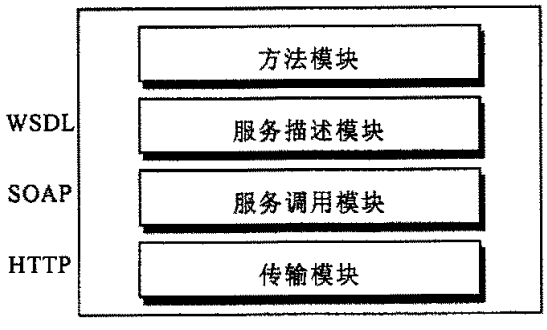

图 3-1 Web Service 体系结构

Web 服务只有发布后才能被检索到，因为服务检索依赖于服务的发布。在

这个层次不同的检索机制和发布机制是相对应的。任何使得服务请求者可以访问服务描述并且在运行时使得其可以为应用所有的机制称为服务检索。从图 3-1 可知，无论是 HTTP 网络协议、还是 SOAP、WSDL 等基于 XML 而定义的协议都是被广泛接受的技术。

3.1.1 扩展标记语言 XML

在 Web Services 中，不同层次的信息都使用可扩展标记语言 XML（eXtensible Markup Language）统一描述。XML 语言定义了结构化描述信息的标准格式，使数据在不同平台、不同系统之间可以使用不同的编程语言来实现互操作。XML 和 Web Services 的关系犹如 TCP/IP 和 Internet 的关系，不过后者着重于网络的互联，而 XML 着重于数据的共享。XML 在四五年中逐渐丰富和完善，已经成为了独立完整的知识体系，在后面的设计中本文会使用到其中的文档对象模型 DOM(Document Object Model)，它规定了 XML 编程的 API。

3.1.2 简单对象访问协议 SOAP

SOAP 是在分布式环境中通过 XML 编码进行通信的一种简单的网络协议，SOAP 协议从本质上讲是一种应用层协议，然而它往往是基于某些应用层协议来实现的，比如 SOAP 协议可以基于 HTTP 协议，可以基于 SMTP 协议，所以也可以把 SOAP 协议看作是应用层协议之上的第五层协议（其它四层协议分别是数据链路层、网络层、传输层及应用层协议），通过 SOAP 协议可以在不同的服务之间实现通信和数据共享，而不用关心服务用什么语言编写，在什么平台上运行，也不用关心服务是采用什么技术来实现的。

3.1.3 Web Service 描述语言 WSDL

WSDL 用来描述 Web Services 的 XML 语法。使用 WSDL，用户可以定位 Web 服务并调用它的任何公共函数，并且通过使用有关工具，可以使这一过程自动化，使应用程序使用很少的或不需使用手写代码就能集成新的服务。WSDL 提供了一种用以描述服务的普通的语言，同时提供了一个自动集成这些服务的平台。

3.1.4 统一描述、发现和集成 UDDI

UDDI(Universal Description, Discovery and Integration)用来描述、发现和集成 Web 服务的一系列技术规范，它是 Web 服务协议栈的一个关键部分，借助

于 UUDI，我们就能够发布并搜索 Web 服务，从而大大降低发布信息和寻找服务的成本，提高企业运营的效率。

3.2 Web 服务的定位和调用

客户应用程序要调用一个 Web service 的时候，程序员会使用到桩。桩(stub)分为两种，客户桩（Client Stubs）是在客户端的一段代码，作用是形成 SOAP 请求和解释来自于服务端的 SOAP 响应。服务桩（Server Stubs）是在服务端的一段代码，它的作用是解释由客户端发送来的 SOAP 请求和产生 SOAP 响应。桩的使用简化了应用程序，程序员没有必要写复杂的客户端程序，来动态的生成 SOAP 请求和解释 SOAP 响应，而只需要关心客户端/服务端的代码编写，其他的工作都留给桩来处理。

一个典型的 Web Service 的调用过程如图 3-2 所示。

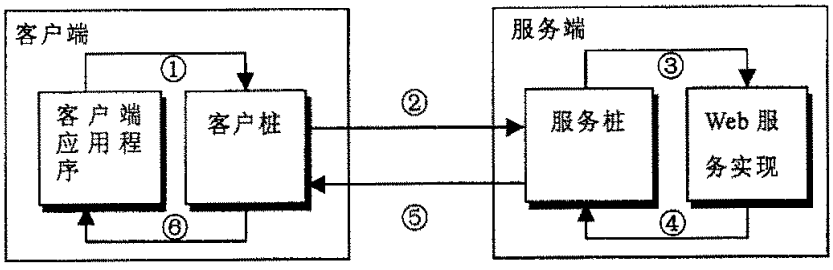

图 3-2 Web Service 调用图

由图可知，Web Service调用过程可以分成以下6个步骤：

①无论什么时候客户应用程序需要调用这个Web服务，它将去调用客户桩，客户桩将本地调用转换成SOAP请求(SOAP Request)；

②SOAP请求通过网络使用HTTP协议传送到服务端，服务端接受到这个SOAP请求后递交给服务桩，服务桩将这个请求转换为Web服务的实现能理解的描述；

③一旦SOAP请求被解释后，服务桩就调用服务实现，执行相应的操作；

④操作执行完毕后，请求操作的结果被交递给服务桩，服务桩负责将这个结果转换成SOAP响应(SOAP Response)；

⑤SOAP响应通过网络使用HTTP协议传送到客户端，客户桩负责接受SOAP响应，并将其转换成客户应用程序能够理解的形式；

⑥最后应用程序接收到 Web 服务调用的结果。

3.3 WSRF 框架

Web 服务资源框架(WSRF)是 GT4 的核心，WRSF 规范通过加入许多的特

征，将Web服务做成有状态的服务。WSRF是网格和Web服务共同努力的结果，标志着网格技术从此和Web Service结合在一起。WSRF既满足Web服务的架构又提供了OGSA需要的状态服务（WSRF扩展了Web服务）。换句话说OGSA是一个体系，WSRF是这个体系结构的基础（根基）。

3.3.1 WSRF框架的概念

WSRF改善了WEB服务的几个方面，使得它们能够更好的提供给网格应用，其中的主要优点就是有状态资源。简单的Web服务通常是无状态的，也就说Web服务是不能记住上一次执行过的状态信息的。但是，网格应用程序一般来说是需要有状态的服务，所以程序设计者希望他们的Web服务能够保持状态信息。WSRF将服务和有状态的信息完全分离，然后将有状态的信息放入Web服务能够保持状态实体的资源中，每一个资源有一个唯一的标识，所以无论什么时候想要和一个Web服务进行有状态的交互，就可以方便地通知这个Web服务来使用这个能够保持状态的资源。

3.3.2 WSRF相关规范

（1）WS-Notification：WS-Notification是一个规范集合。虽然不是WSRF的一部分，但却有非常大关系。这个规范允许一个Web服务被配置成为一个通知生成器，并且一些客户应用程序就成了通知的订阅者。使用这个服务时，客户应用程序首先向Web服务订阅相关的主题信息，当Web服务的这些相关主题资源发生改变时，这个改变就会通知到所有的订阅者。

（2）WS-Addressing：WS-Addressing规范提供一个机制去定位Web服务，Web服务和其上的资源对称为WS-Resource，WS-Addressing规范将WS-Resource的定位称作端点引用EPR（Endpoint Reference）。

3.4 GT4工具包

3.4.1 GT4简介

Globus项目是目前国际上最有影响的与网格计算相关的项目之一。该项目发起于20世纪90年代中期，其最初的目的是希望把美国境内的各个高性能计算中心通过高性能网络连接起来，方便美国的大学和研究机构使用，提高高性能计算机的使用效率。随着对Globus项目的深入研究，针对它的应用目标也进一步扩展，希望通过Globus项目可以方便地对地理上分布的研究人员建立虚拟组织，进行跨学科的虚拟合作。目前，Globus项目把在商业计算领域中Web

Service 技术融合在一起，希望不仅对科学计算领域，还要对各种商业应用提供广泛的、基础性的网格环境支持，实现更方便的信息共享和互操作，从而对商业模式、人员工作方式和生活方式产生深远的影响。

Globus 开发出能在多种平台上运行的网格计算工具包（Globus Toolkit），能够用来帮助规划和组建大型的网格试验和应用平台，开发适合大型网格系统运行的大型应用平台。GT 软件工具包是 Globus 最重要的实践成果，从 1998 年推出 1.0 版，2002 年推出 2.0 版，2003 年初又推出 3.0 版，到目前基于开放网格服务标准的最新版本 GT4.0，GT 工具包已经取得了巨大的进步，虽然目前 GGF 的工作组仍然在努力的为这些服务类型定义标准，但是在这些标准出台之前，GT4 已经是现在事实上的网格标准。

程序设计者通过 GT 就可以开发基于网格的应用程序。在这个工具包中包括了许多构建网格应用程序的高层服务，这些服务满足了 OGSA 提出的许多标准。GT 提供的服务包括资源监控和发现服务，用于作业提交的执行服务，网格安全服务和数据管理服务。

GT4 中的大多数服务是在 WSRF 上实现的（在 GT 工具包还有一些服务称作 non-WS 组件，它们不是在 WSRF 基础上实现的），换句话说 GT4 是 WSRF 规范的一个完全实现，虽然 WSRF 实现部分在 GT4 中的所占比重不大，但却是工具包的一个非常重要的组成部分。

3.4.2 GT4 的结构

GT4 是由五大组件构成，分别是安全组件、数据管理组件、执行管理组件、信息服务组件和公共运行期组件。

五大组件的基本功能如下：

（1）公共运行期组件提供了一组被 WS 和 non-WS 需要的基础库和工具；

（2）安全组件基于网格安全的基础设施（GSI），能够确保通信的安全性；

（3）数据管理组件可以管理大量的数据集，能够提供分布式数据的定位、传输和管理，GT4 中基本的数据管理工具有：用于高性能和可靠数据传输的 GridFTP，用于管理多传输的 RFT(Reliable File Transfer Service)，以及为副本文件维持位置信息的 RLS(Replica Location Service)。

（4）信息服务组件中的 MDS（Monitoring and Discovery Services）用于监控和发现网格中的服务和资源，MDS 服务提供查询和订阅接口来描述资源。GT4 中的 MDS 包括 WS MDS 和 Pre-WS MDS。

WS MDS 是 MDS 的 WSRF 实现，它包括以下一些服务：

①索引服务（Index Service），索引服务从网格资源收集监控和发现信息，

并且在一个单独的位置发布这些信息。

②触发器服务（Trigger Service），在网格上的资源上收集数据，如果和管理员定义的规则相匹配，该服务能够执行多种操作。

③集合框架（Aggregator Framework），WS MDS 服务建立在这个软件框架中。

WebMDS 能够让终端用户不需要安装额外的软件就能通过标准的 Web 浏览器查看监控信息。

Pre-WS MDS 是在 GT2 中被引入的，在 GT4 中为了向下兼容于是保留了 MDS2，但是现在已不提倡使用该组件，估计在 GT 的后续版本会逐渐被取消。

（5）执行管理组件负责用户提交作业的监控、管理、调度和执行。

尽管 GT4 实现了 Web 服务，但工具包也包含了一些没有实现 Web 服务的组件，如 GridFTP 组件使用的是非 Web 服务的协议。

3.4.3 GT4 中服务的使用

GT4 中提供了不少的服务，每个服务都包括服务提供者、服务、资源、服务的端点引用（EPR）以及对这个服务提供的操作，其中资源又包括多个资源属性。GT4 中服务的结构如图 3-3 所示。

我们要定位服务中的资源，直接通过资源对应的 EPR 就可以了。首先，创建一个端点引用（EndpointReferenceType）对象，用来作为这个服务的端点引用，从这个端点引用可以定位一个 WS 资源，接着可获得一个服务的端口类型（portType），这个端口类型是用一个桩类定位器（AddressingLocator）传给这个服务的 EPR 后返回的值，一旦有了这个服务的端口类型，就可以象使用一个本地的对象一样使用这个 Web 服务了。

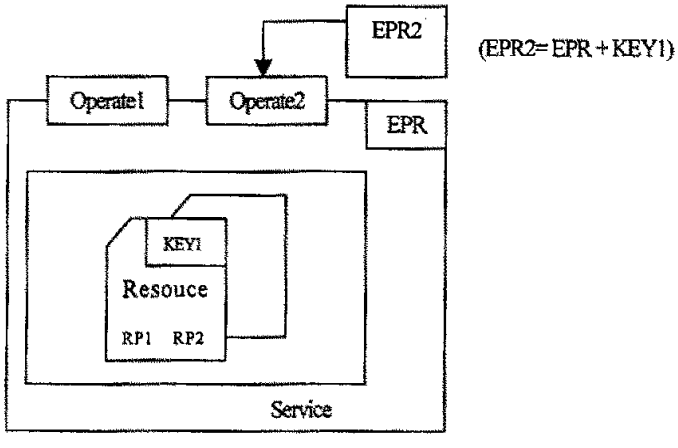

图 3-3 GT4 服务结构图

第 4 章 网格中的资源管理

考虑到计算资源的异构性和分散性，本文使用相关的资源管理工具将它们的这些特性隐藏，为用户提供使用资源的统一接口，屏蔽底层资源的复杂性。

4.1 资源管理软件 SGE

Sun 网格引擎（Sun Grid Engine, SGE）将网格定义为执行任务的计算资源的集合。对用户而言，网格是一个大系统，它提供单个入口点，从该入口点可以访问网格中强大而分散的资源。用户只需要将网格看做单个计算资源，资源管理软件接收由用户提交的作业，并根据资源管理策略安排作业的执行。用户可以一次提交数千个作业，而不必考虑它们在何处运行。

4.1.1 SGE 定义

SGE 从外部接受作业，将这些作业存放在等待队列中直至将它们送往执行设备，随后在执行过程中进行管理，最后在整个过程结束后将执行情况记录下来。SGE 将用以下方式调解可用资源和作业需求：

（1）通过 SGE 提交用户的作业需求概况。此外，SGE 还要检查用户身份及其项目或用户组的从属关系。用户提交作业的时间也将被存储起来；

（2）当确定队列可以对新作业执行操作时，SGE 就决定适合该队列的作业，并立即分派具有高优先级或等待时间最长的作业执行；

（3）SGE 队列允许同时执行多个作业。SGE 将尽量在负荷最小的队列中开始新的作业。

4.1.2 SGE 的特点

SGE 是数十种资源管理系统中功能最全面的软件，具有当前所有其它商业软件所不具备的优势，其功能包括：支持异构平台，运行的平台有 Digital , Intel , HP , SGI , IBM and Sun。支持的操作系统有 OSF/ Ultrix，Linux，UX， Irix， Aix SunOS 和 Solaris。支持进程迁移和负载平衡、具有容错功能、可以在最近的检查点处重新运行、支持 PVM，MPI 等并行环境。不需要附加软硬件，即没有额外的软硬件的要求，支持任务时序控制和优先级处理功能并具有动态资源池。所以本文在校园计算网格资源层设计和实现中直接使用 Sun Grid Engine（SGE）作为本地分布式资源管理软件。

使用 SGE6.0 对底层资源管理能够实现以下的功能：

（1）实现对本地资源的共享和统一调度，提高资源的利用率。

各种计算资源包括各种集群、服务器，甚至是普通的PC机也可以通过SGE进行管理，任何具有计算能力的机器都可以看作SGE中的一个执行节点，这样就能实现更大范围的资源共享。SGE通过资源匹配、资源预留（resource reservation）、资源回填（back filling）和改进的优先级算法等资源调度策略实现资源的统一调度和管理，实现负载均衡，提高资源的利用率。管理员可以根据不同用户的实际需求在SGE中配置和修改资源的调度策略。

（2）提交多种任务

用户可以方便地通过调用图形界面(Qmon)或以客户端命令的形式提交各种作业，包括批处理作业、并行作业（MPI与PVM）、阵列作业和交互式作业。用户提交批处理作业和并行作业时可以指定作业的资源需求、运行环境、作业开始运行时间以及指定运行结果返回路径。对于交互式作业，SGE根据授权用户提出的资源需求找到相匹配的节点机，并自动引导用户登录到该节点，用户操作该节点机就和在本地操作一样。

（3）作业监控和管理

SGE可以监控和管理所有作业的运行状态，给管理员提供删除、挂起、停止某个作业、对作业重新进行优先级设置和重新进行作业调度等功能。通过作业监控，就可以对不同的情况采用不同的管理策略。SGE支持作业的点检查功能。许多应用程序，尤其是那些通常消耗很多CPU时间的应用程序，已经运用了点检查和重新启动机制，以增强容错能力。状态信息和所处理数据的重要部分在算法的某些阶段被重复写入一个或多个文件。这些文件（称为重新启动文件）可以在应用程序中止后重新启动时处理，并达到与点检查之前一致的状态。SGE的点检查作业功能根据要求将某个作业中止并迁移到SGE中的其它计算节点中，从而以动态方式均衡集群中的负载。

（4）提高本地系统的安全性

SGE采用基于证书保密协议(CSP)加密的系统，在这种安全系统中，不再用纯文本方式传送信息，而是用密钥对信息进行加密再传送。密钥是通过公共密钥协议进行交换的。用户在SGE中出示可以证明其身份的证书，并从SGE接收确定通信对象是否正确的证书。完成初始的通信阶段后，通信将以加密的形式继续透明地进行。SGE中的会话仅在某个时期内有效，过期后必须重新认证才能继续会话。

（5）SGE与MPICH集成

SGE可以实现整个校园范围内的资源共享、资源调度和管理，但是要通过Internet实现更大范围的资源整合，就要使用GT4。目前GT4还不支持对SGE的接口，这样就必须通过适配器将SGE和GT4有机的结合在一起。为了支持并行程序在网格环境中运行，还需要安装和配置相应的并行软件，本文选择的

并行编程环境是目前最流行的消息传递标准 MPI 的实现 MPICH，通过安装和配置 MPICH 就可以在多台处理器上同时运行 MPI 程序。但是要在 SGE 中运行并行程序，还要将 SGE 和 MPI 相结合，进行相应的配置。SGE 提供了一个高级并行环境接口以便于与并行环境紧密集成，该接口将创建任务的职责从并行环境转交到 SGE。SGE 通过调用启动过程（利用 exec 系统调用）来启动并行环境。该启动过程的可执行文件名称以及传递给该可执行文件的参数，均可在 SGE 内配置。

4.1.3 SGE 的组成结构

在 SGE 系统中有四种基本的主机类型：主控主机、执行主机、管理主机和提交主机。

主控主机是SGE的中心节点。在主控主机上运行主控守护进程sge_qmaster和调度守护进程 sge_schedd。这两个守护进程控制着 SGE 的所有组件，包括队列和作业，并维护组件状态表和用户访问权限表等。默认情况下，主控主机还充当管理主机和提交主机。执行主机是有权执行 SGE 作业的节点机，该主机上有 SGE 队列，并运行 SGE 的执行守护进程 sge_execd。管理主机可以赋予执行主机权限，使之能够执行任何种类的 SGE 管理活动。提交主机仅允许提交和控制批处理作业，而且登录到提交主机的用户可以使用 qsub 提交作业，使用 qstat 控制作业的状态，并使用 SGE 图形用户界面 Qmon 来控制作业和管理队列。一台主机可能属于上述一个或多个类别。Sun Grid Engine 提供了 4 种守护进程，主控守护进程（sge_qmaster）、调度守护进程（sge_schedd）、执行守护进程（sge_execd）、通信守护进程（sge_commd）。它们之间的关系描述如下：

主控守护进程，是集群和调度活动的中心，它维护主机表、队列表、系统负荷表及用户权限表。它从 sge_schedd 接收调度指令，并从执行主机上请求 sge_execd 进行处理。调度守护进程在 sge_qmaster 的帮助下，维护集群状态的最新视图，指定将哪些作业分派到哪些队列，然后守护进程将这些调度指令转发至 sge_qmaster，后者将启动所需的操作。执行守护进程负责其主机上的队列，以及这些队列中的作业的执行。它会定期将信息转发给 sge_qmaster，通信守护进程通过公共的 TCP 端口进行通信，它用于 SGE 组件之间的所有通信。四个守护进程之间的协作关系如图 4-1 所示。

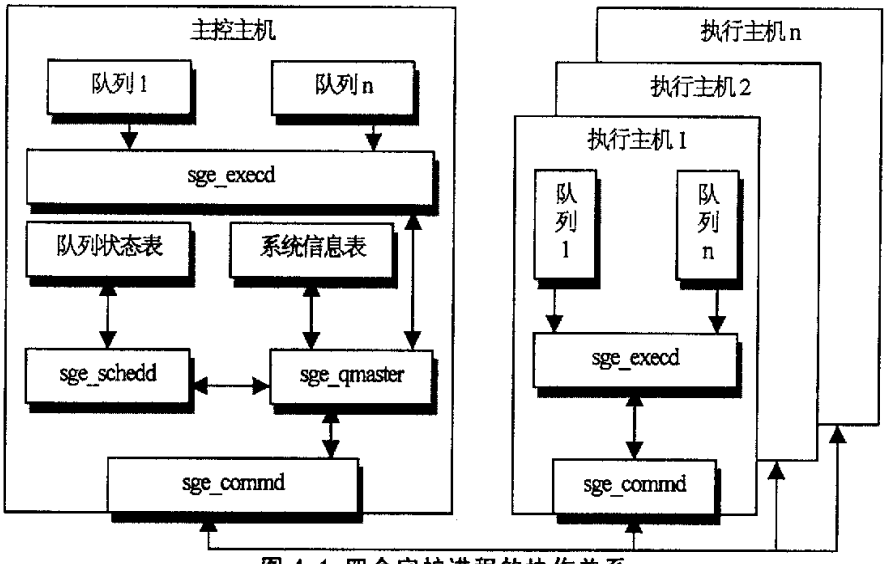

图 4-1 四个守护进程的协作关系

4.2 网格资源定位与分配组件 GRAM

GLOBUS 的资源管理结构主要包含中介者、协同分配器和 GRAM（GLOBUS 资源分配和管理）几个部分，它是应用和资源之间的桥梁。GRAM 的位置处在最靠近资源的地方，其主要功能是处理来自应用的资源请求，并满足应用提出的合理要求。协同分配器可以处理应用提出的对多个资源的请求，为应用协同分配分别属于不同本地资源管理者管理之下的资源，也就是可以满足跨资源的分配。

在 GLOBUS 资源管理器结构中，用资源描述语言（Resource Specification Language, RSL）书写的请求是连接各个部分的信息纽带。GLOBUS 支持的资源管理器有六种，PBS，EasyAll, Fork, LSF, Condor, NQE, SGE。GRAM 和本地资源管理器打交道，负责处理来自远程应用的资源请求，为请求资源的应用分配所请求的资源，并管理作业的执行过程。GRAM 也负责向 MDS 报告资源的有关信息。

GRAM 主要由门卫和作业管理器两部分构成。门卫位于资源一端，是该资源上所有远程请求的一个单一入口点，它负责用户认证，把远程用户映射到本地的一个安全环境中，让本地资源开始为其服务，它本质上可看做一个安全的网络守护进程 Inetd。 作业管理器是门卫的一个服务，门卫启动作业管理器的一个实例之后就把作业的管理和作业执行的整个过程都交给了作业管理器。门卫则可以在简单地处理应用请求之后快速返回，继续监听消息，为用户提供快速响应。作业管理器可直接与本地资源管理器系统连接，处理与作业的远程交

互。

GRAM API 可以实现作业的提交、作业的取消及查询已提交作业的状态。提交请求时，用户需要用 RSL 书写作业描述。GRAM 提供了多种类型的 API，分别适用于不同的用途。资源管理器客户端 API 为开始或结束作业的请求以及查询一个作业的状态提供接口。GRAM 提供的函数允许从客户端检查门卫是否在运行，以便请求开始一个作业，取消一个 GRAM 管理之下的作业，给作业管理器发送一个信号，查询一个作业的状态，为作业状态的改变注册一个回调联系，注销一个回调联系，给作业管理器提供一个新的授权代理证书等。

GRAM 为用户提供一个使用远程系统的简单接口，用户通过该接口可以在远程资源上执行作业，GRAM 最常用的功能就是作业提交和作业控制。GRAM 提供通用协议和 API，可以通过这些 API 请求和使用远程系统资源。GRAM 提供统一的可扩充的接口，连接到资源的本地操作系统，同时 GRAM 提供基于 GSI 的身份认证机制和映射 GSI 身份到本地帐号的机制，允许使用 WSDL/OGSI 客户端接口远程运行作业，实现作业的提交、监控和终止。

4.3 网格文件传输协议 GridFTP

Globus 借鉴了网络上已经被广泛使用的文件传输协议，在此基础上根据网格的特点和需求进行了扩展，形成网格文件传输协议（GridFTP）。

GridFTP 可以提供网格环境下安全传输，高效移动数据块的功能，满足网格计算环境不同的应用对广域范围分布的、大量的数据需求。它对 GSI 提供支持，并支持第三方控制的数据传输、并行数据传输、条状数据传输、部分文件传输、缓冲区大小自动协商、出错重传等。GridFTP 实现中增加了 GSS-API 安全认证，这样可以更可靠、更安全地支持第三方数据传输功能。

第 5 章 校园网格的系统设计

5.1 校园网格体系结构

根据网格的特点，将校园计算网格划分为三层结构：资源层，网格服务层和用户层。其中资源层是网格环境的物理资源部分，是系统的最底层，处在这一层的资源（比如超级计算机，集群系统和单一的 PC 机等）在资源管理软件 SGE 的统一配置之下整合起来，向外界提供统一的资源服务。SGE 是一个优秀资源管理软件，它和 GT4 通过适配器连接在一起。网格服务层是建立在 GT4 上的，通过 GT4 提供的接口向资源管理器提交作业并监听作业执行。用户层向用户提供了统一的 Web 登录界面，用户通过用户层，透过网格服务层来使用资源，同时也可以直接通过用户层来操作网格中间件及其底层的服务，校园计算网格体系结构层次如图 5-1 所示。

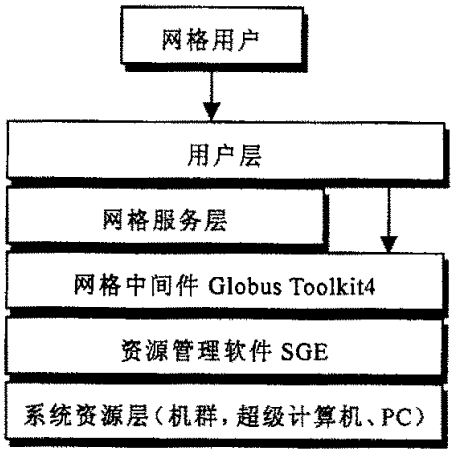

图 5-1 校园网格体系结构层次图

5.1.1 资源层设计

校园计算网格的最终目标是要实现学校资源的高效利用，将一些零散的计算机资源、集群系统和一些使用难度比较高而且价格昂贵的高性能计算机整合起来为教学和科研服务。资源层是整个校园计算网格系统的最底层，是网格服务层和用户层的物理基础，通过对资源层的规划和整合要到达的目的是消除信息孤岛、资源孤岛、提高系统整体性能、提高资源利用率。从用户的角度通过整合的资源为用户提供统一的接口，资源结构以及资源组成对用户透明，用户不需要关心是哪些资源在为自己提供服务，就能在有限的时间得到高效的信息服务。

本文对资源层的整合采用了 SGE 资源管理软件，并按照 SGE 和 Globus 的官方网站的技术规范进行资源层的设计和实施，具体细节参看第 6 章校园资源整合及环境搭建部分。

5.1.2 网格服务层设计

5.1.2.1 功能设计

校园计算网格服务层是连接用户层和资源层的接口，在设计的时候既要满足用户的需求又要能和资源层很好地结合，所以校园计算网格服务层应具有的功能如下：

（1）能够将用户提交给系统的作业分配到合适的计算资源上去执行；

（2）能够发现当前环境中可用计算资源以便系统有资源可以分配给作业；

（3）能够在批处理系统执行作业期间接受作业的状态变化信息；

（4）能够收集作业执行结果。

为了达到上述目的，本文设计了两个和 SGE 执行作业队列相对应的作业等待队列，分别是串行作业等待队列和 MPI 并行作业等待队列，用来接收用户提交的作业。同时设计了一个作业调度中心，该调度中心的作用是启动 MDS 资源发现线程，同时不断的轮询两个等待队列，一旦发现在等待队列中有等待处理的作业，就根据优先级高者优先调度的策略将它提取出来，同时在当前可用的计算资源中去寻找可以分配给该作业使用的最优资源（这个最优资源是指当前负载最低，可以执行的作业个数最多，同时与作业类型匹配的工厂服务资源），然后将作业提交到这个资源上，返回管理作业端口号，同时启动作业监听器，订阅作业状态主题，等待接收作业状态变化的通知。在作业调度中心中一共开启了两个线程，分别是 MDS 资源监控线程和作业监听线程。作业调度中心功能如图 5-2 所示。

其中 MDS 资源监控线程的作用是在索引服务中发现当前可以使用的资源，如果有资源突然关机或者有资源突然离线，通过 MDS 都能够得到及时地更新，使系统获得当前最新的资源信息，避免将作业提交到一个已经不可用的资源上去。而作业监听线程的主要目的是获得作业的 EPR（作业的端点引用），同时开启通知客户管理器，订阅作业状态主题，等待作业状态的变化，使得客户程序可以接收到这个通知，根据通知信息，网格系统能将作业状态信息的变化反映到用户页面上，使客户在提交作业之后及时了解作业的执行情况。

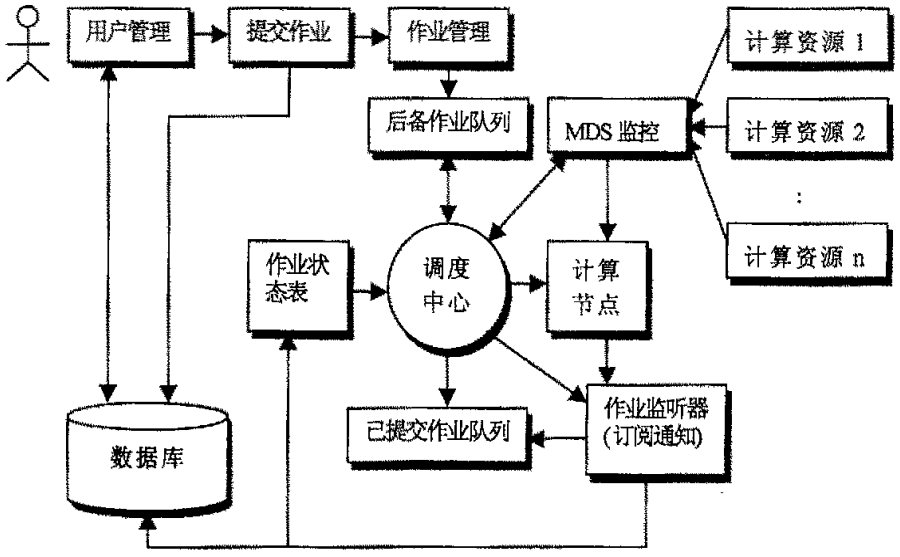

图 5-2 作业调度中心的功能结构

5.1.2.2 网格作业类型及其状态变化

校园计算网格环境的作业执行需经过两次调度，第一次是在网格服务层的调度，第二次是在资源层（SGE 队列）的调度。在本校园计算网格环境中，作业分为串行和并行两类，作业在用户提交到系统后的状态变化如图 5-3 所示。

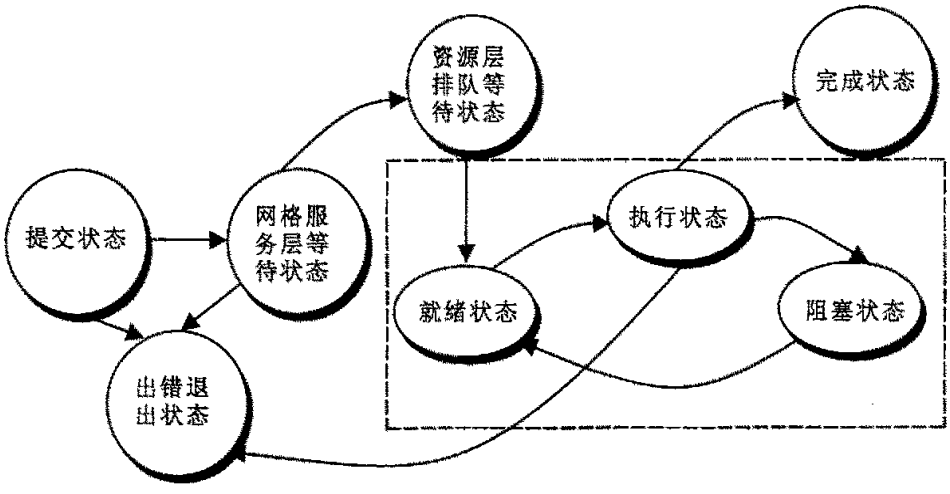

图 5-3 作业状态转化图

作业从客户端上传到服务器的过程中，处于提交状态，作业上传结束后，作业对象产生并被插入到网格服务层的等待队列，这时作业进入了网格服务层的等待阶段。根据一定的网格服务层作业调度策略，作业一旦被作业调度中心

选中就被分配到一个作业工厂资源，作业工厂资源再创建作业管理器通过GRAM将作业递交给作业批处理系统SGE，这时作业就进入SGE的等待队列等待调度。SGE调度进程在系统资源可用的情况下，根据一定的调度策略从后备队列中选择一个作业，为它分配资源，创建进程，新创建的进程就处于就绪状态，当执行主机上的CPU分配给了这些进程后，作业就进入了运行状态，如果进程在执行的过程中要进行相应的I/O操作或者等待一些临界资源，作业就从执行状态转为阻塞状态，当这些资源可用或I/O完成时，就再一次进入就绪状态，等待下一次获得CPU进入运行状态，在这个过程中，作业可以执行完成进入完成状态也可能中途出错进入出错退出状态。

作业一旦被提交给SGE批处理系统后，用户就很难和自己的作业进行交互，这是批处理系统的一个不足：交互能力弱。本文通过WS-Notification服务来订阅作业状态信息，让用户了解到作业的状态变化，并通过Web页面反映出来。提交到SGE后，客户程序接受到的状态通知类型如表5-1所示。

表5-1 状态通知类型

通知	说明
Unsubmitted	未提交成功
StageIn	分段输入
Pending	排队等待
Active	执行
Suspended	阻塞
StageOut	分段输出
CleanUp	清除
Done	成功完成
Failed	失败

5.1.2.3 用户作业调度模型设计

当作业被提交到等待队列后，应该将哪个作业分配到哪个计算资源，就取决于作业调度中心所采用的调度算法，作业调度算法的优劣直接影响到资源利用率、系统性能和用户的需求，所以调度中心的作业调度算法的选择和设计非常重要。目前存在的作业调度算法主要有：

1．先来先服务调度算法（FCFS）

这是一种最简单的调度算法，当采用这种算法时，调度中心每次从等待队列中选择一个最先进入该队列的作业，为它分配工厂资源。FCFS算法看上去对各个作业都很公平，按照先来后到的次序决定调度的先后次序，但是它对一

些执行时间比较短，要求比较紧迫，优先级比较高的作业却不利。

2．短作业优先调度算法（SJF）

这种调度算法是指对短作业优先调度的算法。SJF 算法可以照顾到实际上在所有作业中占很大比例的短作业，使它们能比长作业优先执行，该调度算法每次从等待队列中选择一个估计运行时间最短的作业，为它分配工厂资源，从实践来看短作业优先调度算法能有效的降低作业的平均等待时间和提高系统的效率，但是 SJF 也存在着不容忽视的缺陷：

（1）不利于执行时间长的作业，有可能造成长作业得不到调度；

（2）没有考虑作业的紧迫程度，不能保证紧迫性作业会得到及时的处理；

（3）作业的执行时间是根据用户所提供的估计执行时间来确定的，而用户有可能有意或无意地缩短其作业的估计执行时间，使得 SJF 算法不一定能真正做到短作业优先调度。

3．响应比高者优先调度算法

作业的响应比是作业响应的时间与执行时间的一个比率，当作业在等待队列中等待时间相同时，要求执行的时间越短的作业，其响应比也就越高，这样就有利于短作业，相反，如果作业要求的执行时间相同，等待时间越长的作业，其响应比也就越大，这时又实现了先来先服务。总之，这种调度算法既考虑了短作业，又考虑了作业到达的先后次序，不会使长作业长期得不到服务。因此该算法实现了一种较好的折中，但是这种算法在调度之前要计算响应比，增加了系统开销，同时执行时间也是一个比较难确定的参数。

4．优先级高者优先调度

这种调度算法主要是为了照顾紧迫型的作业，使用这种调度算法的系统每次从等待队列中选择优先级最高的作业，为它分配资源。其中优先级的高低是由优先权来决定的，优先权可以分为静态和动态两种。静态优先权在创建作业的时候就确定了，且在整个调度过程都将保持不变，而动态优先权是在作业创建后赋予的，可以随着作业调度的推进而动态地变化，比如作业随着在等待队列中等待时间的变化而影响优先权的动态变化。

综合以上几种作业调度算法，在网格服务层的作业调度中心使用了优先级高者优先调度的调度算法，调度中心每次从等待队列中选择优先级高的作业调度，当优先级相同时，选择先进入等待队列的作业，这样在调度中心中作业就有了轻重缓急之分，有效地改善了作业调度环境，在一定程度满足了用户的需求。作业优先权是在作业创建前确定的，优先权的大小是根据提交作业用户的条件决定的。优先权划分为 4 级，优先级编号为 0，1，2，3，数值越大优先权越低。在用户被批准为系统合法用户时，由管理员赋予用户级别，该级别在以后随着用户身份的变化可以动态变化。

5.1.3 用户层的设计

5.1.3.1 功能设计

用户层为网格用户使用网格服务提供了安全、方便高效的应用平台，考虑到用户、资源和作业及其相互关系，本文对用户层的功能划分如下：

（1）提供用户注册功能；

（2)提供身份鉴别功能,根据用户的不同类型(管理员或者普通网格用户)，提供不同的 Web 服务平台；

（3）为管理员提供作业管理、用户管理、计算资源管理、作业调度中心管理、在线用户的作业管理及系统服务管理等功能；

（4）为普通用户提供作业提交、作业管理、用户信息管理和资源查询等功能。

各个用户层的功能模块详细描述如下：

用户注册功能：该功能是系统为新加入用户提供的一个接口，新用户通过注册功能能申请一个帐号，但是为了系统使用的安全性，这个帐号在未得到管理员的授权时不能使用，当帐号对应的用户得到了认可，管理员为其授予一定的权限后才能生效，这时用户才能使用帐号登录网格系统，合法使用网格服务。

身份鉴别功能：系统将用户划分为三种类型：管理员、普通用户和非授权用户。这个功能可以根据用户类型不同提供不同的服务页面。

对于管理员，系统为其提供的功能如下：

（1）用户管理：用户管理分为申请用户管理和授权用户管理。其中申请用户管理可以让管理员为新申请的用户授权，使其成为合法用户。授权是指开通网格服务权限和确定优先级，为调度中心的调度策略提供依据。通过管理员授权的用户可以使用系统服务，提交作业。对授权用户的管理可以使管理员对合法的网格用户进行信息查询、权限类别以及优先级的更改、用户帐号的删除等。

（2）作业管理：用户提交作业后，作业执行情况都会被记录下来，同时执行的结果会被保存在 Gridftp 服务器上，该功能可以帮助管理员查看合法网格用户提交的作业信息，包括作业的编号、名称、所有者、类型、提交时间、完成时间、执行状态以及结果存放地址等等。同时提供下载作业结果，删除作业及相关目录等功能。

（3）工厂信息管理：GT4 实现了 WSRF 后，使得网格技术和 Web Service 相结合向外提供许多的服务，其中包括作业管理工厂服务(ManagedJobFactoryService)，管理员可以通过工厂信息管理来增加、更改和删除能够提供作业管理工厂服务的工厂信息。工厂信息可以通过 MDS 查询来及

时地更新，以保证当前工厂都为可用的资源。

（4）在线用户作业管理：通过在 web.xml 中增加监听类，管理员可以了解当前用户的情况、记录下用户提交的作业、查看当前在线用户以及他们提交的作业的执行情况，包括作业当前的执行状态，作业的提供者以及提交时间等。

（5）系统服务管理：管理员可以在任意接入网格环境的计算机上通过浏览器对系统服务（包括 MyProxy、Gridftp、PostgreSQL、Globus 容器）进行控制。

对于普通网格用户，系统主要提供作业提交功能。根据用户的不同需求，作业分为串行和并行作业两种类型，用户可以选择进入不同类型的作业提交页面，提交串行或并行作业，作业提交时序如图 5-4 所示，其操作步骤如下：

①在作业提交页面上输入作业信息（包括作业存放地址以及作业所需参数等），如果是并行作业还应给出进程个数。用户输入相应信息后点击提交按扭，系统便将用户作业从客户端上传到服务器。

②作业上传完毕后，通过 RSL 生成器形成 XML 文档类型的作业说明文件。在 XML 文件中，定义了执行文件的名称、作业的执行目录、作业输入输出文件存放地址、作业类型、作业分段输入、分段输出以及清除临时文件等标签元素。根据文件类型不同，形成的 XML 文件的形式是有所区别的。如果用户向系统提交了名为 cpi 的并行作业，并且要求启动 5 个进程并行执行，则产生的 XML 描述文件 RSL 如表 5-2 所示。作业的 XML 文件生成后，再通过 GT4 提供的 Java 类 RSLHelper 的 readRSL 方法转换成 WS-Gram 能够理解的 RSL 形式。

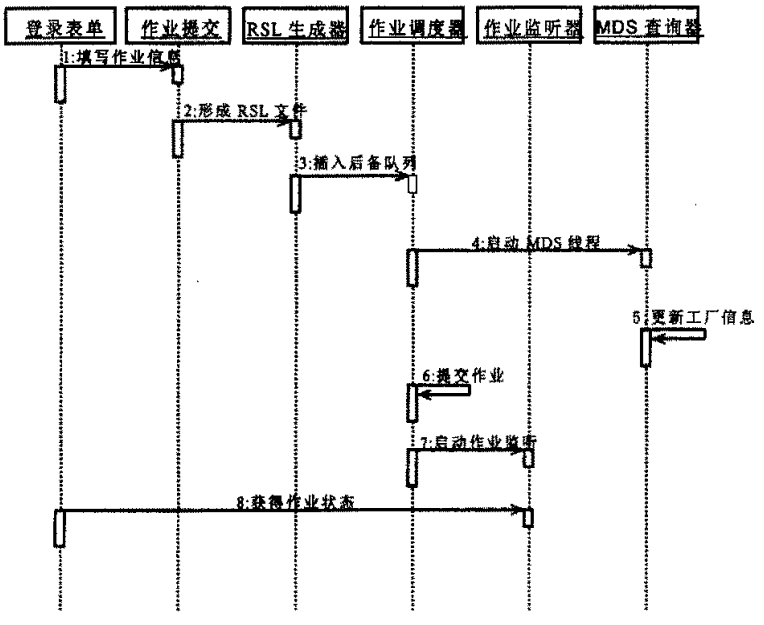

图 5-4 用户作业提交时序图

③作业描述文件 RSL 形成后，通过 UUIDGenFactory 类获得一个 128 位的

全球惟一字符串作为作业标识符，同时获取作业优先级，最后形成作业对象，并将这个新作业对象插入到相应的等待队列中等待调度。

④启动 MDS4 资源监控线程。

⑤MDS 启动后，会发现当前可用的工厂服务，并通过询问和测试的方法了解当前这些工厂资源的状态，系统通过 MDS 可以及时地获得可用资源，避免作业调度中心将作业递交给一些已经不可用的资源。

⑥根据等待队列中作业的优先级，作业调度中心从中选择一个优先权最高的作业，并且从当前的可用资源中选择一个与作业类型一致的最优资源，然后将这个作业递交给该工厂资源，并返回管理作业端口号，以便对作业进行进一步的控制。

⑦作业调度中心为该作业创建作业监听器，订阅作业状态信息。

⑧接受作业状态变化的通知，根据作业的不同状态，做出不同的处理，包括更新数据库信息，更新作业状态表，以及更新队列信息等，同时通过 JSP 将作业的状态变化及时反映到客户页面上，为用户提供一个良好的作业管理界面。

用户层功能结构如图 5-5 所示。

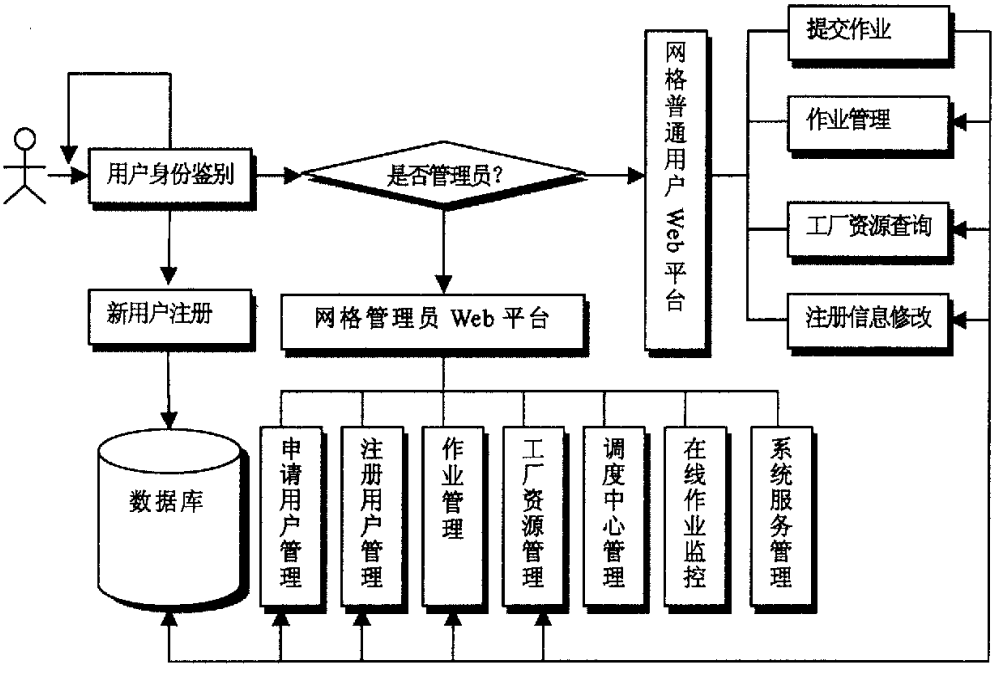

图 5-5 用户层各功能间的关系图

5.1.3.2 使用 MyProxy 代理用户证书

在 Globus 环境中，用户访问网络资源的服务是由代理完成的，而在校园网格中用户可以通过浏览器方便地访问网格资源，最开始的设计思路是将所有用

户的证书和私钥保存在一个固定的目录下面，当用户通过浏览器访问网格资源的时候，利用该目录下的用户证书和私钥产生临时代理，以后的认证工作便由此代理完成。但是在固定的目录下保存用户的证书和私钥很不安全，同时也不便于用户整合现有的网格安全系统（如网格安全设施 GSI），使得在网络安全机制上缺乏代理能力，所以本文使用一种在线的证书系统 MyProxy 来解决这个问题，MyProxy 主要是用来在不相容的 Web 安全和网格安全之间搭建桥梁，允许用户层在一种标准、安全的形式下使用 GSI 访问网格资源。

在用户层使用 MyProxy 代理的方法如图 5-6 所示，其代理过程如下：

（1）使用 MyProxy 系统将一个用户的代理证书委托给证书仓库；

（2）因为用户使用网格资源的时间和地点是不同的，所以他们可以通过 Web 浏览器或者配置文件向 Web 门户（Web Portal）提供自己的身份信息，同时指定所使用的 MyProxy 代理服务器的名字；

（3）接着用户层会使用 myproxy-get-delegation 程序去连接 MyProxy 代理服务器的证书仓库，提供用户认证信息（包括用户名和密码），为用户请求一个代理证书；

（4）证书仓库在确认了这些认证信息后，将为用户委托一个代理证书，并返回给用户层。用户层就可以成功地通过网格应用程序访问网格资源。

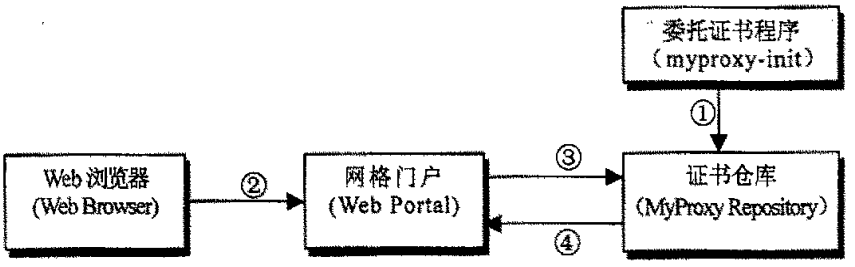

图 5-6 用户层的证书代理过程图

一旦用户从用户层注销或停止使用系统资源，就要从客户端删除用户的委托证书，如果用户忘记注销，那么证书会在指定生命周期到达的时候过期。如果证书过期，用户可以使用 myproxy-init 程序来为自己向仓库系统委托新的证书。证书的生命周期为一个星期。

5.1.3.3 作业描述文件

作业描述文件是用户在用户层形成的 XML 文件。其中<executable>元素指出执行机上的执行命令名，<directory>元素指出命令在执行机上执行的目录，而元素<stdout>指出执行结果在执行机上的标准输出路径，元素<stderr>指出执行结果的错误信息在执行机上的输出位置，<argument>元素说明命令执行过程

中要使用到的参数。

为了进行文件的分段传输，必须在作业描述文件中增加指定的元素。通过RFT语法中的文件传输指令来使用第三方传输。每个<transfer>元素必须指定一个源URL和一个目的URL，如果是远程文件URL被写成GridFTP URL的形式，如果是本地文件则被写成文件URL的形式。例如，分段传入一个文件，源URL是一个GridFTP URL（gsiftp://192.168.0.169:2811/uploaddir/cpi）分段传输到作业GRAM机的文件系统上（file:///executable dir/cpi）。在运行时，GRAM服务利用RFT服务在远程机器上采用GridFTP协议获得远程文件并可靠地写到指定的本地文件系统。

用户提交的作业在校园计算网格环境中为两类：串行作业和并行作业。对于串行作业，执行命令即为用户提交的命令，而并行作业需要通过mpirun命令来执行，要由参数给出待执行的并行程序名、进程数以及参与并行的各计算节点机名等。所以两种类型的作业描述文件有一定的差别，但是所涉及到的XML元素基本不变。

本文以表5-2所示的一个并行作业cpi对应的作业描述文件（david_cpi.xml）为例来说明各个元素的作用。元素<executable>指出执行机上可执行命令是/usr/local/mpich/bin/mpirun，元素<directory>给出执行目录是/executabledir/，命令执行过程所需要的参数是由元素<argument>指出，分别是-np、5、要执行的并行程序cpi以及由machinefile参数给出的参与并行计算的节点机器名文件，元素<stdout>指出执行的结果被重定向到/executabledir/davidcpi.stdout，执行完毕后的错误信息由<stderr>说明被写到执行主机/executabledir/davidcpi.stderr。在<fileStageIn>元素定义中，可执行文件/uploaddir/cpi从作业提交机分段传输到GRAM主机的本地文件系统，目的位置是/executabledir/cpi。分段传输完毕后，就排队等待执行作业，在元素<fileStageOut>的定义中，标准的输出结果通过分段传输从执行机送到作业提交机/downloaddir/davidcpi.stdout，而标准错误输出将送到提交机的/downloaddir/davidcpi.stderr。最后通过< fileCleanUp >元素从执行主机中清除传输阶段产生的临时文件/executabledir/davidcpi.stdout.1 和/executable dir/davidcpi.stderr.1。

表 5-2 cpi 的并行作业的 XML 文件（david_cpi.xml）

```
<?xml version="1.0" encoding="UTF-8"?>
<job>
<executable>/usr/local/mpich/bin/mpirun</executable>
<directory>/executabledir/</directory>
<argument>-np</argument>
<argument>5</argument>
<argument>/executabledir/cpi</argument>
<argument>-machinefile</argument>
<argument>/tmp/dlt/hosts.1</argument>
<stdout>/executabledir/davidcpi.stdout</stdout>
<stderr>/executabledir/davidcpi.stderr</stderr>
<fileStageIn>
    <transfer>
        <sourceUrl>gsiftp://192.168.0.169:2811/uploaddir/cpi</sourceUrl>
        <destinationUrl> gsiftp://192.168.0.169:2811/executabledir/cpi
        </destinationUrl>
    </transfer>
</fileStageIn>
<fileStageOut>
    <transfer>
        <sourceUrl> gsiftp://192.168.0.169:2811/executabledir/davidcpi.stdout.1
        </sourceUrl>
        <destinationUrl>gsiftp://192.168.0.169:2811/downloaddir/davidcpi.stdout
        </destinationUrl>
    </transfer>
    <transfer>
        <sourceUrl> gsiftp://192.168.0.169:2811/executabledir/davidcpi.stderr.1
        </sourceUrl>
        <destinationUrl>gsiftp://192.168.0.169:2811/downloaddir/davidcpi.stderr
        </destinationUrl>
    </transfer>
</fileStageOut>
<fileCleanUp>
    <deletion>
        <file> gsiftp://192.168.0.169:2811/executabledir/davidcpi.stdout.1</file>
    </deletion>
    <deletion>
        <file> gsiftp://192.168.0.169:2811/executabledir/davidcpi.stderr.1</file>
    </deletion>
</fileCleanUp>
</job>
```

5.2 数据库的设计

为了便于校园网格信息的管理和组织，数据库的设计也是非常重要的。在校园计算网格应用中为了高效地组织数据，设计了以下数据表，分别描述如下。

（1）用户表 usertable

用户表是用来记录网格用户的基本信息，如表 5-3 所示。

表 5-3 用户表

字段名	类型	初始值	主键	备注
Userid	int	NOT NULL	√	用户编号
Account	varchar(10)	NOT NULL		用户帐号
Realname	varchar(10)	NOT NULL		真实姓名
Emailaddr	varchar(30)	NOT NULL		email 地址
Certificate	varchar(30)	NOT NULL		证件编号
Registtime	varchar(30)			注册时间
Confirmtime	varchar(30)			批准时间
Lasttime	varchar(30)			上次登录时间
Priority	int	3		优先权

用户帐号和用户一一对应，email 地址是方便系统向用户发送确认信息和作业执行结果；证件编号方便管理员确认用户身份；优先权是作业调度的依据。

（2）作业表 jobtable

作业表主要是用来记录网格用户提交的作业信息如表 5-4 所示。

表 5-4 作业表

字段名	类型	初始值	主键	备注
Jobid	varchar(128)	NOT NULL	√	作业编号
Jobname	varchar(30)	NOT NULL		作业名
Owner	varchar(30)	NOT NULL		提交者
Submittime	varchar(30)			提交时间
Endtime	varchar(30)			结束时间
Jobstate	varchar(10)			作业状态
Jobtype	varchar(10)			作业类型
Stdout	varchar(256)			作业结果地址
Stderr	varchar(256)			错误信息地址
Factoryuri	varchar(30)			工厂地址

其中作业编号（jobid）是一个 128 位的全局惟一数值，与作业对象一一对应，而作业结果存放地址便于用户下载作业和向订阅作业结果的用户发送结果文件。

（3）工厂表 factory

工厂表记录提供 ManagedJobFactoryService 的网格节点，如表 5-5 所示。

表 5-5 工厂表

字段名	类型	初始值	主键	备注
factoryid	int	NOT NULL	√	工厂编号
factoryuri	varchar(30)	NOT NULL		工厂地址
factorytype	varchar(30)	NOT NULL		工厂类型
maxjobnumber	int	0		最大作业个数
completenum	int	0		已经执行作业

工厂表中记录的工厂信息是 MDS 查询线程更新工厂资源的基础，通过对工厂表的增加和删除，可以根据实际情况配置当前实际的工厂信息，和 MDS 配合可以得到当前可用的资源，这样就解决了资源的动态性的问题。

除此之外，为了便于系统的扩展，还可以增加其他的数据表，例如为了共享更多的其他资源增加资源表；为了增加计账功能引入用户资源使用登记表等等。

第 6 章 校园资源整合及环境搭建

6.1 SGE 的安装和配置

为了实现校园计算网格设计模型，搭建了校园计算网格实验环境。网格环境由三台计算机所组成，分别是 master.dncpc、david.dncpc 和 helen.dncpc。

安装 SGE 的过程：

1.规划安装环境，设置 master.dncpc 为主控主机，david.dncpc 和 helen.dncpc 为执行主机。

2．修改节点机的/etc/hosts 文件，为各节点机互访设置许可权限。

3．下载 SGE 需要用到的安装包 sge-6.0u6-common.tar.gz 和 sge-6.0u6-bin-lx24-x86.tar.gz，将它们存放到这三台计算机的/usr/sge_root 目录下，并且解压，等待执行安装程序。

4．在 master.dncpc 上安装主控主机的过程：

①在/etc/profile 文件中设置与 SGE 相关的环境变量，SGE 的环境变量的设置情况如表 6-1 所示；

②进入安装包解压目录执行 install_qmaster 安装文件；

③设置主控主机信息，包括管理帐号、安装目录、SGE_CELL、网络服务；

④信息设置完毕后，启动 sge_qmaster，sge_schedd 守护进程，主控机安装完成。

表 6-1 网格环境变量设置表

```
MPIHOME=/usr/local/mpich
JAVA_HOME=/usr/j2sdk
ANT_HOME=/usr/apache-ant
SGE_ROOT=/usr/sge_root/
SGE_CELL=default
GLOBUS_LOCATION=/home/globus/gt4
PATH=$PATH:$SGE_ROOT/bin:$JAVA_HOME/bin:$ANT_HOME/bin
PATH=$PATH:/usr/local/pgsql/bin:$GLOBUS_LOCATION/bin
export SGE_ROOT SGE_CELL MPIHOME JAVA_HOME ANT_HOME
export GLOBUS_LOCATION PATH
source $GLOBUS_LOCATION/etc/globus-user-env.sh
```

5．在 david.dncpc 和 helen.dncpc 上安装执行主机：

①将主控主机上的/usr/sge_root/default 目录拷贝到这两台主机的/usr/sge_root 下，以确保 david.dncpc 和 helen.dncpc 有统一个主控主机 master.dncpc；

②和主控主机的安装一样，需要在/etc/profile 文件中设置与 SGE 相关的环境变量，变量的设置情况如表 6-1 所示；

③进入安装包解压目录执行 install_execd 安装文件；

④设置执行主机信息，包括安装目录、 SGE_CELL、网络服务等；

⑤设置完毕后，启动 sge_execd 守护进程，执行主机安装结束。

6. 向系统中加入管理主机和提交主机。

主控主机在默认情况下就已经是管理主机和提交主机了，所以不需要单独设置，但其他的计算机是需要单独设置，可通过 qconf 命令将 david.dncpc 和 helen.dncpc 两台计算机设置为提交主机，以后就可以在命令行方式下通过 david.dncpc 和 helen.dncpc 两台计算机提交用户作业。其实，SGE 还提供了一个图形界面的管理工具 Qmon，设置过程也可以通过 Qmon 更简便地完成。

6.2 GT4 的安装和配置

6.2.1 支持软件的安装

正确安装 Globus Toolkit 4 首先需要完成相关支撑软件的安装和配置，需要单独安装的支撑软件有 j2sdk，ant，MPICH 和一种关系型数据库等。

1.Java 开发工具包 j2sdk

从 SUN 的官方站点 http://java.sun.com 下载 JDK1.4 for Linux 版，目前最新的是 1.4.2_06，然后运行 j2sdk-1_4_2_06-windows-i586-p.bin，将其安装到/usr/j2sdk 目录下，同时设置环境变量 JAVA_HOME=/usr/j2sdk。

2.ANT 编译工具 apache-ant

Apache Ant 是基于 Java 的编译工具,可以从官方网站 http://ant.apache.org 下载 Apache Ant 1.6.1，然后解压，编译安装到/usr/local/apache-ant 目录,并设置环境变量 ANT_HOME=/usr/apache-ant。

3.关系数据库 postgreSQL

PostgreSQL 是一种非常复杂的对象-关系型数据库管理系统（ORDBMS），也是目前功能最强大，特性最丰富和最复杂的自由软件数据库系统。有些特性甚至连商业数据库都不具备。这个起源于伯克利（BSD）的数据库研究计划目前已经衍生成一项国际开发项目，并且有非常广泛的用户。从官方站点 http://www.postgresql.org 下载程序 postgresql-7.4.11-1PGDG.i686.rpm 并将它安装到/usr/local/psql 目录下。

安装完毕后，需要在/etc/profile 文件中设置相应的环境变量，在 SGE 和 GT4 全部安装完毕后的/etc/profile 文件内容如表 6-1 所示。

6.2.2 安装 GT4

首先创建一个名为 globus 的非特权用户，这个用户将用来安装 GT4，并且可以执行管理任务，比如启动和停止容器，部署服务等等，建立 globus 用户可以读写的安装目录/home/globus/gt4。下载 GT4 二进制安装包，然后解压 GT4 安装包到安装目录，进入解压包目录，通过以下命令完成安装。

$ ./configure --prefix=/home/globus/gt4

$ make

$ make install

为了让系统知道刚才安装的 GT 命令的位置，必须在/etc/profile 文件中设置一个环境变量 GLOBUS_HOME 并且对脚本 globus-user-env.sh 执行 source 操作，如表 6-1 所示。

6.2.3 GT4 的安全配置

1．安装 SimpleCA，首先在 master.dncpc 计算机上安装 CA 中心，然后，将 CA 中心的 simpleCA 包分发给其它计算机，接着安装 GSI，在主控机上为主机和用户申请和签署证书。

2．为使容器可以访问主机证书，需要将主机证书复制为容器的证书，即将 hostkey.pem 复制为 containerkey.pem，将 hostcert.pem 复制为 containercert.pem 并修改文件 containerkey.pem 和 hostcert.pem 的所有者为 globus 用户。

4．为 test 用户申请并签发证书

5．为用户增加授权，以 root 用户登录系统并在/etc/gird-security 目录下创建文件 grid-mapfile，然后将 test 用户的证书主题和对应的帐号名写入其中，也可以通过命令# /home/globus/gt4/sbin/grid-mapfile-add-entry –dn \
“/O=Grid/OU=GlobusTest/OU=simpleCA-master.dncpc/OU=dncpc/CN=test” –ln test 写入。

6．为可靠文件传输（RFT）建立数据库系统 rftDataBase。

6.3 SGE 和 Gt4 的接口安装和配置

在安装了 SGE 和 GT4 之后，还不能通过 WS-GRAM 向 SGE 提交作业，还需要安装与 GT4 和 SGE 整合相关的软件包。在执行了 SGE 的完全安装以后，管理员应该能够激活报告日志文件，创建 MPICH 配置和用 SSH 整合 SGE 的

qrsh。所有的这些操作在默认的情况下是没有完成的，在进行 SGE 和 GT4 整合的时候必须完成。步骤如下：

（1）作业的日志文件需要通过在 SGE 全局配置中将 reporting 和 joblog 两个参数设置为 true 来打开，可以通过使用 Qmon 来设置，通过 qconf 查看参数设置的操作为表 6-2 所示。

表 6-2 reporting 和 joblog 参数设置表

```
$ qconf -sconf
.....
reporting_params   accounting=true   reporting=true
\flush_time=00:00:15   joblog=true   sharelog=00:00:00
.....
```

（2）设置并行环境需要完成下面的一些操作：①在系统中安装 MPICH 软件，本文现在使用的是 MPICH2。②使用 SSH 无密码访问各个主机节点。③进行 SGE 中 MPICH 队列的配置，同样使用图形工具 Qmon 来进行这个队列的设置，设置的结果可以通过 qconf 查看，结果如表 6-3。④支持并行作业集群队列的设置，在配置了 MPICH 并行环境以后，需要将这个并行配置加入到集群队列配置中，作为变量 pe_list 的一部分，增加并行队列的操作也可以在 Qmon 中完成，可以通过 qconf 来查看配置情况，如表 6-4 所示。

表 6-3 MPICH 环境的设置表

```
$ qconf   -sp   mpich
pe_name           mpich
slots             4
user_lists        mcnc_list gridwisetech_list
xuser_lists       NONE
start_proc_args   /opt/sge/apps/mpi/startmpi.sh -catch_rsh $pe_hostfile
stop_proc_args    /opt/sge/apps/mpi/stopmpi.sh
allocation_rule   $round_robin
control_slaves    TRUE
job_is_first_task FALSE
urgency_slots     min
```

(3)安装 GT4-SGE6 的整合文件，这一步操作很容易，因为所有与整合相关的文件都是 GPT 的安装包，要顺利完成这一步操作，需要下载相关的 GPT 安

装包，设置 GLOBUS_LOCATION 环境变量，确认 MPICH 已经安装到这台计算机并且设置了 MPIR_HOME 环境变量来指示 MPICH 的安装目录，设置 SGE_ROOT 环境变量来指示 SGE6 的安装目录，这些工作都完成之后就可以编译和安装 GPT 包了。

表 6-4 集群队列配置表

```
$ qconf -sq parallel
qname          parallel
hostlist       @allhosts
.....
pe_list        make mpich
.....
```

经过这三步的操作，就完成了 SGE-GT4 适配器的安装，可以通过下面的一个命令来测试安装和配置过程是否成功。下面是将前面表 5-2 所示的 david_cpi .xml 作为测试文件，通过命令 globusrun-ws 提交给 GT4 之后的执行输出结果如表 6-5 所示。如果能看到类似于表 6-5 的输出，表明安装配置过程已经顺利完成。可以到这个 xml 文件所指示的 stdout 和 stderr 的目录位置去查看作业执行完毕后的结果。

表 6-5 作业 david_cpi.xml 通过命令的屏幕输出

```
[test@master test]$ globusrun-ws -S -submit -Ft SGE -f ./david_cpi.xml
Delegating user credentials...Done.
Submitting job...Done.
Job ID: uuid:8e7ad490-c149-11da-9cdb-00e04c3d276e
Termination time: 04/02/2006 06:34 GMT
Current job state: StageIn
Current job state: Pending
Current job state: Active
Current job state: StageOut
Current job state: CleanUp
Current job state: Done
Destroying job...Done.
Cleaning up any delegated credentials...Done.
```

第 7 章 网格服务层的实现

网格服务层是校园计算网格的中间层，起着连接用户层和资源层的重要重用。本章主要讲述网格服务层中作业调度中心、MDS 资源监听器以及作业监听器的实现。用户作业提交后，系统通过作业调度中心统一调度作业、分配工厂服务并及时通过作业监听器反馈作业状态。

7.1 网格服务层软件环境

整个系统是基于 Globus 中间件并使用 Java/Java servlet/Java server page 来进行开发。因为 Globus 为网格开发提供了丰富的 Java API，所以为使用 Java 的网格开发人员提供了便利的接口，在校园计算网格的开发中主要使用到了以下一些重要的包。

（1）org.globus.gsi 包

该包是为了获得网格安全，提供了诸如识别资源的 X.509 证书，创建临时证书以及执行单点登录和权限委托的接口等。

（2）org.globus.myproxy 包

org.globus.myproxy 中包含了 MyProxy 和 MyProxyTest 两个重要的类以及一个 MyProxyException 异常类。其中 MyProxy 类提供了与 MyProxy 服务器通信的 API，提供了从 MyProxy 服务器上获取、删除和存储证书的主要功能。

（3）org.globus.wsrf 包

org.globus.wsrf 中定义了 NotifyCallback，ResourceProperties，ResourceLife time，Subscription，Topic，TopicListener 等重要的接口。主要用于客户端应用订阅主题和接受通知。

（4）org.w3c.dom 包

org.w3c.dom 提供访问文档对象模型（DOM）的接口，提供了处理 XML 文档的一组 Java API，包括了 Document，Element，Node 等重要接口。

（5）org.oasis.wsn 包

包含了 NotificationConsumer，NotificationProducer，SubscriptionManager 等重要接口，以及 Notify，Subscribie，SubscribeResponse 等方法。

（6）org.globus.exec.generated 和 org.globus.exec.utils 包

在这两个相关包中包含了作业处理的 Java API。

7.2 作业调度中心的实现

本文在实现过程中定义了作业调度中心 Scheduler 类，用来处理调度事务。

该类是 Thread 的一个子类，通过 schedulerState 变量来控制作业调度中心线程的启停。根据资源和提交的作业类型的不同，在作业调度中心类中设置了 simple 和 MPI 两种不同类型的等待队列，当有作业被提交到系统，就根据其类型将作业插入到相应的等待队列中进行排队，作业调度中心的实现流程如图 7-1 所示。

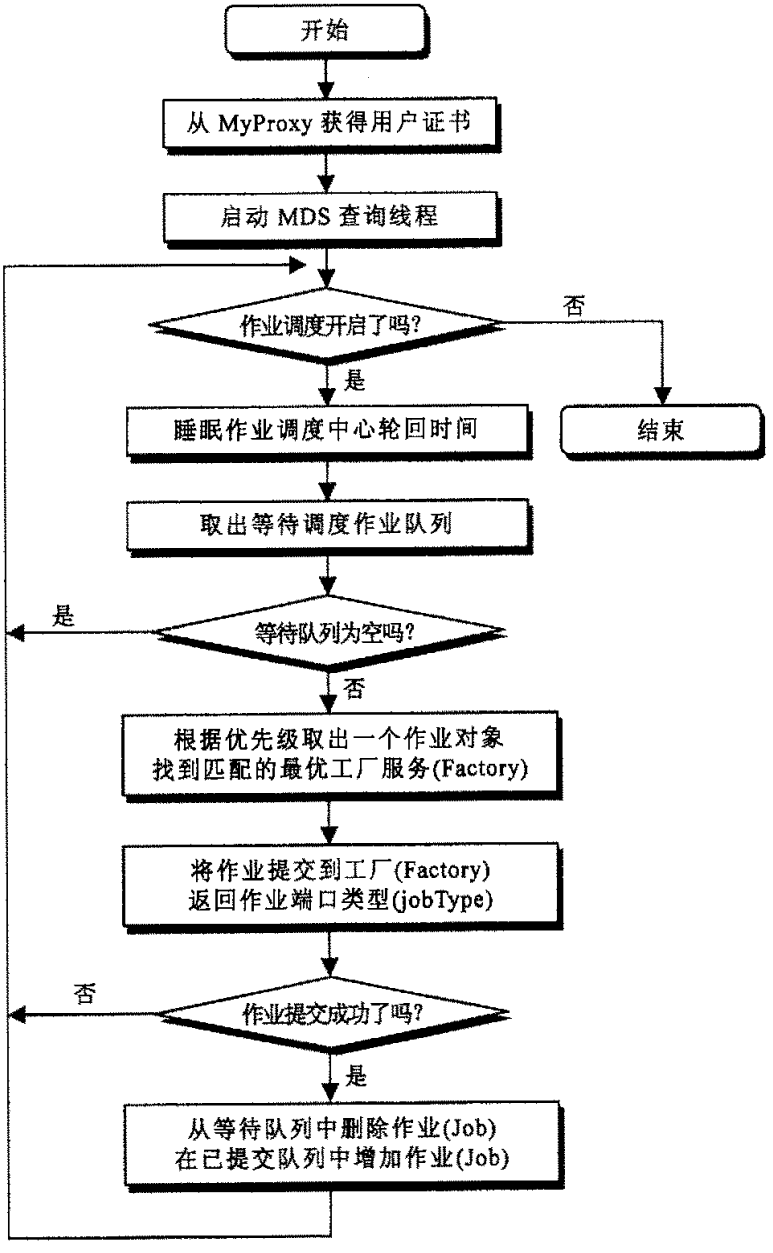

图 7-1 网格作业调度中心线程执行流程

当作业调度中心线程被运行起来后，首先要从 MyProxy 代理服务器中获得用户证书，使得用户可以合法地使用系统资源，然后开启 MDS 线程查询更新当前可用工厂资源，接着轮询等待队列，查看是否有等待处理的作业，如果有

就分配到由 MDS 资源监听器找到的最优工厂服务中，返回作业控制句柄，并为该作业创建作业监听器，订阅作业状态主题，接收通知并获得当前作业状态。如果作业控制句柄不为空，表明作业已经成功提交到工厂服务，需要更新队列信息，包括从等待队列中删除该作业记录，在已提交队列中增加该作业记录等，关键代码见表 7-1。

表 7-1 作业调度中心线程关键代码段

```
public void run() {
  //创建用户代理证书
  proxyCreate("",3600);
  //启动 MDS 监听线程
  mdsThread.start();
  //作业调度中心是否启动
  while (this.schedulerStates.equals("start")) {
    //作业调度线程睡眠 5 秒
    Thread.sleep(5000);
    //根据作业类型从后备队列中选择一组作业
    ArrayList it = getQueuebyJobType (unsubmitList);
    //从等待作业列表中挑选优先级最高的作业，获得作业的 id 号
    String jobid = getJobidbyPrioriry(it);
    //根据 jobid 获得作业对象 job
    Job job = unsubmitJobTable(jobid);
    //从可用资源中挑选最优工厂资源
    GramFactory gramFactory = getOptimizationFactory(validFactories);
    //向工厂递交作业对象，返回管理作业端口
    jobPort = submitJob(gramFactory, job);
    //作业监听器
    jobListener.start();
    //判断端口号，确定作业是否提交成功
    if(jobPort!=null){
      //从未提交作业队列中删除作业对象
      this.unsubmitQueue.deleteJob(currentJob);
      //向已提交队列中添加该作业对象
      this.submitedQueue.addJob(currentJob);
    }
  }
}
```

作业通过方法 submitJob 提交到指定的工厂后，会建立作业服务器桩，并返回作业控制句柄，这时用户便失去了对作业的控制，用户要了解自己提交作

业当前的执行情况只有依靠作业监听器的返回信息，递交作业到指定工厂的流程如图 7-2 所示。

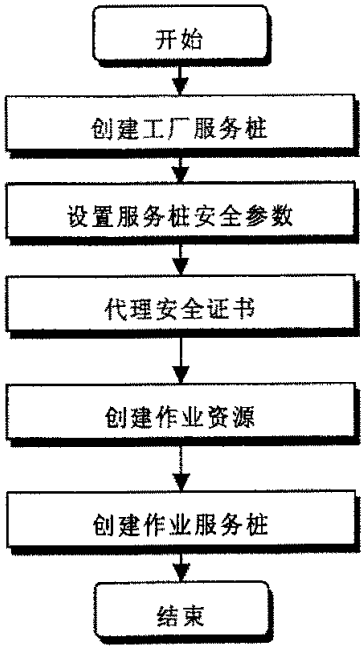

图 7-2 作业提交到工厂的流程

7.3 MDS 资源监听器的实现

网格资源存在着动态性，利用 MDS 对资源进行监控，可以保证将作业正确递交到可用工厂资源。

7.3.1 MDS 的发展

到目前为止，MDS 随着 GT 的发展大致经历了三个阶段，分别是 MDS2、MDS3 和 MDS4。MDS2 是元计算目录服务(metacomputing directory service)，是基于轻量级目录访问协议(light directory access process, LDAP)信息服务的实施，主要由网格资源信息服务(GRIS)和网格目录信息服务(GIIS)来实现。MDS3 是网格信息服务在 OGSI 上的实现，主要包括了索引服务及服务数据提供者以及三种接口，网格服务的接口包括 Factory、Grid Service Handle(GSH)、Grid Service Reference(GSR)、Query、Registry 以及 Notification 。而 MDS4 是基于 WSRF 的网格信息服务的实施。

7.3.2 MDS 的结构及作用

GT4 中的 MDS 包括了 MDS4 和 MDS2，MDS4 关注的是服务与资源配置

及状态信息的获取、分布、索引、归档以及处理。有时是为了发现服务及资源，有时是为了进行系统的监控。可以利用 Java、C 以及 Python 开发基于 WSRF 和 WSN 的 Web 服务核心组件。

MDS 可以被理解为一个协议沙漏，为信息访问和传输定义标准协议，为信息表示定义标准的文档计划，在沙漏的颈部以下，对不同的本地信息源的 MDS4 接口，转化他们的不同的计划格式到统一的 XML 格式。在颈部以上，利用统一的 Web Service 查询，订阅和通知接口构件不同的工具和应用。也就是 MDS 中的角色可以分成三个层次，中间层是 MDS，上层是各种可以访问 MDS 的高层应用，下层是为 MDS 提供各种信息的信息提供者。应用与 MDS 之间、信息提供者和 MDS 之间采用特定的协议通信。信息提供者用软状态注册协议向 MDS 注册自己感知的信息，应用通过查询协议从 MDS 中请求自己需要的信息。

MDS4 是在查询、订阅和通知协议和接口上构建的，而这些服务和接口是在 WSRF 和 WS-Notification 中定义的，在 GT4 中被实现。基于这个基础，已经实现了一组信息提供者，用于从指定的信息源收集信息。这些组件经常与其他的工具和系统连接，比如 ganglia 集群监控器、PBS 和 Condor 调度器。

MDS4 的主要组件包括 Index service、Trigger service 和 Aggregator source，其中 Index service、Trigger service 有时也称为 Aggregator Services，这些组件都是构建在 Aggregator Framework 上的服务。

- WS MDS Index Service — 该服务搜集网格资源的状态信息，并将其存入一个存储单元。通常来说，虚拟组织会部署一个或多个索引服务，这些服务随后搜集该 VO 中所有可用的网格资源的数据。
- WS MDS Trigger Service — 该服务按照管理员的指示，从网格资源中搜集数据，并将这些数据传递给适当的程序，从而根据事件执行各种操作。例如，当一个计算资源的队列长度达到一个特定的阈值时，系统就可能给管理员发送一封电子邮件。
- Aggregator Source — 这是一个 JAVA 类，用于搜集 XML 格式的数据，我们可以使用它来聚合 WSRF 服务。

基于以上的三个组件，MDS4 主要具有以下的作用：

（1）注册网格资源。Aggregator Service 能够通过 WS-ServiceGroup 的 Add 方法来注册网格资源。注册能通过 MDS 的 Index Service 或者网格资源的形式进行。注册拥有一个生命周期，若不时常更新，则会过期。目前有两种注册方式：①能够获取任意一种网格资源（不管它是否遵循 WSRF），这种模式提供了一种应用程序使用户更新数据。②遵循 WSRF，这样的服务把状态信息看成资源属性。

（2）收集信息。Aggregator Source 可以聚合我们所需的网格资源并且将这

些资源信息通过 XML 的形式发送到 Aggregator Service，也就是 MDS Index Service 和 Trigger service 中。

（3）发布信息。Aggregator Services（Index Service 和 Trigger Service）将收集到的信息以各种不同的格式发布给用户，比如用户可以通过 WebMDS 以及触发器来获取信息。它们的关系如图 7-3 所示。

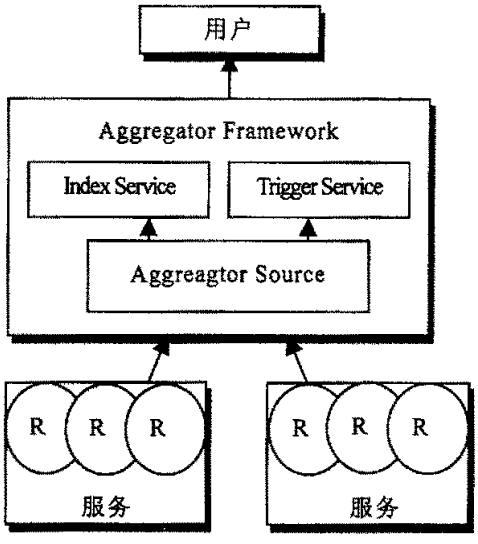

图 7-3 MDS4 服务框架

在安装 GT4 之后，我们可以通过启动容器并查看容器中可用的信息服务清单来验证默认的索引服务的确存在。这个过程如表 7-2 所示。

表 7-2 容器启动后的相关服务表

server at: https://192.168.0.169:8443/wsrf/services/ With the following services:
https://192.168.0.169:8443/wsrf/services/IndexFactoryService
https://192.168.0.169:8443/wsrf/services/NotificationConsumerService
https://192.168.0.169:8443/wsrf/services/ReliableFileTransferFactoryService
…… …… ……
https://192.168.0.169:8443/wsrf/services/DefaultTriggerService
https://192.168.0.169:8443/wsrf/services/TriggerService
https://192.168.0.169:8443/wsrf/services/ManagementService
https://192.168.0.169:8443/wsrf/services/ManagedExecutableJobService
https://192.168.0.169:8443/wsrf/services/DefaultIndexService
https://192.168.0.169:8443/wsrf/services/ManagedJobFactoryService
… … … … … …

通常，在一个具有各种服务和相关资源的大型网格环境中，我们可以查询特定服务以及查询各类资源属性的变化。在校园计算网格环境中，作业被递交给 GRAM，也就是递交给能够提供 ManagedJobFactoryService 服务的主机，而在系统中 MDS4 能够帮助我们收集这些服务资源，这个过程只需要在 Index Service 中进行就可以了。

7.3.3 MDS 线程工厂服务查询流程

用户可以通过 GT4 提供的 ManagedJobFactoryService 服务来为用户执行作业，为了将作业提交到可用的作业工厂服务，需要使用 GT4 中的 MDS4 来查询作业管理工厂服务的存在。

由于在 GT4 中资源属性是用 XML 描述，所以使用 Xpath 查询和获得相应的属性值，Xpath 是一种搜索 XML 文档的非常方便的查询语言，为了在 Globus 服务中发现每一个包含 ManagedJobFactoryService 的 Addressing 资源，我们可以将下面的 Xpath 查询请求发送到索引服务（Index），//*/*/*/*[local-name()=' Address' and contains(.,"ManagedJobFactoryService")]。索引服务提供了一种方法来发现基于资源属性的服务，可以通过返回的响应得到查找到的服务，然后从中选择所有包含 ManagedJobFactoryService 的服务实体。用得到的返回结果来更新可用工厂列表，这个时候的工厂列表就是当前可用的资源了，以后用户提交的作业就可以递交给这些作业管理工厂服务了，再由他们来产生作业管理服务，并返回作业句柄，然后就可以使用这个句柄与作业进行交互，这个作业句柄实际上是一个作业 ERP。MDS4 资源监听器流程如图 7-4 所示。

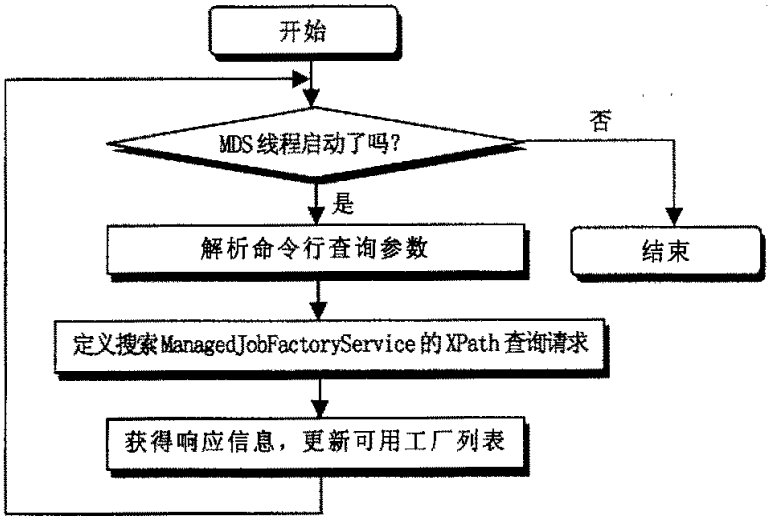

图 7-4 MDS 工厂查询线程图

对应的关键代码如表 7-3 所示。

表 7-3 MDS4 查询工厂服务关键代码表

```
String dialect=WSRFConstants.XPATH_1_DIALECT;
//设置 MDS 查询服务信息
String searchString= "//*/*/*/*[local-name()='Address'
    and contains(., "ManagedJobFactoryService" )]" ;
WSResourcePropertiesServiceAddressingLocator locator=
    new WSResourcePropertiesServiceAddressingLocator();
...
//设置 XPATH 查询语言
query.setDialect(dialect);
//设置查询请求
query.setValue(searchString);
...
//获得查询资源属性端口
port=locator.getQueryResourcePropertiesPort(this.getERP());
...
//设置 MDS 查询请求信息
request=setQueryExpression(query);
//获得 MDS 查询的响应信息
response=port.queryResourceProperties(request);
...
```

7.4 作业监听器的实现

作业一旦被作业调度中心调度，分配到工厂资源后，用户就不能再和它的作业进行交互，如何了解到作业当前的执行情况呢？可以为每个被调度的作业创建一个作业监听器。让作业监听器开启相应线程，实现作业状态的订阅和通知的接受，作业监听器流程如图 7-5 所示。

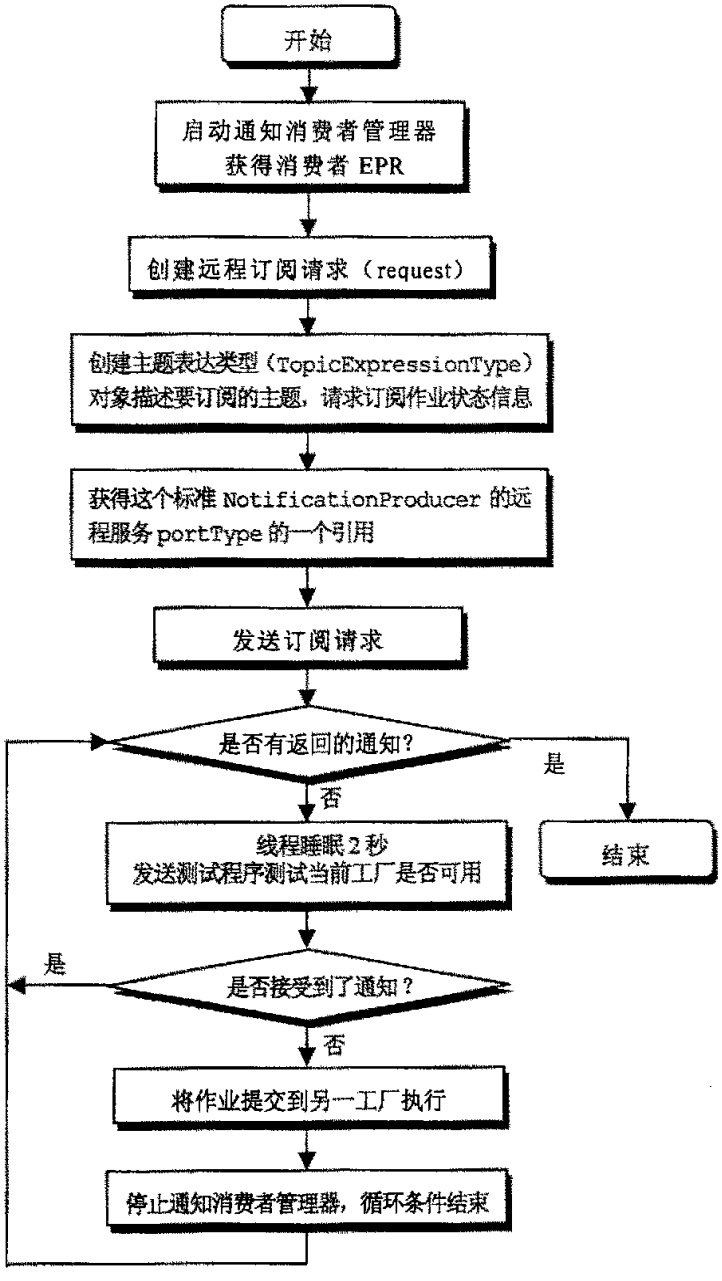

图 7-5 作业监听线程流程图

在客户程序中除了订阅主题以外，还要实现通知回调 NotifyCallback 接口的 deliver 方法，如果没有实现这个方法，客户端应用程序就没有办法接受到来自服务端的通知，通过实现 delver 方法，我们可以添加通知处理程序，流程如图 7-6 所示。

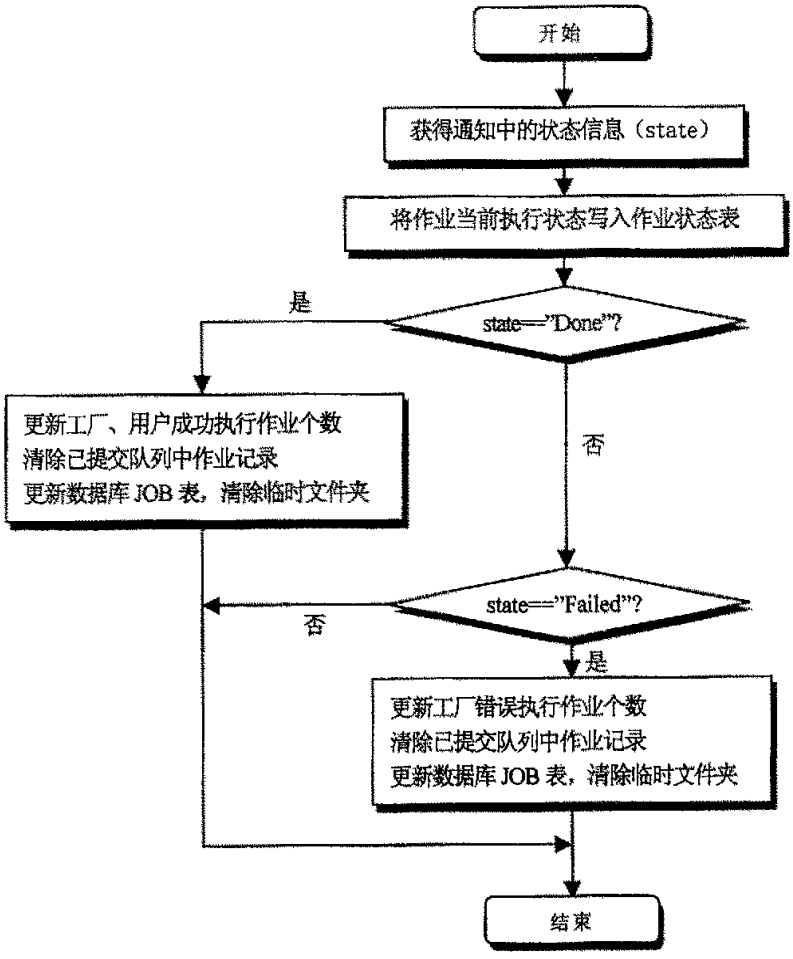

图 7-6 客户通知处理程序流程

第 8 章 用户层的实现

使用校园计算网格服务的用户群可分为三类（新用户、管理员和普通用户），用户层直接面向这三类用户，为他们提供了一个安全、高效、方便稳定使用网格服务的接口。为了达到这个目标，该层向新用户提供了注册功能，向管理员提供了用户管理、作业管理、在线作业监控、作业调度中心管理、配置信息管理、工厂资源管理以及系统服务管理等功能，向普通用户提供了作业提交、作业管理、资源查询以及个人信息管理等功能。

8.1 校园计算网格的注册与登录

本文使用 Jsp/Servlet 实现了校园计算网格的用户层，在这一层中使用到的类和方法都在业务逻辑层通过 Java 实现，在脚本中直接使用这些 Java 类及其方法就可以了。实例中用户通过任意一台接入到网络中的 PC 机，在浏览器的地址栏输入 http://192.168.0.169:8080/ campusgrid/ jsps 就可以访问校园计算网格的首页，如图 8-1 所示。

图 8-1 校园计算网格首页

如果是已被批准的合法用户可以在首页输入帐号和密码进行登录；如果是网格新用户，可以通过首页的用户注册功能进行注册，注册时应给出用户的详细信息，以便管理员确认申请者的身份，为其授权提供依据，新用户注册页面如图 8-2 所示。

图 8-2 用户注册页面

8.2 管理员接口

管理员接口向管理员提供了用户管理、作业管理、在线作业监控、作业调度中心管理、配置信息管理、工厂资源管理以及系统服务管理等功能。其中对用户的管理是校园计算网格的一个重要模块。从注册到撤消，管理员都通过这个模块来对用户在整个生命周期中进行全程的管理，如图 8-3 所示。

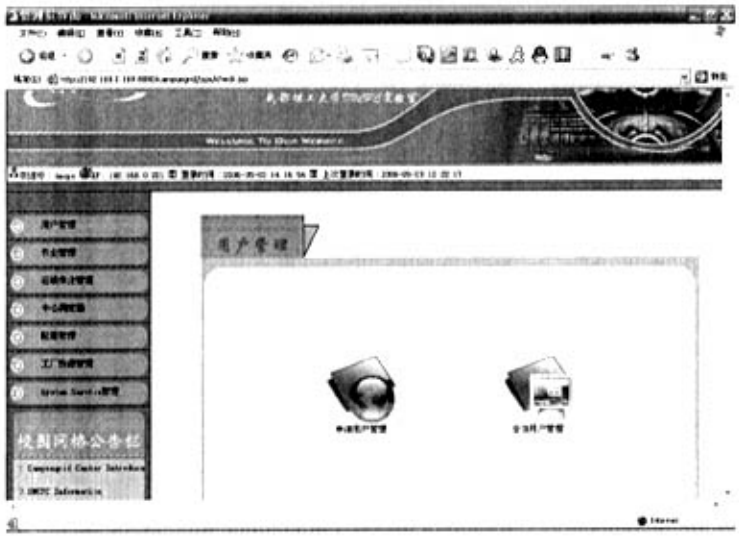

图 8-3 用户的管理页面

8.2.1 申请用户管理

当用户通过注册页面注册了用户信息后，就成为申请用户，为了提高系统的安全性，这类用户在未得到任何批准的情况下是不能进入网格环境的，必须等待管理员审查。申请用户管理功能为管理员查询、管理这类用户提供了方便。管理员首先可以通过注册用户列表查看当前已经注册但还未授权的用户信息，注册用户列表信息如图 8-4 所示。

图 8-4 注册用户列表

然后管理员可以选择一个申请用户记录查看用户注册信息，确定用户身份后，再根据用户身份通过申请用户管理页面对新用户授予一定的权限。经过授权的用户就可以通过主页登录到校园计算网格使用相关服务了。本文对用户的权限设置为三个等级，第一级是普通级，这类用户是可以通过用户应用平台提交作业，另外系统还预留了两级权限便于系统功能扩展，例如将来加入了更多的资源和服务，可以利用这两级权限实现资源和服务的分等级使用，使资源和服务得到更合理更有效的利用，同时本文考虑到不同用户具有不同的身份，系统应该对不同权限等级的用户提供不同级别的服务，于是将用户划分为四个等级（高级、副高、中级和初级），不同级别的用户提交的作业的优先级被静态地分配为一个整型数字（0，1，2，3），数字越高优先级越低，它将会作为作业优先级调度的依据。申请用户管理页面如图 8-5 所示。

图 8-5 申请用户管理页面

8.2.2 合法用户管理

经过管理员授权的申请用户为合法网格用户，这类用户就可以通过帐号和密码登录网格环境使用网格服务了。合法用户管理是为管理员提供对这类用户的管理功能，管理员可以通过该功能更改用户权限、修改用户身份级别以及查看、删除合法用户信息等。合法用户管理页面和申请用户页面很类似，如图 8-6 所示。

图 8-6 合法用户管理页面

8.2.3 在线用户作业管理

在线用户作业管理为管理员提供了在线监控作业执行状态的功能。通过这个功能管理员可以了解在线的所有用户提交的所有作业信息包括作业编号、作业类型、用户名、提交作业时间以及作业当前状态等，便于管理员对上线用户和作业进行及时监控，在线用户作业管理页面如图 8-7。

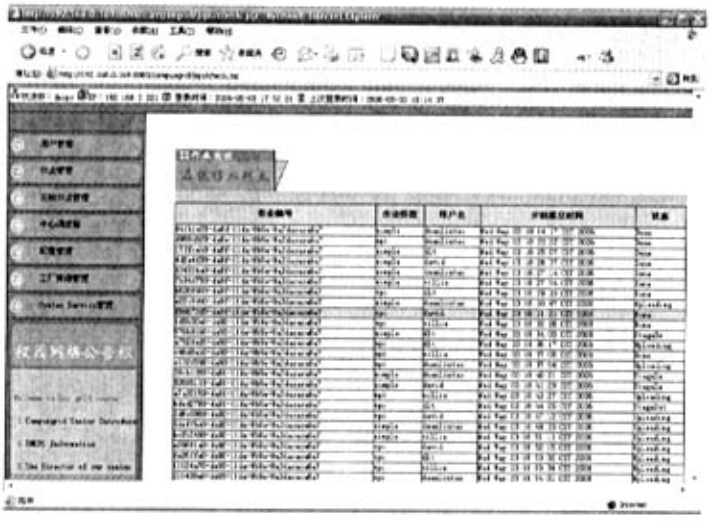

图 8-7 在线用户作业管理

8.2.4 资源信息管理

用户提交到网格环境的作业要被递交到工厂服务才能被执行，但具有工厂服务的 GRAM 主机在网格环境中是动态变化的，为了不将作业提交到一个失效的服务上去，系统使用 MDS4 来检测和更新工厂服务信息。系统引入了资源信息管理功能为管理员管理工厂服务信息提供了方便的操作界面。该功能可以让管理员能够查看、增加与删除工厂服务。这些工厂服务信息是存在与数据表中是被 MDS4 查询和更新的信息，为用户得到当前可用工厂服务提供原始的工厂服务的数据。资源信息管理如图 8-8 所示。

图 8-8 工厂信息管理

如果有新工厂服务加入到网格环境，可以通过增加工厂资源页面将其加入并记录在数据库的 factory 表中，作为 MDS 更新可用工厂资源的依据，增加工厂资源的页面如图 8-9 所示。

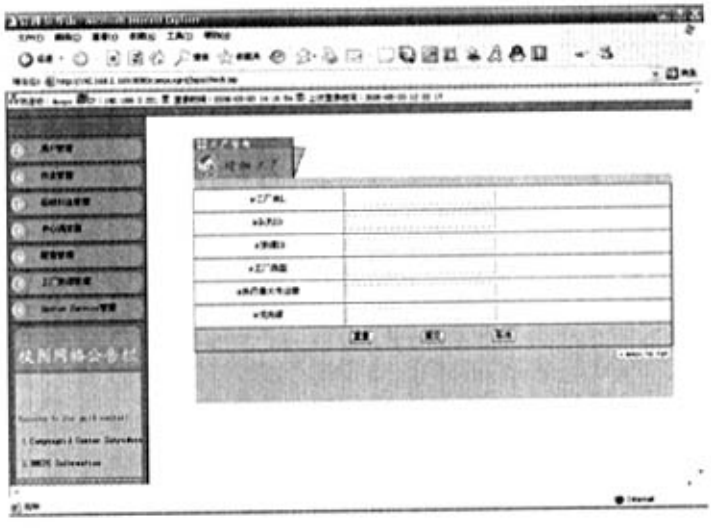

图 8-9 增加工厂资源页面

开启 MDS4 资源监测线程后，根据查询可用工厂资源的结果更新可用工厂信息，通过页面显示当前可用工厂资源，如图 8-10 所示。

图 8-10 当前可用工厂资源信息

8.2.5 作业管理

管理员通过这个页面来对网格所有用户提交的作业进行全局的管理，包括下载作业的执行结果，查看作业信息（包括作业名，作业所有者，作业提交时间，作业执行时间，作业的类型和状态等），删除作业等等，作业管理页面如图 8-11。

图 8-11 作业管理页面

8.2.6 服务管理

为了方便管理员对校园计算网格环境进行统一管理，系统设计了网格环境服务管理。本文提供了对 Globus 容器、Gridftp 服务器、Myproxy 服务器以及 PostgreSQL 数据库的管理，管理员通过该服务页面就能方便地完成对以上四种服务的启停任务，如图 8-12 所示。

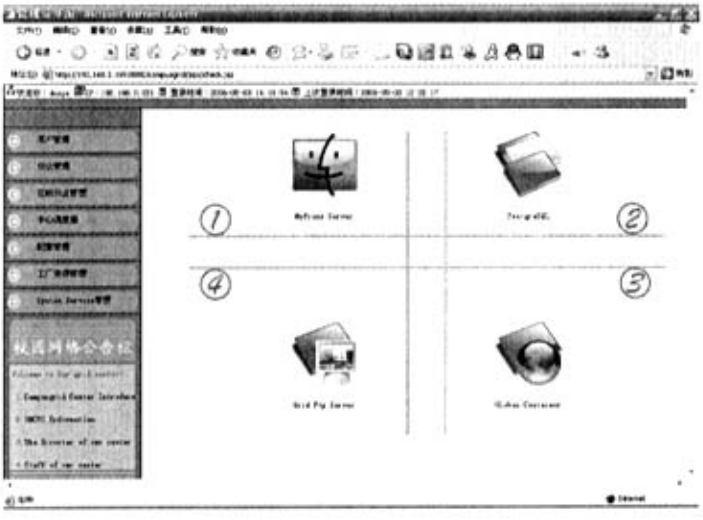

图 8-12 服务管理页面

网格服务管理功能不仅为管理员提供了一个安全、方便可靠的异地操作网格服务的 Web 接口，避免管理员直接登录服务器，使用 shell 命令控制和管理网格环境，而且还为系统的安全性和健壮性提供了保证。

为了实现对各种服务的控制，本文设计了 GlobusContainerManager、GridFtpManager、MyproxyManager 和 PostgresSQLManager4 个类，分别用于管理容器、gridftp、代理证书仓库以及数据库，这些类向外界主要提供判断服务状态、启动服务以及获得服务状态等接口。

当管理员通过网格服务管理页面去启动某项服务时，系统要完成以下操作：

（1）读取 SHELL 脚本文件，初始化环境变量 GLOBUS_LOCATION；

（2）通过 SSH 连接服务器，如果连接成功(Done)，进入第三步操作，否则显示连接服务器失败的信息；

（3）检测服务当前状态，如果状态为关闭(Off)，进入第四步，否则显示当前服务正在运行的信息；

（4）运行服务进程，开启服务，如果运行成功显示成功启动服务的信息(Done)，否则显示启动服务错误，返回第二步重新尝试。

为了说明服务控制功能的使用情况，首先将 Globus 容器所在的服务器

master.dncpc 上的 globus-gridftp-server 服务停掉，管理员在服务管理页面上点击 Grid FTP Server 图标便可以启动 gridftp 服务，服务启动后页面的输出结果如图 8-13 所示。

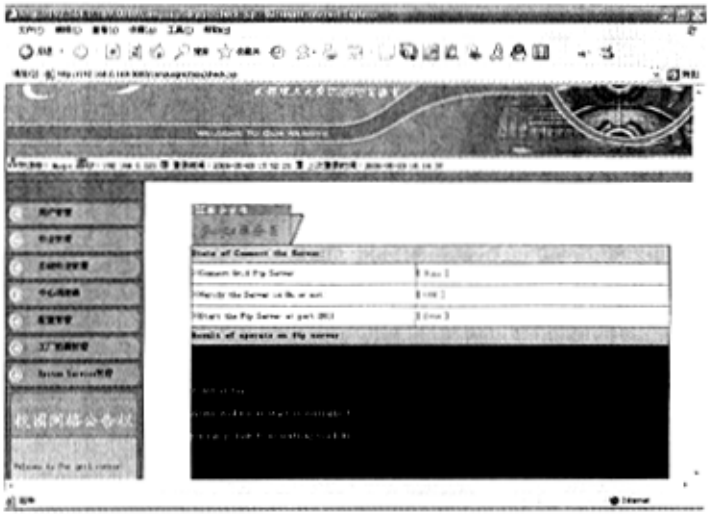

图 8-13 Gridftp 服务器启动页面

在 Gridftp 被成功开启后，如果管理员再一次启动服务，系统显示服务已经开启的通知如图 8-14 所示。

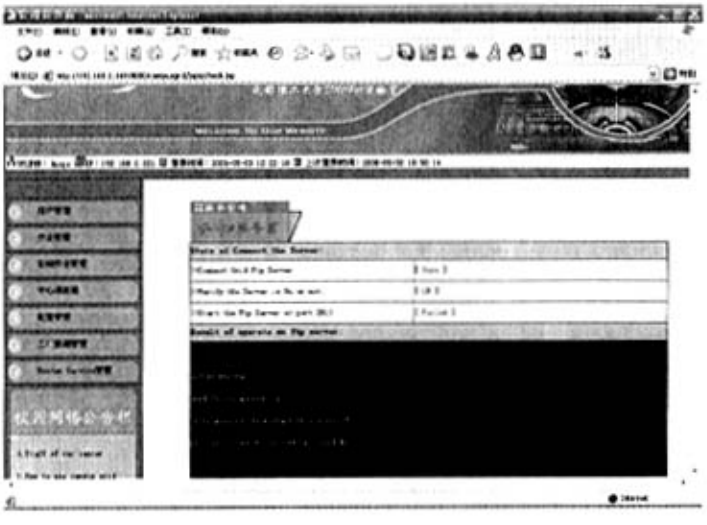

图 8-14 Gridftp 开启后的启动页面

GridFtpManager 类的关键代码如表 8-1 所示。

表 8-1 GridFtpManager 类的关键代码

```
public class GridFtpManager {
  ……
  //检测当前服务的状态
  public int isEnableFtpServer(){
    String executString = "ssh test@"+ftpServer+" ps ax";
    … …
     Process proc = runtime.exec(executString);
     InputStream is = proc.getInputStream();
     BufferedReader br = new BufferedReader(new InputStreamReader(is));
     while((tmp=br.readLine())!=null){
       if(tmp.indexOf("gridftp")!=-1){
         isEnable = 1;
         … …
         break;
       }}
     … …
    return isEnable;
  }
  //启动gridftp服务
  public boolean start(){
       String executeString = "ssh root@"+ftpServer+" /home/test/gridftp.sh&";
       … …
       Process proc = runtime.exec(executeString);
       InputStream is = proc.getInputStream();
       BufferedReader br = new BufferedReader(new InputStreamReader(is));
       while((tmp = br.readLine())!=null){
         if((tmp.indexOf("[")!=-1)||(tmp.indexOf("David")!=-1)){
           state = true;
           … …
           break;
         }}
       … …
       return state;
    }
  }
    … …
  //获取服务状态
  public boolean getFtpState(){
     return this.ftpState;
  }
}
```

8.3 普通网格用户接口

校园计算网格通过网格层和资源层为用户封装了底层资源，向用户提供一个计算力更强，结构更复杂的计算资源池，而普通网格用户接口的设计和实现为用户提供了使用这些强大计算资源的一个方便、安全、健壮的统一接口，普通用户利用这个接口可以向计算网格提交任意复杂度的串行和并行作业，也可以对自己作业进行有效的管理等。

8.3.1 作业提交

用户通过作业提交页面填写作业信息，包括程序名，程序所用参数等，然后由 Java Bean 形成作业的 RSL 描述，产生作业对象，然后将其插入到作业调度中心的相应作业等待队列中进行排队等待，这是网格用户使用校园计算网格资源的主要接口。在系统中用户主要递交两种类型的作业，串行和并行作业。用户可以通过并行作业提交页面提交 MPI 并行程序，在提交页面上用户需要填写作业的相关信息，包括 MPI 程序名及其存放地址，程序执行需要开启的进程数量，作业详细信息填写完成后就可以通过提交按钮递交到作业描述（RSL）产生器，如图 8-15 所示。

图 8-15 并行作业提交页面

当用户作业提交到最优作业工厂服务后会返回该作业的控制句柄，用户可以通过等待页面查看该作业执行的状态和进度，这个页面是每 5 秒钟刷新一次，根据通知判断作业的不同状态，然后更新进度条，让用户在一种比较友好的界面下了解作业的执行情况，如图 8-16 所示。当用户看到这个页面表示作业已经成功提交到服务器，使用者不需要一直等待该作业执行完毕就可以继续提交他

的下一个作业或者查看系统其它信息。

图 8-16 作业提交成功用户等待页面

为了动态的显示用户提交作业当前的执行状态，设计了 Progress 类，用于表示进度条对象，向外提供根据作业状态设定进度条位置的 setPositionbyState (String state)方法、进度条增加量 increment 的 increasePosition(int increment)以及获得当前进度的 getProgressPosition()等方法，同时定时刷新页面，表现作业当前执行状态，部分刷新页面代码如表 8-2 所示。

表 8-2 作业状态刷新页面代码

```
<script type="" language="javascript">
  function detect(){
    ... ...
     xml.open("POST","jobstatedetail.jsp",false);
     xml.setrequestheader("content-type", "application/x-www-form-urlencoded");
     xml.send(post);
     //获得jobstatedetail.jsp页面的返回信息
     var res = xml.responseText;
     content.innerHTML = res;
     //设置页面刷新时间间隔
     setTimeout("detect()",5000);
  }
</script>
</head>
<body bgcolor="#ffffff" onload="detect()">
<a id="content"></a>
……
```

8.3.2 作业管理

作业管理可以使得用户能够查看自己的作业信息，下载作业执行结果和删除作业等等，它的页面效果和管理员对作业管理的效果一样，如图 8-17 所示。

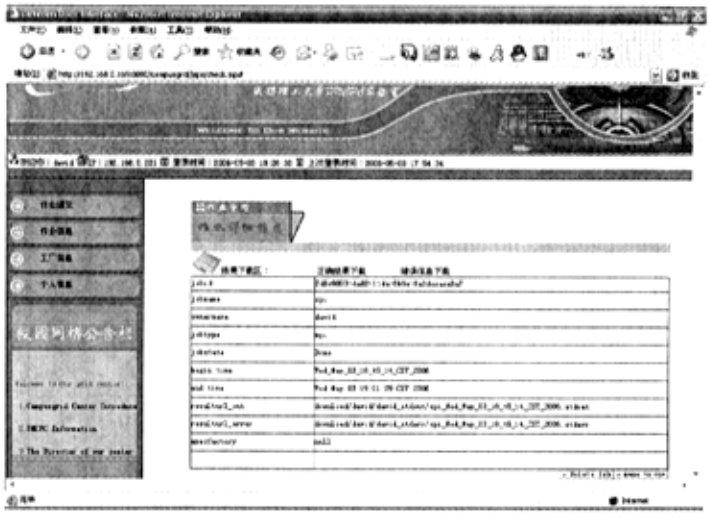

图 8-17 普通用户作业管理

可以从 GridFTP 服务器上下载作业执行的结果到本地计算机文件系统，用户可以通过页面上的“结果下载”获得作业执行结果，图 8-18 给出了 MPI 程序 cpi 由 10 个进程并行计算的执行结果的下载文件。

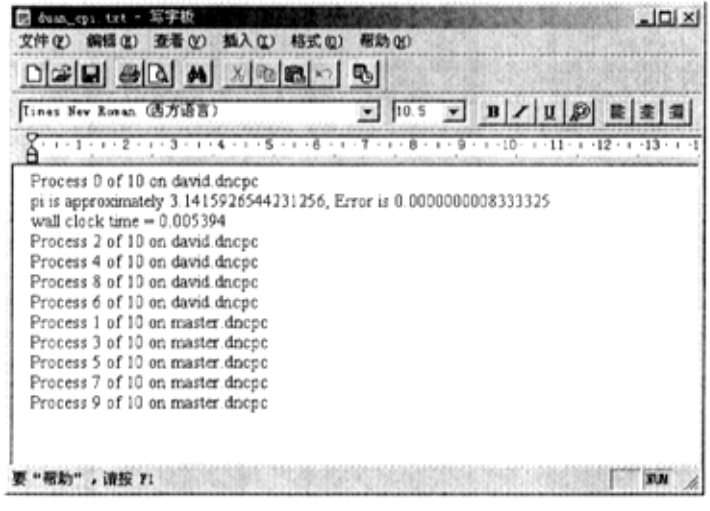

图 8-18 cpi 由 10 个进程并行计算的结果下载文件

8.3.3 工厂信息查询

通过工厂信息查询，用户可以了解当前主要有哪些网格节点在提供工厂服务。页面如图 8-19 所示。

图 8-19 工厂信息查询页面

8.3.4 用户信息修改

用户可以通过个人信息修改页面了解到自己的注册信息，同时可以修改自己的个人信息，如图 8-20 所示。

图 8-20 用户个人信息修改页面

结论与建议

网格计算技术要达到的目标是用户可以像使用电力一样方便地使用计算资源。目前学校中存在这样的问题，学校的高性能设备价格昂贵且不可复制，导致使用者少、利用率低，大量分散的计算资源闲置造成资源浪费，没有达到资源的共享和充分利用而形成高性能计算能力。网格计算技术在校园中应用研究使得我们能够在计算资源和校园用户之间搭建校园计算网格应用平台，从而能够充分利用资源，降低资源使用门槛，为全校师生的教学和科学研究提供全新的计算平台。

本文研究成果

本文较深入地研究了校园计算网格的建设和管理，介绍了计算网格以及 Web Services 的基本概念和体系结构，引入了将网格技术和 Web Services 结合的 Globus 项目 Globus Toolkit4，在此基础上利用 SGE 搭建了网格资源管理平台，研究了基于此平台的校园计算网格应用的架构，并实现了校园计算网格环境。本文的主要研究成果包括：

（1）在对开放网格服务体系结构 OGSA 的深入研究基础上，提出并设计了具有可扩展性的校园计算网格层次模型；

（2）通过对现有作业调度算法的分析和研究，设计了校园计算网格的作业调度模型并通过静态分配优先级的方法实现了基于优先级和先来先服务的作业调度算法；

（3）通过对校园计算网格中的作业两级调度的研究，设计和实现了作业调度中心，为完成作业的调度提供全局的服务；

（4）基于通知和服务定位机制，研究了校园计算网格中作业管理服务的定位、作业执行状态的收集以及 Globus 与 SGE 适配器的作业递交关系，设计与实现了用户作业在线监控；

（5）在以网格技术与 Web Service 相结合的 Web 服务资源框架（WSRF）为核心的 GT4 基础上，研究了校园计算网格的关键技术，设计并实现了校园计算网格的可用工厂服务监控功能，可以提供动态维护网格作业管理工厂服务的功能；

（6）采用最新的网格中间件技术和平台，设计和实现了校园计算网格管理平台和普通用户应用平台，为用户使用后台资源提供一个统一的访问接口，用户在异地异构机上可以通过浏览器控制、访问和使用网格服务，为以后进一步研究和发展校园计算网格服务奠定了良好的基础。

深入进行网格研究的建议

本文对校园计算网格模型进行了设计和实现，对将来的校园计算网格研究提供了较好的架构和平台，但是在校园计算网格上的研究还需要继续深入和完善，主要体现在以下方面：

（1）在网格作业的两级调度中，进一步优化和改进网格层作业调度算法。

（2）在现有工厂服务的基础上整合更多可用的计算资源；完善资源的发现功能，不但可以直接更新和发现工厂服务资源，还能检测各种加入到网格环境的硬件资源及其使用效率；将校园计算网格的计算资源扩大到整个学校。

（3）突破校园计算网格目前仅能够为用户提供计算节点资源的限制，通过进一步研究还应为用户提供使用其他设备资源的能力，扩大校园计算网格的应用范围；

（4）完善用户使用网格资源的统一接口，为用户提供更多的服务和功能。

随着高校对高性能计算资源需求的不断增加，校园网格计算技术的发展和应用将更加迫切，高校希望搭建属于自己的网格计算环境的需求也将变得更加普遍，校园计算网格的建设将会在更多的高等院校兴起，并具有长远的发展前景。

致谢

在三年的研究生生活和学习即将结束之际，首先衷心感谢我的导师罗省贤教授。三年间，罗老师在学习和生活上都给予了我极大的帮助，在她的谆谆的指导下，我对分布式计算以及网格技术有了深入的学习和研究，在她的帮助下，我开始接触新的技术和新的学术思想。本篇论文从选题，收集资料到研究和最后的撰写都是在罗老师的悉心指导下完成。罗老师学识渊博，治学态度严谨，为人言而有信，她为我们树立了学习的榜样。她在科研上严谨求实的精神、在教学上务实的工作态度都深深的影响和感染着我，对我以后的学习和工作都有重要的激励作用。

感谢南京军事网格中心主任刘鹏博士，感谢他在我参与南京河海校园网格项目工作及学习期间给予我的关心和帮助，他常常召集我们进行前沿的学术讨论，指导我们了解和学习国际上的先进技术和理念，他的师者风范给我们留下了深刻的印象。

感谢河海校园网格项目组组长王毅博士，在他主持的这个项目中，我学到了很多东西，让我对校园网格有了更深刻的认识和理解，感谢李建锋硕士，张燕鹤硕士以及其他所有河海校园网格项目组的成员，在他们的帮助下，我能够顺利完成自己的任务并学习到了书本中学不到的知识。感谢丁科教员，冯亚军硕士，杨涛硕士，宁雄雁硕士，李淑静硕士、陈东伟硕士以及南京军事网格中心的其他朋友，他们在我论文期间给了我极大的关心和鼓励。

感谢信息工程学院，外国语学院，研究生院所有帮助过我的老师和同学，感谢室友易涛、刘刚、李钢、张冠东，感谢李军博士、侯建花硕士、樊富有硕士、陈晓丹硕士、陈斌全硕士、王芳硕士、高建伟硕士、王金峰硕士、谌俐利硕士，以及 2004、2005 级的师弟师妹，感谢他们在学习、生活和工作中对我的支持，关心和帮助。

感谢我的父母、哥哥和嫂嫂，是他们给我建立的和睦家庭环境使我能够健康成长，是他们对我的爱和关心让我能够树立远大的理想和抱负。

最后要衷心的感谢所有出席论文答辩会的专家和老师，感谢你们在百忙中审阅我的论文。

参考文献

[1] 罗省贤，何大可著. 基于 MPI 的网络并行计算环境及应用[M]. 成都：西南交通大学出版社，2001.

[2] 都志辉，陈渝，刘鹏著. 网格计算[M] 北京：清华大学出版社，2002

[3] 徐志伟，冯百明，李伟著. 网格计算技术[M] 北京：电子工业出版社，2004

[4] 李安渝等著. Web Services 技术与实现[M] 北京：国防工业出版社，2003

[5] Bill Brogden 著，邱仲潘等译. Java 开发指南-Servlets 和 JSP 篇[M] 北京：电子工业出版社，2001

[6] 马斗等著. 专家门诊,JSP 开发答疑 200 问[M] 北京：人民邮电出版社,2005

[7] 杨学瑜著. JSP 入门与提高[M] 北京：清华大学出版社,2002

[8] Bruce Eckel 著 Java 编程思想[M] 北京：机械工业出版社,1999

[9] Bil Lewis, Daniel J.Berg 著 深入学习:Java 多线程编程[M] 北京：电子工业出版社,2000

[10] 张龙祥著. UML 与系统分析设计[M] 北京：人民邮电出版社 2001

[11] Bruce E.Wampler 著 Java 与 UML 面向对象程序设计[M] 北京：人民邮电出版社,2002

[12] 汤子瀛，哲凤屏，汤小丹著.计算机操作系统[M] 西安：西安电子科技大学出版社，2001

[13] 东方人华编. JBuilder 10 入门与提高[M]. 北京：清华大学出版社. 2005

[14] Richard Petersen 著. Linux 编程命令详解[M]. 北京：电子工业出版社. 2001

[15] 张琦 校园网格系统中的资源管理和任务调度 华中科技大学学位论文，2004

[16] 樊富有 校园计算网格模型研究及 UCGrid 管理系统的实现 成都理工大学学位论文，2005

[17] 侯正雄 校园计算网格环境下作业管理的研究与实现 西北工业大学学位论文，2005

[18] 任开军，张卫民，胡庆丰等. 网格门户下基于 GSI 的 MyProxy Login 的实现[J] 计算机工程与应用 2004,13-0036-04

[19] 陈娟，王汝传 开放网格服务结构中 WSRF 与 OGSI 的比较分析[J] 江苏技术师范学院学报 2005.8

[20] Joshy Joseph, Craig Fellenstein 著，战晓苏，张少华译.网格计算[M] 北京：清华大学出版社 2005

[21] 孙卫琴 李洪成著，TOMCAT 与 JAVA WEB 开发技术详解[M] 北京：电子工业出版社 2004

[22] I. Foster, C. Kesselman, etc. The Grid: Blueprint for a New Computing Infrastructure USA: Morgan Kaufmann Publishers, 1999

[23] http://www.casa-sotomayor.net/gt4-tutorial/

[24] http://www.chinagrid.net

[25] http://www.globus.org/toolkit/docs/4.0

[26] http://www-unix.mcs.anl.gov

[27] http://www-unix.globus.org

[28] http://www.postgresql.org/

[29] Borja Sotomayor . The Globus Toolkit 4 Programmer's Tutorial Univeristy of Chicago Department of Computer Science,

[30] IBM red Books: Enabling Applications for Grid Computing with Globus

[31] Steve Graham, Karl Czajkowski, Donald F Ferguson, Ian Foster etc. Web Services Resource Properties 2003

[32] Jason Novotny, Steven Tuecke, Von Welch. An Online Credential Repository for the Grid: MyProxy

[33] mcs.anl.gov. MDS4 and Project Deplyments. July 28, 2005

[34] Ben Clifford. Globus Monitoring and Discovery. The Globus Alliance

[35] Jarek Gawor, Sam Meder. GT4 WS Java Core Design. May 24, 2004

[36] Jennifer M. Schopf. Distributed Monitoring and Information Services for the Grid. Argonne National Laboratory. Sept 30, 2005

[37] Ian Foster, Carl Kesselman, Jeffrey M. Nick, Steven Tuecke. Grid Service for Distributed System Integreation, IEEE, 2002

[38] Ian Foster, Carl Kesselman, Jeffery M. Nick, Steven Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration

[39] Bogdan Lobodzinski, Krzysztof Wilk, Pawel Plaszczak. Globus Toolkit 3.9.5-Sun Grid Engine 6.0(update 3) Interface: description. Gridwise Technologies, March/April 2005

[40] http://www.gridwisetech.com

[41] http://www.sun.com/software/gridware/

[42] http://www-128.ibm.com

[43] Asia Pacific Science and Technology Center, Sun Microsystems, Nanyang Center for Supercomputing and Visualization, Nanyang Technology University, Nanyang Avenue, Singapore. Sun Grid Engine Installation Guide.