**研究报告**

**基于网格的多Agent系统架构研究**

**王璐**

**2015年4月**

**目录**

[1 网格计算 3](#_Toc430166647)

[1.1 网格研究的背景和意义 3](#_Toc430166648)

[1.2 网格的特点 4](#_Toc430166649)

[1.3 网格的发展历程及趋势 5](#_Toc430166650)

[1.4 网格研究的主要内容 7](#_Toc430166651)

[2 网格体系结构 9](#_Toc430166653)

[2.1 五层沙漏网格体系结构 9](#_Toc430166656)

[2.2 基于web服务的OGSA网格体系结构 10](#_Toc430166660)

[2.3 Web服务资源框架 11](#_Toc430166665)

[3 网格应用 12](#_Toc430166667)

[3.1 网格监控 12](#_Toc430166669)

[3.2 调度 14](#_Toc430166670)

[3.3 资源管理 18](#_Toc430166671)

[3.4 安全 21](#_Toc430166672)

[4 多Agent在网格中的应用 24](#_Toc430166673)

[4.1相关学者 28](#_Toc430166674)

[4.2 国际会议 30](#_Toc430166675)

[5 基于网格的多Agent系统架构研究 31](#_Toc430166676)

[5.1 基于中间件的体系结构 31](#_Toc430166677)

[5.2 基于网状结构的体系结构 35](#_Toc430166678)

[5.3基于层次的体系结构 36](#_Toc430166679)

[5.4 其他系统的体系结构 39](#_Toc430166680)

[6 总结与展望 46](#_Toc430166681)

[参考文献 48](#_Toc430166682)

# 网格计算

网格(Grid)概念起源于20世纪90年代中期。1998年，Ian Foster对于网格做出过这样的描述：“通过网格技术可以将在地域上广泛分布的诸如计算机、数据库之类的各种计算资源、存储资源以及其他各种网络设备组合成一个有机的整体，从而使这些原本孤立的各种计算机资源通过网络为我们的生产、生活提供更多、更好的服务。”

2001年，Ian Foster进一步将网格定义为：支持在动态变化的分布式虚拟组织间共享不同资源，协同解决问题的系统[1]。

2002年，Ian Foster强调一个网格系统必须满足三个条件[2]：

(1)在非集中控制的环境中协同使用资源；

(2)使用标准的、开放的、通用的协议和接口；

(3)能够提供非凡的服务质量。

## 网格研究的背景和意义

随着生产、科研活动的不断深入和拓展，人类迫切需要功能强大的计算机系统以解决一些超大规模的应用问题，如核爆炸模拟、高能物理实验分析，生物研究和航天飞行等。然而它们所需要的计算能力已不可能从单一的计算机或集群系统中获得。另一方面，随着计算机网络技术的飞速发展，实现全球范围内计算资源的高速互连成为可能。网格的提出从根本上改变了人们对“计算的看法”，突破了以往强加在计算资源之上的种种限制，它使人们以更加自由、方便的方式使用各种计算资源，并带来了前所未有的计算能力[3]。

1．突破了计算能力和存储能力的限制[4]

网格能够把地理位置上广泛分布的各种计算资源、存储资源、通信资源等通过高速网络大规模集成，提供单个或者一组计算机无法提供的计算和存储能力。利用可扩展的大规模的资源，网格可以为人类源源不断地提供几乎无限的计算能力和存储能力。在这种先进工具的支持下，人们可以求解极其复杂，甚至以前无法解决的问题，大大提高了人类研究自然和改造自然的能力。

2．突破了地理位置的限制

在传统的分布式计算系统、集群系统中，计算资源通常局限在一个有限的空间内，这不仅限制了系统的计算能力同时也给系统的使用带来了不便。尤其是一些稀有资源和特殊设备更是无法和特定的地理位置分开，要使用这些资源必须到相应的地方去，很大程度上限制了这些资源的使用。而网格系统则通过使用共享语言和标准协议使各种资源延伸到全球范围以满足本地用户的需求。在网格系统中，对用户来说不存在“资源位置”的概念，整个网格系统是透明的，异地用户可以远程地使用网格上的各种资源，就像使用本地计算机一样。

3．打破了传统资源共享的限制

以往的资源共享往往停留在计算机之间的文件交换和远程登录，而网格资源的共享则允许对其它资源的直接控制和资源之间的互操作。网格技术能够实现虚拟组织中各成员之间大规模的科学、工程和商业协作，可以将各个领域内的专家、数据、设备等充分的结合在一起，协同解决问题，提高工作效率。

4．降低资源使用代价

随着计算机技术的迅速发展，计算机更新换代的速度也在加快，每年大量尚有利用价值的计算机设备被废弃，这不仅是一种资源浪费，同时还会造成一定的环境污染。另一方面，目前一些计算机设备的利用率很低，甚至处于长时间闲置状态，造成计算资源的一种浪费。网格可以有效整合网络上闲置的计算资源，为人们提供便宜的、通用的计算能力，达到“费半功倍”的效果。用户不用花很多钱购买设备，只要能够接入网格就可以得到所需的计算能力，因此网格也被认为是一种廉价的高性能基础设施。

## 网格的特点

网格作为一种新型的、在地理位置上广泛分布的基础设施，其用户数量巨大、资源千差万别、需求种类繁多、应用功能各异。网格要给用户提供安全、可靠、随处可得、高质量的服务，必须具备以下特点[5]：

1．虚拟性和抽象性。网格中的资源和用户都要经过抽象，把实际的用户和物理资源虚拟为网格用户和逻辑网格资源。网格用户使用标准、开放、通用的协议和接口来访问网格中的各种资源，实际的用户和物理资源是互不可见的，资源对外提供的只是一个虚拟化的接口。网格向用户屏蔽了物理资源的细节，使得用户可以只关心“实现什么”，而不必在意“如何实现”。

2．共享性。网格的最终目标就是资源共享，任何网格用户都可以非常方便地共享整个网格资源。这种共享是深层次和多样化的，网格中的多个用户不仅能够共同使用网格中的一个资源，网格中的一个用户也可以同时使用多个网格资源。

3．集成性。网格把地理位置上广泛分布的各种资源无缝地集成在一起，成为一个有机整体。用户不仅可以使用单个资源提供的功能，还可以联合使用多个资源的合成功能。网格能够集成来自不同管理域、不同管理平台、具有不同能力的资源，从而向用户提供统一的超级计算环境。

4．自治性和动态性。网格上的资源首先属于某一本地的个人或组织，网格资源的所有者对自己的资源具有最高的管理权限，网格应该允许资源所有者对其资源的自主管理。网格资源所有者具有自治性，可以根据自身情况随时决定加入或退出网格系统，“何时”，“以何种方式"向用户共享自己的资源。这种动态变化的网格环境，增加了资源管理的难度，提高了任务调度的复杂性，使得服务质量难以保证。一方面资源具有自治性，另一方面又必须接受网格系统的统一管理，否则不同的资源就无法建立相互之间的联系，无法实现共享和互操作，这也是网格计算和分布式计算及并行计算最本质的区别之一。

5．协商性。网格支持资源的协商使用，资源请求者和资源提供者可以通过协商得到不同质量的服务，满足不同的实际需求。请求者可以指定系统响应时间、数据带宽、资源可用性、安全性等各种要求，得到非平凡的服务质量。

## 网格的发展历程及趋势

网格从诞生到现在，主要经历了三个发展阶段。第一阶段是网格的萌芽阶段，开始于上世纪90年代早期，研究内容是关于千兆网络的试验床以及一些元计算方面的工作。其典型代表是FAFNER(Factoring via Network-enabled Recursion)和IWAY两个项目。这个时期的网格都是把超级计算节点连接起来为高性能应用提供计算资源，研究工作主要解决通信、资源管理和远程数据管理的问题。FAFNER发展之后出现TSETI@home和Distributed．Net等项目，IWAY发展之后则出现YGlobus和Legion等项目[6]。

第二阶段的大致时间是从1995年至U2000年。该阶段网格研究的目标是基于大规模计算和数据共享的科学研究。其典型代表项目是Globus，Legion等，尤其是Globusl工具集的开发和应用，对网格的发展起了很大的推动作用。主要解决的问题涉及网格的异构性、可扩展性、适应性以及网格安全等。这个时期的主要工作是支持大规模数据和计算的网格中间件的开发，建立了面向协议和模块的网格体系结构[7]。

第三阶段从2000年至今，它是网格技术迅猛发展的阶段。该阶段网格研究的目标由科学研究延伸到商业和工业应用。其典型代表是万维网服务和开放网格服务结构(OGSA)，典型的基础设施为OGSI(Open Grid Services Infrastructure)和WSRF(web Services Resource Framework)。该阶段网格的主要特点是面向服务的网格结构，2002年2月发布的OGSA把万维网服务和计算网格结合起来，建立了以服务为核心的网格结构。

网格的发展遵循以下几个趋势[8]：

1．统一标准化

正如其它技术一样，随着技术发展的日臻成熟完善，必然要求业界制定统一的技术标准。就像Internet需要依赖TCP／IP协议一样，网格也需要依赖标准协议才能共享和互通。开放网格服务体系结构OGSA把Globus标准与以商用为主的Web Services的标准结合起来，成为事实上的网格标准。OGSA得到业界的广泛支持，并被认为是网格发展的未来。

2．技术融合化

网格由传统的分布式计算发展而来，构架在计算机网络基础之上，因而不能完全独立于已有的、成熟的传统技术。网格中的各种资源基于各种不同的技术实现，要想实现异构资源的无缝整合必然要求网格技术实现多技术集成化。而OGSA正是利用了Web Services的标准，并借助一些现成的、与平台无关的技术来实现网格服务的描述、查找、访问和信息传输等功能，从而屏蔽一切平台及所使用技术的异构性。

3．广泛大型化

目前的网格系统大部分还具有地域性和行业性，这与网格的精神是相悖的。网格的发展目标就是全球范围内的资源共享，实现跨组织的协同合作，这也正是它优于传统分布式计算的地方。只有构建大范围的网格系统，才能保证网格资源的多样化，丰富化，才能凸显网格计算的优势和价值。目前许多国家和政府在不断地投入大量的经费来兴建国家网格，可以想象在不久的将来必然会出现更加大型化的全球网格。

4．应用商业化

网格计算是一种基础设施，只有走商业化的道路才能得到广泛的应用和发展。目前的网格应用主要集中于科学研究领域，还没有比较成功的商业应用，对人们的日常生活并没有发挥它应起的作用。如何找到网格应用商业化的切入点是业界比较关心的话题。目前出现的云计算在一定程度上可以说是网格计算概念的商业实现，但网格技术商业化的道路尚不明确，需要进一步地探讨和实践。

## 网格研究的主要内容

目前网格领域的研究工作可以被划分为以下几个方向[9]：

1．网格体系结构

网格体系结构是关于如何建造网格的技术和规范的定义，包括划分和定义网格基本组成部分、定义各个部分的功能、描述各部分之间的关系以及把这些不同部分有效组织、集成在一起的方法。可以说网格体系结构是网格的骨架和灵魂，是建立网格基础设施的蓝图。Foster等在网格研究的早期提出了以协议为中心的“五层沙漏模型”的网格体系结构；之后在以IBM为代表的工业界的影响下，Foster等又提出了以服务为中心的开放网格体系机构(OGSA)。目前OGSA是最新的一种体系结构，并成为事实上的行业标准。

2．网格信息管理

网格信息管理模块在网格中的地位非常重要，它为整个系统提供行动的依据，是保证分布资源协调一致工作的基础，在整个网格系统中处于中心地位。网格信息是关于网格中主体、客体，以及网格系统的数据，包括资源信息、用户信息、记账信息、日志信息等。如果说网格体系结构是网格系统的骨骼，那么网格信息管理模块则是网格系统的神经中枢，它通过网格信息把网格的各部分有机地连为一个整体。网格信息管理机构为网格资源的透明使用提供支持，对网格信息的表示、存储、收集、查询、更新等进行管理。

3．网格资源管理

网格的重要特征之一就是实现深层次的资源共享，因此资源管理是网格的基本功能模块。由于网格资源是异构的，通常分散于不同的自治管理域中，因此很难对所有资源进行统一管理。尽管存在许多有关资源管理方面的研究，但是目前还没有一种通用于各种环境下的资源管理模式。资源管理涉及的内容包括资源描述、资源发现、资源分配等。科学有效地管理资源可以充分发挥资源能力、提高资源利用效率，从而改善网格性能，更好地支持网格应用。资源管理的重要性吸引了许多相关的研究。

4．网格任务管理

网格的最终目的是为用户的应用提供计算、存储等各种服务。用户的各种应用在网格中以网格任务的形式来体现，任务的完成情况直接影响到用户对网格系统的感受，也是最受用户关注，与用户利益直接相关的内容。目前的任务管理主要涉及任务描述、任务提交、任务调度、任务执行、任务迁移等内容。

5．网格数据管理

数据作为一种特殊的网格资源具有可复制、可移动、可压缩、可加密等特性，通用的资源管理机制不能很好地满足数据管理需要，因此网格要有专门的数据管理模块，甚至出现了专门对数据进行管理的数据网格。数据管理作为一个单独的模块，其主要功能是为用户提供透明地访问、存储、传输、管理数据资源的手段，使用户能够较容易地实现网格环境下的数据共享。数据管理的核心是数据传输、数据存储、数据访问、副本管理等[10-12]。

6．网格安全

由于网格的规模比较大、资源构成比较复杂，这使得网格任务面临各种安全威胁，如信息被篡改、身份被冒用等，因此建立一个安全的网格环境尤为重要。目前的网格系统大多是实验性质的平台。环境比较封闭，用户比较单一，采取的安全措施也比较有限。一旦网格全面走向商业应用，开放复杂的应用环境必然会带来更多的威胁。建立一套有效的网格安全机制是网格成功走向商业应用的前提。网格安全涉及的内容可分为通信安全和访问安全。网格资源一般都分布在不同的地理位置上，保证它们之间通信的机密性是建立安全网格的基础。另一方面，这些资源通常隶属于不同的管理域，如何实现互相之间的身份认证和安全访问也是一个必须解决的问题[13]。



# 网格体系结构

网格体系结构是探讨网格技术时首先要回答的问题。因为网格体系结构给出了网格基本组成部分与功能，描述网格各组成部分的关系以及各组成部分之间如何交互集成。网格体系结构从最初的五层沙漏结构，演进到现在的WSRF，代表了网格技术发展的最新成果。



## 五层沙漏网格体系结构

以实现互操作为目的，基于协议和服务的网格五层体系结构包括：构造层、连接层、资源层、汇聚层和应用层，结构如图1所示。其中资源层和连接层共同组成了瓶颈部分，使得该结构呈沙漏形状。其内在的含义就是各部分协议的数量是不等的，对于其最核心的部分，要能够实现上层各种协议向核心协议的映射，同时实现核心协议向下层各种协议的映射，核心协议在所有支持网格的地点都应该得到支持，因此核心协议的数量不应该太多，这样核心协议就形成了协议层次结构中的一个瓶颈[14-16]。

(1)构造层：控制局部资源，包括查询机制(发现资源的结构和状态等信息)、控制服务质量的资源管理能力等，并向上提供访问这些资源的接口。构造层资源是非常广泛的，可以是计算资源、存储系统、目录、网络资源以及传感器等等；

(2)连接层：基本功能是实现相互的通信。它定义了核心的通信和认证协议，用于网格的网络事务处理。连接层的安全认证具有单点登录、代理、与局部安全方法集成和基于用户的信任机制等特征；

(3)资源层：基本功能是实现对单个资源的共享。该层定义的协议包括安全初始化、监视、控制单个资源的共享操作、审计以及付费等；

(4)汇聚层：基本功能是协调多种资源的共享。该层协议与服务描述的是资源的共性，说明不同资源集合之间是如何相互作用的；

(5)应用层：这层是网格上用户的应用程序。应用程序通过各层的API调用相应的服务，再通过服务调用网格上的资源来完成任务。



图1 五层沙漏结构

基于五层沙漏结构原理，Globus项目开发了网格计算领域应用非常广泛的模型G1obuS Toolkits，定义了构建计算网格的一组基本服务和功能。五层沙漏体系结构影响十分广泛，它的特点就是简单，主要侧重于定性的描述而不是具体的协议定义，容易从整体上进行理解。



## 基于web服务的OGSA网格体系结构

开放网格服务结构OGSA(Open Grid Service Architecture)是由全球网格论坛(Global Grid Forum，OGF)工作组于2002年制定的。网格的五层沙漏结构是以协议为中心的“协议结构”，而OGSA是基于Web service的“服务结构”。OGSA中的服务包括各种计算资源、存储资源、网络、程序、数据库等，简而言之，一切都是服务[17-19]。

OGSA有四个主要层次：物理与逻辑资源、web服务与定义网格服务的OGSI扩展、基于OGSA架构的服务，以及网格应用程序，如图2。物理和逻辑资源层：资源的概念是OGSA以及通常意义上的网格计算的中心部分。它们通过虚拟化和聚合物理层的资源来提供额外的功能。通用的中间件，比如文件系统、数据库管理员、目录和工作流管理人员，在物理网格之上提供这些抽象服务。

Web服务层：OGSA架构中的第二层是web服务。这里有一条重要的OGSA原则：所有网格资源(逻辑的与物理的)都被建模为服务。OGSI规范定义了网格服务并建立在标准Web服务技术之上。而且OGSI进一步扩展了Web服务的定义，提供了动态的、有状态的和可管理的Web服务的能力。



图2 OGSA架构

基于OGSA架构的网格服务层：包括四种类型的服务，即网格核心服务、网格程序执行服务、网格数据服务、特定领域的服务。其中网格核心服务是基于OGSA架构的服务层的主要构成部分。

网格应用程序层：随着基于网格架构的服务不断被开发出来，使用一个或多个基于网格架构的服务的新网格应用程序亦将出现。这些应用程序构成了OGSA架构的第四个主要的层。

网格系统采用统一的Web服务框架，很自然就具备了原来Web服务的有利因素：可以从服务描述中自动产生客户端与服务端的代码；将服务描述和互操作的网络协议绑定在一起；与新出现的高级开放标准、服务和工具兼容；有广泛的工商业支持等。



## Web服务资源框架

由于OGSI规范过分强调网格服务和Web服务的差别，没有对资源和服务进行区分，主要表现为网格服务和Web服务之间不能很好地融合，缺乏通用性，移植性差。为了解决OGSI和web服务之间不能很好地融合的问题，2004年GlobuS联盟和BIM推出了web服务资源框架(Web Services Resource Framework，WSRF)。Web采用了与网格服务完全不同的定义：资源是有状态的，服务是无状态的。WSRF的规范是针对OGSI规范的主要接口和操作而定义的，它保留了OGSI中规定的所有基本功能，只是改变了某些语法，并且使用了不同的术语进行表达。作为OGSA最新核心规范的Web服务资源框架WSRF，它的提出加速了网格和Web服务的融合，以及科研界和工业界的接轨。OGSA和WSRF目前都处于不断的发展变化之中，对于WSRF本身而言，由于其提出不久，其规范还有待在实践中进一步检验，并逐步得到完善[20-22]。



# 网格应用



## 网格监控

网格系统具有可扩展性、异构性、多级管理域、动态性等特征。网格是由分布在不同地理位置的不同管理域的资源组成，资源可能随时加入，也可能随时因失效或自愿等原因退出。网格的使用者和开发维护人员如何处理数量庞大、种类繁多的网格资源信息，如何监控网格环境的运行状态以及时发现故障的根源，如何为任务指派资源，如何监控网格资源状况和性能瓶颈并对其进行调整和控制，这些都成为网格监控亟待解决的问题。

网格监控系统本质上就是一种能有效地对网格资源进行管理，实现网格资源的统一“映像”，保证用户的需求被正确地反映到网格资源上的集成系统。当网络发生故障时，网格管理人员可以通过系统迅速定位，排除故障，同时对被影响的进程与用户状况了如指掌，提高网格的服务质量。一般地说，网格监控系统可以为网格提供以下支持：

(1)网格资源信息

网格监控系统可以探测并报告网格中的可用资源，由于网格的动态性，这些可用资源的信息不可能全部预先知道，需要检测系统实时发现。

(2)网格资源状态

这是网格监控系统最重要的工作内容，它服务于下面几个方面：

·为用户提供当前资源状态；

·提供历史数据归档和历史数据分析；

·为资源状态预测提供支持；

·为资源调度提供支持。

(3)网格资源监测

这里说的监测，不仅仅是探测资源的存在，还包括检测资源的健康程度。不仅仅获得数据，而且要通过某些手段处理资源状态的异常情况，提高系统资源的可用程度和系统的健壮性。

(4)网格资源控制

监控系统不仅仅读取资源的状态信息，还提供途径对资源的状态进行干预，如恢复错误等。资源控制相当于获得写的能力，与读结合在一起，形成闭环系统，其将增加控制系统的复杂程度。

(5)支持上层应用

在同一网格资源信息模型里，网格资源分为密切关联的物理资源和逻辑资源两种。在改变某个资源对象时，要同时兼顾与之关联的其它资源信息，才能保证资源信息数据的一致性。

网格监控的目标是对分布在不同地理位置的计算资源透明访问以及对它们运行状态的统一集中监视和管理。计算资源仅仅是一个抽象的概念，其指的是网格中存在的与计算相关的网络系统、主机系统和应用系统。按照通用计算机的体系结构，把监控对象分为三类：

(1)系统监控对象

主机硬件资源使用状态，例如，CPU负载，主机负载，内存使用情况，硬盘占用率，用户数量等。

(2)网络监控对象

域间和节点间的通信状态，与其他节点的通信延迟，数据传输带宽，路由状态等。

(3)应用监控对象

过程设置运行状态，多为面向应用的，例如，网络服务器，数据库服务器，监控服务器和其他事件服务器等。

以上三类对象涵盖了网格系统中可能出现的大多数监控对象类型。

监控事件是与监控对象紧密联系的，是在特定时间由特定输入触发的输出结果。监控对象将其状态信息以监控数据的形式在监控系统中传播的行为就是我们通常所说的监控事件。监控系统中一般有两种性质的监控事件：预警和崩溃。

(1)预警

预警事件是指可能对系统造成威胁(比如可能造成系统崩溃)，但是还没有切实危险，只是监控对象的状态超过预设的警戒值的情况。可以根据模拟仿真给出警戒值。一旦系统发出警告，监控系统应该立刻做出反应。调用相应的传感器(sensor)获取监控对象状态，并向监控管理系统发回预警事件。

· 自动根据网络状况进行调度(可以考虑某些算法)，并将预警事件写入日志，历史数据存入数据库。

·管理员可以设定对某些事件采用手动处理。遇见此类事件，系统不进行处理，而是通知管理员，通过人工判断处理方法(此处亦可设定时限，如果时限内管理员没有进行处理，则系统自动按默认方式处理)。

(2)崩溃

崩溃事件是指以下情况：

·对于任何输入没有输出；

·被监控事件停止，网络对象不可达；．

·对象状态超出限制如存储溢出。

遇到崩溃事件，可向监控管理系统发回崩溃事件，将其写入日志，并将历史数据存入数据库，节点信息显示为不可用。记录该节点崩溃前原运行进程的信息和运算有关结果，将任务分配给其他节点。实现任务的自动迁移，做到对高层用户透明，尽可能减少用户的损失。

## 调度

网格任务是指在网格环境中为了解决某一问题而运行的对象。根据任务之间的相关性可以把网格任务划分为独立任务和关联任务。独立任务又称元任务(meta-task)，它们之间没有时间依赖性或数据依赖性，每个任务可以独立的执行。而关联任务之间则存在着一定的时序关系和因果关系，必须按照一定的先后顺序来执行任务。

按照任务对完成时间的要求，又可把网格任务分为非预留任务和预留任务。前者对完成时间没有明确的要求，而后者则要求任务在截至时间(deadline)之前必须完成。预留任务一旦开始运行就不允许被其他的网格任务打断，而非预留任务可以被预留任务打断。非预留任务又可分为可抢占式任务(preemptable task)和非抢占式任务(non-preemptable task)。前者在运行过程中如果被打断，可以在以后的某个时间从断点处继续运行，而后者一旦被打断则必须从头开始运行。

网格资源是指能够处理网格任务，且隶属于网格系统，服从网格调度的实体。比如，可以进行计算的处理器，可以传输数据的网络连接。而任务调度是指网格任务及其执行次序在选定网格资源上的映射。通常能够处理用户任务的网格资源不止一个，这些资源的性能、使用代价以及提供的服务质量可能会有很大差异，因此需要一定的策略选择合适的资源来执行用户任务。给用户任务寻找合适的计算资源的过程称为资源匹配(matching)，或资源分配。另一方面，可能会有多个任务同时选择一个资源，需要解决多个任务在该资源上执行的先后顺序问题，这一过程叫做次序调度。

在已有的网格系统中，任务调度系统结构不尽相同，一个基本的任务调度模型如下：用户首先向网格调度器提交自己的任务，指明自己的资源需求。网格调度器通过网格信息服务查找所有能够满足用户需求的网格资源，代价估计模块根据用户的任务信息和网格资源信息预测出作业在不同资源上运行所需花费的代价(例如，作业执行时间)。然后任务调度器根据一定的调度策略为用户的任务选择最合适的网格资源，并把任务提交给该资源所在管理域的本地资源管理系统。任务调度器通过监控模块跟踪作业的执行情况和资源的运行状态，当任务执行完毕后把运行结构返回给用户。值得注意的是，由于网格资源具有自治性，网格任务调度不能剥夺资源本地的任务管理权利，它与资源本地的任务调度要协同工作，共同完成对用户提交的网格任务的调度。

网格任务调度方式可分为在线方式(on-line mode)和批处理方式(batch mode)。在线调度方式中，任务一旦到达调度器就要为其分配资源，即调度器是根据任务的到达情况而工作的。这种模式能够及时对用户的请求做出响应，实时地执行任务调度。在批处理调度方式中，任务在到达的时候调度器并不立即给它分配资源，而是将其放到一个任务集合中。当网格调度事件(scheduling event)被触发时(如达到一定的时间间隔或者任务集合达到一定规模)网格调度器再对整个任务集合进行调度。

在线调度方式适合分布式的调度结构，且任务调度策略容易实现。但由于每一个任务到达时调度器就要查询一次资源及任务的信息，因而调度效率不高，适用于网格用户少并且任务提交不频繁的情况。另外，由于每一次任务调度都仅针对于当前任务，这使得调度器只能够做出局部最优的调度。在批处理调度方式下调度器只需要获得一次资源及任务的信息就可以对一个任务集合进行调度，因而调度的效率较高，适用于网格任务到达较为频繁的情况。对任务集合进行调度可以考虑任务执行之间的相互影响，更容易做出相对较优的调度安排。

主要的网格调度器如下：

1) AppLeS

AppLeS是加利福尼亚大学的网格计算实验室发起的一个研究项目，其研究对象是面向应用层的调度。AppLeS使用Network WeatherServices(NWS)来动态地预测资源负载和可用性。用户提供有关应用性能偏好和平台偏好的信息。AppLeS开发了若干个与应用类型相关的模板，对应不同的性能模型和调度策略。用户可以针对开发的应用程序来选择对应的模板信息，确定具体的性能指标和调度策略。AppLeS缺乏全局协调性，即可能有不止一个AppLeS智能代理在同时做出相同的调度，这会造成调度冲突，使得资源为每一个应用提供较低的性能。AppLes调度器的缺点在于它不能够提供对应用的服务质量保证。

2) PBS

PBS(Portable Batch System)是一个由NASA开发的灵活的批处理系统。它被用于集群系统、超级计算机和大规模并行系统，支持批处理任务、串行任务和MPI、PVM并行任务。PBS的独立的调度模块允许系统管理员定义资源和每个任务可使用的数量。调度模块存有各个可用的排队任务、运行任务和系统资源使用状况信息。在任务调度策略上，PBS提供了默认的FIFO调度策略。但它的调度策略易于修改，即系统管理员可以方便地实现自己的调度策略以适应不同的计算需要和目标。PBS具有一定的任务监控能力和安全机制，但没有解决单点故障、不提供检查点操作和进程迁移，并要求对资源具有完全的控制。PBS作为本地任务管理系统，可以充当网格系统中的底层调度器；同时也可以把PBS作为网格计算环境的入口，使用GRAM协议通过提交PBS任务实现对Globus资源的调用。

3) Nimrod-G

Nimrod-G是澳大利亚Monash大学的一个网格研究项目，它使用了基于经济学原理的资源管理机制。Nimrod-G是一个支持网格应用的Nimrod改进版，它使用了Globus支持的功能，比如动态发现可用资源，但Nimrod-G的构架具有很好的扩展性，可以基于除Globus以外的网格中间件。它由四个关键的部分组成：任务管理引擎、调度器、分派器和智能主体。任务管理引擎允许嵌入用户自定义的调度器，定制应用或问题分析环境；分派器利用Globus把Nimrod-G智能主体部署到远程资源上以管理分派任务的运行；调度器则可以根据用户服务质量需求中说明的能力、费用等把任务分配给合适的资源和服务，能够做出基于最迟期限(Deadline)和预算(Budget)的调度。Nimrod-G的计算经济目前还不是十分完善，它缺乏一种定义资源费用的全面策略。Nimrod-G的研究成果还包括GridSim[48]T具集，该工具集支持基于QoS的应用调度的建模和仿真，可用于调度算法的性能评价。

4) Lemon

Legion发起于1993年，是弗吉尼亚大学的一个基于对象的元计算系统软件项目。它的目标是为与用户提供单一、一致的虚拟机器模型。它支持透明调度、数据管理、容错、站点自治和多种安全机制。在Legion中，一切软件实体和硬件实体都是对象。每一个对象由它的类对象定义和管理，类对象除了定义实例以外，还制定管理策略。Legion规定了对象交互的消息格式和高级协议，通过对象的方法来描述其接口，但没有限制具体的编程语言和通信协议。

5) Condor

Condor是威斯康星．麦迪逊大学的研究项目，它是一个计算密集型的任务管理系统，支持在大规模分布式资源集合上的高吞吐量计算。Condor采用分层体系，支持串行和并行应用。它使用分类公告语言ClassAd来描述用户的资源请求，采用半结构化数据模型描述资源，提供相应的查询语言以允许公告发布方在资源请求中说明约束条件。在Condor中用户代理管理一个有关应用描述的队列，并把资源需求发送给匹配器(matchmaker)。每一个资源有相应的资源代理，资源代理在实现资源所有者所指定的资源使用策略的前提下把资源可用信息发送给匹配器。匹配器根据一定的匹配约束(比如用户优先级)来实现系统范围的任务和资源的匹配。Condor可以通过使用一些特有的特性，例如检查点、任务迁移以及远程系统调用来有效地利用非专用资源。Condor采用批处理方式集中对任务进行调度，不提供服务质量支持。

## 资源管理

网格资源管理是网格系统的核心组成部分。从早期的计算网格到现在的服务网格，网格资源管理技术伴随着网格系统的研究不断发展。如何进行资源管理、如何进行任务调度、如何对资源进行动态监控等，已经提出了很多技术和方法，可以有效地对网格中的资源进行管理。主要有Web Service 技术、LDAP 轻量级目录访问协议、Globus 资源管理、Condor系统资源管理、DataGrid、AppLes 项目、Legion、Nimrod-G 网格资源代理等。

1) Globus 资源管理

Globus项目是著名的网格计算项目，其最初目的是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。随着项目研究的发展和深入，目前已将Web服务技术融入到Globus项目中。Globus对信息安全、资源管理、信息服务、数据管理等网格关键技术进行了深入的研究。这里介绍Globus 在资源管理方面的研究。

Globus 资源管理由GRAM、RSL 和DUROC 组成，GRAM（Grid Resource Allocation Manager）是资源分配管理者，位于Globus 资源管理体系结构的底层，主要处理资源请求、资源分配和管理活动等任务，并根据计算资源的情况把资源更新信息发送给监控和发现服务MDS；RSL 为网格资源规格语言，用来对网格资源分配过程中的任务和所需的资源进行描述，它是一种统一的资源任务描述语言；DUROC 为动态更新请求在线协作分配，GRAM 和DUROC 都使用RSL。为了实现信息服务中的数据生成、数据分布、数据存储、数据搜索、数据查询和数据显示等功能，资源监控与发现服务MDS 提供了一个标准的、可配置的信息提供者框架，称为网格资源信息服务GRIS（Grid Resource Information Service）。GRIS 基于LDAP 协议，并提供资源相关信息。GRIS 可响应网格计算环境中其他的信息查询请求，并且可配置成聚集目录服务（如GIIS），以接收其他GRIS 上的资源信息。GRIS 与信息提供者之间通过API 进行通信。在缺省情况下，一个GRIS 服务可以自动配置，并且在网络端口2135 上进行监听。MDS 还提供GIIS，通过GIIS 可构成一个层次结构的简单索引目录框架，形成分布式的信息服务。这样只要一个简单的命令就可从多个GRIS 中得到信息。一个GIIS 可配置为单节点范围或多结点合作范围的信息服务器。GIIS 可提供信息集合查询并支持对多个GRIS 的有效查询。

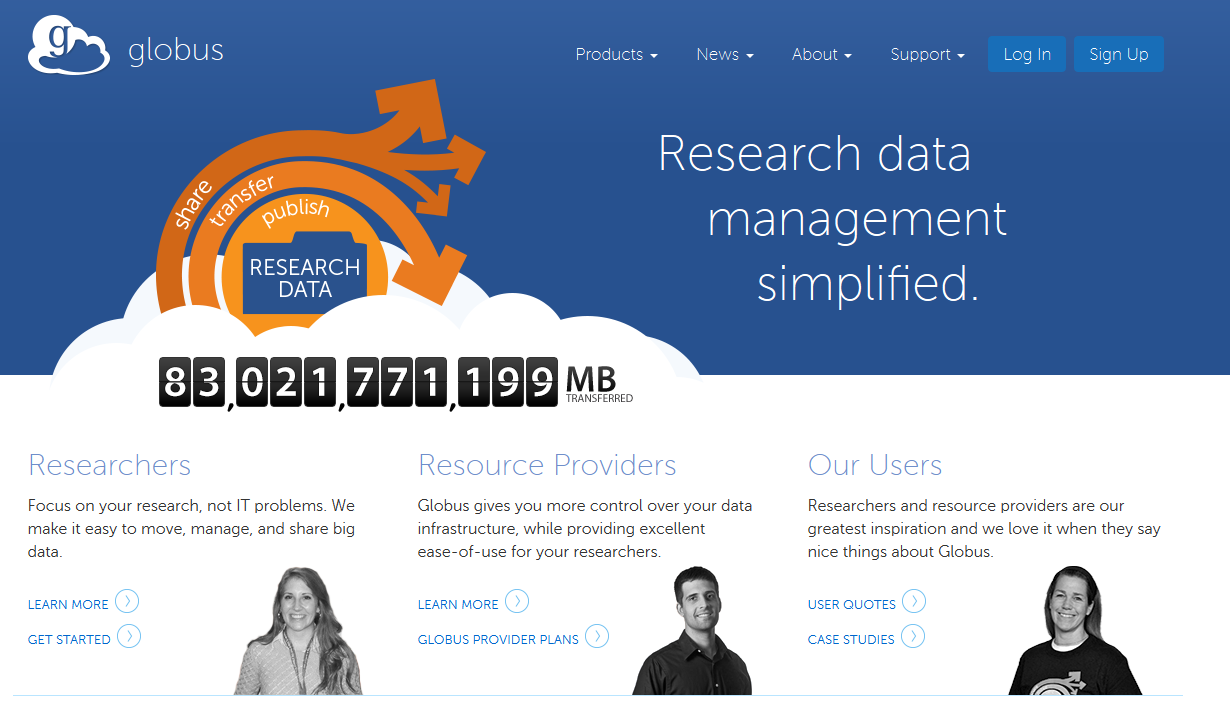


图3 Globus 资源管理系统主页

2)Condor 系统资源管理

Condor 系统是美国威斯康辛大学研究开发的一个计算密集型作业的资源管理系统，它的核心工作方式是使用Condor 池中的空闲资源来获得高吞吐量计算。Condor 环境可以为连续的和并行的应用程序提供强力的和灵活的资源管理服务。它能管理个人计算机、工作站和机群的计算机集合。该系统最初是建立在Unix 系统之上的分布式批处理系统。用户可以从Condor 池中的任何一台机器提交任务，系统自动从Condor 池中查找最合适的空闲机器运行任务，通过将大型计算任务派至远程空闲计算机执行的方式，把空闲的处理机资源分配给其他用户使用，从而实现负载的共享。Condor 系统由作业管理和资源管理两部分组成。作业管理负责管理用户作业，资源管理负责监控资源的可用情况，进行资源的分配调度等。在资源管理中，Condor 系统不同于以往的分布式计算系统的集中控制模型。系统中的资源拥有者拥有对资源使用的最高优先权，他可以根据自己的实际情况，制定资源的使用策略，其他用户提交给系统执行的作业不会妨碍资源所有者对其资源的使用。事实上，Condor 是一组资源和资源请求的集合。每一个Condor 池有一个中央管理器（Central Manager）用于收集信息，进行资源与资源请求匹配。

3) AppLes 项目的资源管理

AppLes(Application Level Scheduler)项目由美国加州大学、圣地亚哥大学开发，主要开发用于在生产网格的个人应用的调度代理。它使用网络气象服务（Network Weather Service NWS）来监控资源性能的动态变化。AppLes Agent 在选择一组有效的资源和资源配置时，使用静态或动态的应用和系统信息。AppLes Agent 可以与其他的资源管理系统如Globus、Legion 等交互，执行应用程序任务。应用程序嵌入AppLes Agent 中，因此可以在网格中自我调度。AppLes 关注的核心是调度，它沿袭网格中间件系统支持的资源管理模型。AppLes 调度器是把用户任务映射到资源上的核心部件，本地资源调度器则用来执行实际的任务运行。AppLes 已经被应用到许多应用领域，如磁流体动力学、基因顺序比较等。

4) Legion

Legion是一个面向对象的网格操作系统，它提供一个软件基础设施，使异构、地理上分布的、高性能的计算机能够进行无缝的交互。Legions 提供给用户一个单一的、一致的虚拟计算机。Legions 的对象代表Legions 的组件，这些对象由类对象定义并管理，类对象创建新的实例、调度他们的运行、激活或解除对象并且提供客户对象的状态信息。每个对象是一个活动的进程，它接受来自系统中其他对象对它的调度。当需要和其他对象进行通信时，对象可以被自动激活。Legions 定义了对象交互的API，但没有指定编程语言和通信协议，所有Legions 系统中的类都是通过Legions Class 进行分级组织。Legions 支持主机负载控制机制，提供资源预留能力以及应用级的调度器来进行周期或批量任务调度。Legions 资源管理体系结构采用等级化、分布式的策略。

5) Nimrod-G 网格资源代理

Nimrod-G 网格资源代理[13]可以管理和操纵计算网格中的任务管理，它采用经济模型进行资源管理和调度，用户通过公布的参数模型语言或GUI 进行参数设置。Nimrod-G 对网格节点进行资源发现、资源交易、资源调度、资源分段运输、结果收集以及将结果提交给用户。Nimrod-G 在资源管理上遵循等级化和计算市场模型，它利用网格中间件系统提供的服务来进行资源发现，通过GRACE 基础设施提供的计算经济服务来支持资源预留和QoS。QoS 可以由用户指定，QoS 包括最后期限、预算和首选的优化策略等。资源调度策略是面向应用的，可以根据用户指定的参数进行调整。

## 安全

网格技术有着广阔的前景和发展空间，但由于它的研究还处在初期阶段，尤其在网格技术的研究向商业服务转变过程中，就存在有许多关键技术问题需要研究解决的，尤其是网格安全问题。

在传统的Intemet中一般提供的安全保障机制是访问控制和安全通信。访问控制是用来保护各种资源不被非授权使用，而安全通信保证数据的保密性及完整性。在网格环境中，由于节点异构、资源规模动态可变、可扩展的特点，各种资源都是动态的连接到Intemet上，不同节点之间是通过了Intemet通信。用户通过Intemet向网格提交计算和监控管理任务时，都是可以动态的加入或撤离。这就使得网格安全与传统的Intemet安全相比，所涉及的范围更广，解决方案也更加复杂。从而使得现有安全技术不能满足共享化的更高要求，迫使一种新的安全体系结构出现。新的安全体系结构在采用了现有的安全机制技术，不仅提供各种资源的满足了用户安全、高效的使用网格的要求，也提供了高效的安全保障在节点之间通信中，消除节点间的安全结构和机制的差异。从网格技术的长远发展来看，具有一个结构良好、可靠和可扩展的安全体系结构对网格起着非常重大的作用，它将会是网格技术能够实际运用于现实世界的强有力的保障，对网格技术的发展有着重要的意义。

网格是跨越多个管理域的异类计算资源和服务的动态集合，提供了大规模的分布式异类的资源共享和一体化信息服务。在网格环境中，各种资源及不同网格之间的通信都是动念地连接到Intemet上的，不同的网格结点之间的通信也是通过Intemet来完成任务的。所以，网格环境必须具备预防各种非法攻击和非法入侵的能力，如果意外受到攻击和入侵，网格环境要有相应策略和措施来维护系统正常运行，同时能保证信息的完整性和安全性。

自从1995年美国的I—WAY实验中提出网格计算的概念以来，就非常重视网格环境的安全，相继研究出台了一系列网格安全策略，1997年由美国Argonne国家实验室及其科研机构等联合开发的Globus项目，它是由多个机构联合开发的项目，力图在科学计算领域和商业领域对各种应用进行广泛的、基础性的网格环境支持，实现更方便的信息共享和互操作。目前网格技术并未形成真正的安全标准，近两年全球网格论坛GGF(Global Grid Forum)正在致力于建立网格信息基础设施、安全策略和技术等标准。

目前网格环境中所面临的安全问题包括相互影响的主机环境之间的信任关系、不同主机环境之间协同工作能力、现有技术和安全协议的集成及扩展。

1)相互交互的主机环境之间的信任关系。由于网格的动态特性，有些情况下不可能在应用程序执行前预先在这些域中建立信任关系。而网格环境是跨越多个安全域，信任的建立过程对每个会话来说或许是一次性的活动，也有可能对于每一次请求都要动态地进行评估。网格环境中的信任关系非常复杂，需要支持动态的、用户控制的配置管理。

2)不同主机环境之间协同工作能力。穿越多个域和主机环境的服务需要能够互相影响，协同工作。主要表现在下面几个层面：

(1)策略层：为了进行安全的会话，参与协同工作的每一方必须能够详细说明它要求的任何策略比如隐私权策略，要求使用特定的加密算法，各方才能尝试建立安全的通信信道和有关互相认证、信任关系的安全语义。

(2)身份鉴定层：在进行交互的过程中需要有在不同域间相互鉴别用户身份的机制。为了在安全环境中成功实现跨越多个域，必须要实现身份和信任的映射，这可以通过建立相应的身份代理服务来完成。

(3)协议层：即消息传输过程中需要保证消息的完整性和保密性并对消息进行数字签名，这就需要安全的域间交换信息的机制。

3)现有技术和安全协议的集成及扩展。为了获得成功，网格安全体系结构需要过度到对现有安全体系结构和跨平台、跨主机模式的集成。这意味着该体系结构可由现有的安全机制来实现，同时，要有可扩展性和可集成性。所以现有的安全架构不可能在一时被完全取代，反而被认为安全可靠的认证机制会继续使用。这三方面的安全性问题之间都是相互影响的，即每一个问题的解决通常依赖于另一问题的解决。

目前技术有：密码、公开密钥体系、安全Socket层、SOAP加密等。

轻量目录访问协议LDAP(Lightweight Directory Access Protoc01)是一个独立的开放网络协议标准，基于TCP／IP定义的一个目录服务标准。资源信息在LDAP目录中存储在一个多层次的树型结构中。LDAP和普通的数据库在实现上有差别，在性能上也有所不同，它的主要优点包括：(1)LDAP是跨平台的协议，可以运行在任何平台上。(2)对LDAP的读操作的完成速度比普通的数据库访问要快得多。LDAP专门针对读操作进行了优化，适用于频繁读取的操作。(3)LDAP服务器可以是分布的，用户访问到信息可以是本地的LDAP服务器，也可以是全局的，各地服务器之间可以通过LDAP内部机制很容易地实现同步。

目录是一个以一定规则排列的对象的属性集合，是一个存储着关于对象各种属性的特殊数据库，这些属性可以供访问和管理对象时使用，类似于电话簿和图书馆卡片分类系统。目录服务是指一个存储着用于访问、管理或配置网络资源信息的特殊数据库，它把网络环境中的各种资源都作为目录信息，在目录树结构中分层存储，对这些信息可以存储、访问、管理并使用。网络中的这些资源包括用户、各个应用系统、硬件设备、网络设备、数据、信息等。

在目录服务系统中对象可以根据名字或功能、属性访问，而不是根据机器地址、文件服务器名字和E．mail地址等访问

LDAP的功能模型涉及以下三个方面：询问、更新、身份验证。

中间件从LDAP服务器中获得监控对象的位置及刷新频率，然后定时向驻留在主机中Agent组件发出监控事件请求，Agent组件根据请求调用相应的移动Agent以获取监控对象的状态，并向GridMonitor中间件发回监控事件。GridMonitor中间件将监控事件存储于LDAP服务器。来自客户端的监控事件请求被GridMonitor中间件分为定时事件请求和实时事件请求，前者是从LDAP服务器中获得监控事件数据，后者直接从移动Agent取得监控对象的实时状态，最后由GridMonitor中间件将结果返回给客户端。GridMonitor中间件的用户管理模块可根据存储在LDAP服务器中的用户权限信息决定客户端的权限。

# 多Agent在网格中的应用

自2000年6月在爱尔兰软件工程领域顶级会议ICSE召开了第一届AOSE的国际会议，标志着基于Agent的软件工程（AOSE）的研究已经成为研究热点。AOSE主要研究系统地应用软件工程领域的理论与原则创建基于Agent的软件系统。

由于Agent本身具有的自主性、积极性、反应性、社交能力及移动性非常适合用于网格计算中；并且多Agent系统所具有的为共同目标，联合多Agent进行协作和Agent独立自主不必集中控制的特点，也非常适合网格计算的需求。

在网格应用开发过程中，采用Multi-Agent技术能够提高网格系统的智能性、灵活性和健壮性。按照网格之父Foster的观点：一个系统之所以被称为网格，是因为系统在非集中控制的环境中，通过标准的、开放的和通用的协议和接口，协同使用资源，并提供高质量的服务。Agent具有自主性，无需进行集中的控制就可以自主地决定自己的行为。目前，Multi-Agent技术在理论上已经有很多的成果，有一整套完备的、标准的、开放的和通用的Agent接口及协议，很容易就能与现有的网格技术整合。Agent具有交互性，相互间可以进行协作，协同完成任务。此外，可以将较复杂的任务封装在Agent中，由多个Agent相互协作，可以完成单个Agent或其它软件不能完成的任务。由此可见，Agent的特性完全满足网格应用与开发的要求。

基于Multi-Agent的网格系统在逻辑上可以分为四个层次，如图1所示。系统中，客户端的任务主要是确定用户的操作请求，资源端负责处理用户请求，网格中间件致力于整合各种网格资源。网格资源请求Agent作为用户的代理，负责协调用户与网格资源Agent间的交互。其目的是使网格资源与用户之间无需进行直接联系，用户可以更加方便地使用所需的资源。

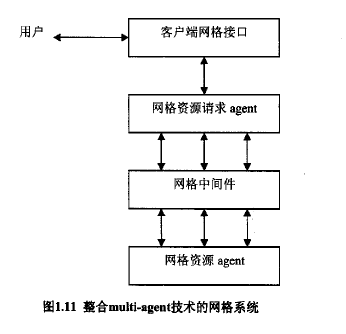


图4 结合Agent技术的网格系统

用户通过客户端的应用程序提出服务请求，激活网格资源请求Agent。网格资源请求Agent依据用户提出的要求，在众多的资源提供者中进行筛选。满足用户要求的资源可能会有多个，网格资源请求Agent代替用户做出最终决策，确定一个最佳资源提供者。满足要求的资源也可能没有，此时，网格资源请求Agent可将若干个资源进行组合，以此得到满足用户要求的资源。任务完成后，再由网格资源请求Agent将最终结果传给客户端。

网格资源Agent对资源请求进行分析处理，智能、主动地使用网格资源，为用户提供所需的资源。网格资源Agent还可以根据网格环境、用户要求，以及自己的信念和目标，与网格资源请求Agent进行交互、协商，并自主地决定是否提供资源，及如何提供资源等。网格资源Agent不仅接受网格资源请求，为用户提供网格资源，还可以发出资源请求或协作请求，与网格中的其它资源Agent协同完成任务。

利用Multi-Agent技术的资源查找和定位的能力，基于Multi-Agent的网格系统具有适应性的资源调度能力，能够在一个高度动态变化的环境中发挥作用。网格系统中资源的分布、定位、选择及访问等相关问题都被屏蔽起来。用户只需进行接入网格系统的简单操作，就可以使用其中的各种资源。Multi-Agent技术本身的容错功能也使得基于Multi-Agent的网格系统比其它一些网格系统(如Globus，Legion等)拥有了更强的容错能力。当网格中某个正在提供资源的节点忽然失效时，Agent可以保存当前的工作状态，迁移到另外的节点上继续完成承担的任务；或者，Agent可以把当前的请求任务转交给其它节点完成。而这一过程对用户来讲完全是透明的，客户端程序的执行不会受到任何影响。基于Multi-Agent的网格系统还能处理系统运行中出现的一些意外突发事件，比如网络故障、硬件设备故障等。系统通过在特定网格节点上保留原始备份并实施跟踪等措施，降低网络不稳定等因素对系统性能的影响，从而保证系统运行的稳定性和可靠性。

Agent的移动性和自主性使得基于Multi-Agent的网格系统具有更强的负载均衡能力。基于Multi-Agent的网格系统能将网格节点上的某些功能及状态迁移到其它空闲的节点上，实现系统中动态的负载均衡以及透明的容错功能。而且，这种负载均衡方式对系统中节点的拓扑结构没有任何限制，能够根据指定的资源使用方式选择特定的资源提供节点。这比其它的负载均衡实现方案能更有效地提高系统的整体服务性能，使网格系统更加智能化。

近年来已经出现了一些研究实例将Multi-Agent技术用在网格中。例如，CoABS(Control of Agent-based System)网格提供了一个统一的、异构的分布式计算环境，在这个环境中计算资源能够进行无缝的连接，而且CoABS的网格体系结构还提供广域的异构架构的集成。ACG(Advanced Computation Group)将网格环境中的所有实体都封装为Agent，实现了网格计算资源和服务的高层统一管理，为用户提供一个一致的、透明的访问服务的接口。Rana和Walker提出的PSEs(problem solving environments)支持特定算法和不同工具间的互操作，每一个Agent代表集成服务和资源，不周的PsE Agent的行为规则可以根据与其它Agent或环境的交互进行修改。

从本质上讲，网格技术是分布式计算技术和高性能计算技术结合的产物，传统的分布式计算技术缺乏动态灵活性，如：对象委派机制、进程迁移机制等，而移动Agent技术的动态迁移能力、异步性以及智能性等，为构建广域范围的网格计算环境提供了独到的解决方法。在开放网格服务体系结构下，移动Agent技术应用很广泛，其中包括：

(1)信息发布：将移动Agent技术应用到当前的信息发布机制中，主要通过各个网格服务在注册服务中注册自身的信息来完成发布。

(2)N格服务助理：利用移动Agent的移动及恢复执行的特性，可以代表网格服务在网络中完成一定的任务，成为网格服务的助手。

(3)监视和通知：作为被控制方，由于状态的多样，是没有义务随时向控制方报告其状态的。作为控制方，可以派遣专门的Agent到被监视机上负责这项工作，再也不必采取一些被动的措施。

(4)实时控制：主要是对内部及外界的环境变化做出实时的反应。

(5)分布信息查询：当信息资源分布于不同的计算机上时，将各个信息源的信息数据通过网络传递到本地机上，再在本地机上建立应用进行查询。

(6)并行处理：当计算需要大量的处理能力而必须分布到多个处理器上时，支持移动Agent的主机通过网格可以构成巨大的并行计算系统。因此目前将Agent技术与网格计算结合的研究方向，包括：移动Agent技术、软件架构设计、网格监控与管理、对虚拟组织的支持。

其中，网格监控与管理方面，主要是对网格资源进行管理。网格资源指的是所有能通过网格远程使用的实体。网格资源监控与管理就是指对网格环境中的资源进行有效地描述、组织和管理。其中主要方法包括：Web Service 技术、LDAP 轻量级目录访问协议、Globus 资源管理、Condor系统资源管理、DataGrid、AppLes 项目、Legion、Nimrod-G 网格资源代理等。

虚拟组织，即指分布式虚拟组织，它指代网格中一起协作的人形成的一种自然结构。这方面研究主要集中于网络协商的研究，即在网格环境中，存在资源申请者和资源提供者，而二者都倾向于用最少的代价，获得最大的利益。主要采用的研究方法是以博弈论为主体的相关算法。

而移动Agent，是一种独立的计算机程序，可自主地在网络上按照一定的规程移动并寻找合适的资源，代表用户完成特定的任务。目前大多数研究主要是将Agent应用在网络安全方面，即网格环境具有动态性，信任关系很难预先建立。利用移动Agent，建立层次代理结构。以及资源管理方面，即全局与局部的LADP，将服务请求Agent移动到目标主机，直接访问资源，减少网络通行量和安全隐患。

由于本文主要关注的是多Agent系统架构的研究，下面将重点介绍Agent技术应用在网格系统架构部分的研究。

## 4.1相关学者

* 史忠植

来自：中国科学院计算所智能科学实验室

主要研究方向的变迁：

Data Mining, Granular Computing

Semantic Web

Probabilistic Latent Semantic Analysis, Dynamic Description Logic Mobile Grid

相关项目：主体网格智能平台AGrIP(Agent Grid Intelligence Platform) 由底层支撑平台，中间件层和应用层组成。

相关文章：Track on Intelligent Computing and Applications, Shifei Ding Zhongzhi Shi, Neuro computing(2014)

Parallel extreme learning machine for regression based on MapReduce, Qing He Tianfeng Shang Fuzhen Zhuang Zhongzhi Shi, Neurocomputing(2013)

* 李春林

来自：武汉理工大学计算机科学与技术学院

主要研究方向的变迁：

Grid、Grid User

Computational Grid

Grid Computing

Mobile Grid

相关项目：基于下一代互联网的计算网格资源管理理论及技术的研究，国家自然科学基金项目，2005.1~2007.12

下一代服务网格中基于多维QoS的资源调度及其优化技术的研究，国家自然科学基金，2008.1~2010.12

主要研究点：

1. 将移动Agent，微观经济学模型与计算网格资源管理及调度问题相结合，提出了基于移动Agent的计算网格资源管理模型体系结构框架
2. 提出了分布式多维QoS约束的计算网格资源调度优化策略，建立了一种新的基于非合作博弈的多维QoS资源调度优化数学模型
3. 设计了多维QoS约束的动态多播路由通信协议
4. 基于分解优化策略的计算网格资源分配算法

相关文章：A Market based Approach for Sensor Resource Allocation in the Grid. Li Chunlin Huang Jin Hui Li Layuan，Informatica (Slovenia)(2012)

* Giancarlo Fortino

来自：the Department of Computer, Control, and Management Engineering

Antonio Ruberti at Sapienza University of Rome

主要研究方向的变迁：

Rtp, Java

Mobile Agent, State Charts

Sensor Node

相关文章：Editorial: Integration of Cloud computing and body sensor networks，Giancarlo Fortino, Mukaddim Pathan, Mukaddim Pathan, Future Generation Computer Systems(2014)

An agent-based approach for the design and analysis of content delivery networks, Giancarlo Fortino, Wilma Russo, Marino Vaccaro, Journal of Network and Computer Applications(2014)

BodyCloud: A SaaS approach for community Body Sensor Networks, Giancarlo Fortino, Daniele Parisi, Vincenzo Pirrone, Giuseppe Di Fatta, Future Generation Computer Systems(2014)

* Stefano A. Cerri

来自：Univ. of Montpellier II

主要研究方向的变迁：

Strobe model Artificial Agent

Human Learing Agent Communication

Multi-agent system

相关文章：A Causal Graph Based Method to Evaluate e-Collaboration Scenarios

Raoudha Chebil, Wided Lejouad Chaari, Stefano A. Cerri, WETICE '13 Proceedings of the 2013 Workshops on Enabling Technologies

Dynamic Difficulty Adaptation in Serious Games for Motor Rehabilitation., Nadia Hocine, Abdelkader Gouaich, Stefano A. Cerri, GameDays(2014)

* Clement Jonquet

来自：Stanford Center for Biomedical Informatics Research (BMIR), Stanford University

主要研究方向的变迁：

Strobe model

Agent communication, multi-agent system

Grid, web service

相关文章：Towards a Mixed Approach to Extract Biomedical Terms from Text Corpus, Juan Antonio Lossio Ventura, Clement Jonquet, Mathieu Roche, International Journal of Knowledge Discovery in Bioinformatics(2014)

NCBO Resource Index: Ontology-based search and mining of biomedical resources, Clement Jonquet, Paea Lependu, Sean Falconer, Adrien Coulet, Web Semantics: Science, Services and Agents on the World Wide Web(2011)

## 4.2 国际会议

表1 与网格相关的国际会议

|  |  |  |
| --- | --- | --- |
| 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing | Shenzhen  China  Nov 3  2014 | Applications, including eScience and eBusiness Applications  Architectures and Fabrics  Distributed and Large-Scale Data Access and Management  Peer to Peer Protocols in Grid Computing  Network Support for Grid Computing  Middleware and toolkits for Grid Computing  Creation and Management of Virtual Enterprises and Organizations  Resource Management, Scheduling, and Runtime Environments  Web Services, Semantic Grid and Web 2.0 |
| Conference Grid, Cloud & High Performance Computing in Science | Cluj  Napoca  Jun 15  2015 | Theoretical model for P2P, Grid, Cloud and Internet Computing Systems  Programming models, tools, and environments for P2P, Grid,  Cloud and Internet Computing Systems  Grid and P2P Infrastructures for Data Storage and Data Mining |
| Tenth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing | KRAKOW  POLAND  Jun 26  2015 | Development, evaluation, and  optimization of large-scale cloud, grid, and HPC architectures;  Programming models, systems software and environments for cluster, cloud, and grid computing;  Techniques to schedule jobs and resources on cluster, cloud, and grid |

可以看出网格计算仍然是目前关于大规模系统和并行计算领域的重要研究方向。并且以网格技术为基础产生的云计算技术，也同样是最新的研究热点。

# 基于网格的多Agent系统架构研究

目前将AOSE与网格技术相结合的研究，集中于建立基于网格的新型分布式系统。而在这些研究中，可以分为以下四类：

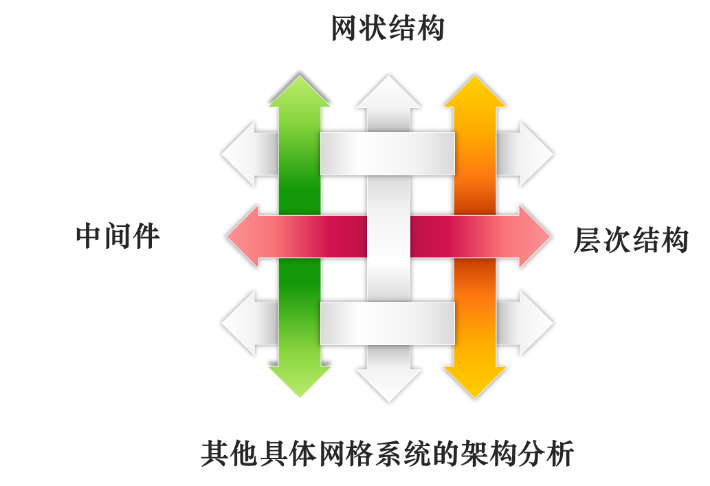


图5 结合Agent技术的网格系统体系结构分类

下文将分别从这四类出发，介绍其中典型系统的体系结构。

## 5.1 基于中间件的体系结构

介绍一个平台软件：HLA\_Grid\_RePast[23]。它主要针对的是大规模分布式仿真问题。建立了一个中间件平台，使使基于RePast的Agent仿真模型可运行在网格上。主要由HLA\_Grid系统和HLA\_RePast系统两部分组成。

其中HLA （High Level Architecture）即高层体系结构，以及其著名的软件支持平台RTI（Runtime Infrastructure）。RePast指Recurslve Porous Agent Simulation Toolkit开源仿真软件，建立基于Javaswarm脚本运行的模型。而为了远程通信，该平台另外采用了一个网络通信技术，即Proxy，网络代理，是一种特殊的网络服务，允许一个网络终端通过这个服务与另一个网络终端进行非直接的连接。

*  RePast

是芝加哥大学社会科学计算研究中心研制的多主体建模工具，她提供了一系列用以生成、运行、显示和收集数据的类库，并能对运行中的模型进行“快照”，记录某一时刻模型的当前状态，还可以生成模型运行过程中状态动态演化的视频资料。

RePast建模相当于设计一个状态机，状态机的核心状态是模型中所有成员的集体状态属性集合。成员则分为底层结构和表层结构。底层结构是各种各样用于运行模型、显示和收集数据等架构的机制；表层结构是设计者创立的模型。底层结构的状态就是模型的种种显示状态、数据收集对象的状态等等；表层结构的状态指模型的描述状态，比如所有主体变量的当前值，模拟环境空间的当前状态值，或他们运行的空间以及别的可能有的各种表层对象（例如聚合“协同”的对象统计值等）。在这种状态机模式下，所有对状态机的改变都通过同一对象界面接口来实现，这个对象界面接口是SirnModellmnl类。这种设计为建模者减轻了学习负担，也简化了在工具包的功能不足时的扩展编程。

*  Proxy

一些网关、路由器等网络设备具备网络代理功能。一般认为代理服务有利于保障网络终端的隐私或安全，防止攻击。

一个完整的代理请求过程为：客户端首先与代理服务器创建连接，接着根据代理服务器所使用的代理协议，请求对目标服务器创建连接、或者获得目标服务器的指定资源（如：文件）。一些代理协议允许代理服务器改变客户端的原始请求、目标服务器的原始响应，以满足代理协议的需要。代理服务器的选项和设置在计算机程序中，通常包括一个“防火墙”，允许用户输入代理地址，它会遮盖他们的网络活动，可以允许绕过互联网过滤实现网络访问。

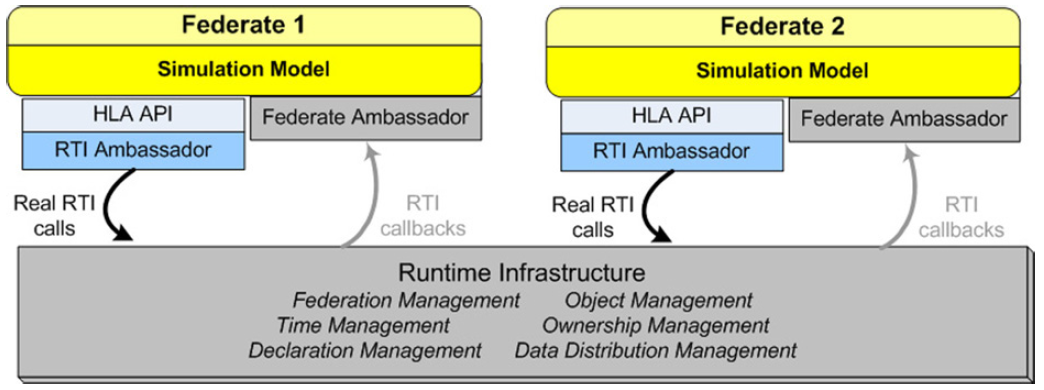


图6 基于HLA的仿真平台

该项目是基于HLA分布式仿真平台的，如图所示，每个被仿真的对象想吃一个仿真联盟，对外提供一个HLA接口和其他联盟进行交互。而联盟之间并不是直接的通信交互而是通过一个中间平台进行的。中间平台就是Runtime infrastructure 即RTI平台，它主要通过Local RTI Component为联盟中的成员提供数据交换和信息同步。

而在该体系结构设计的过程中，考虑到仿真联盟之间的通信问题等，决定采用网格技术扩展基于HLA的仿真体系。提出了基于HLA-Grid的分布式仿真平台。

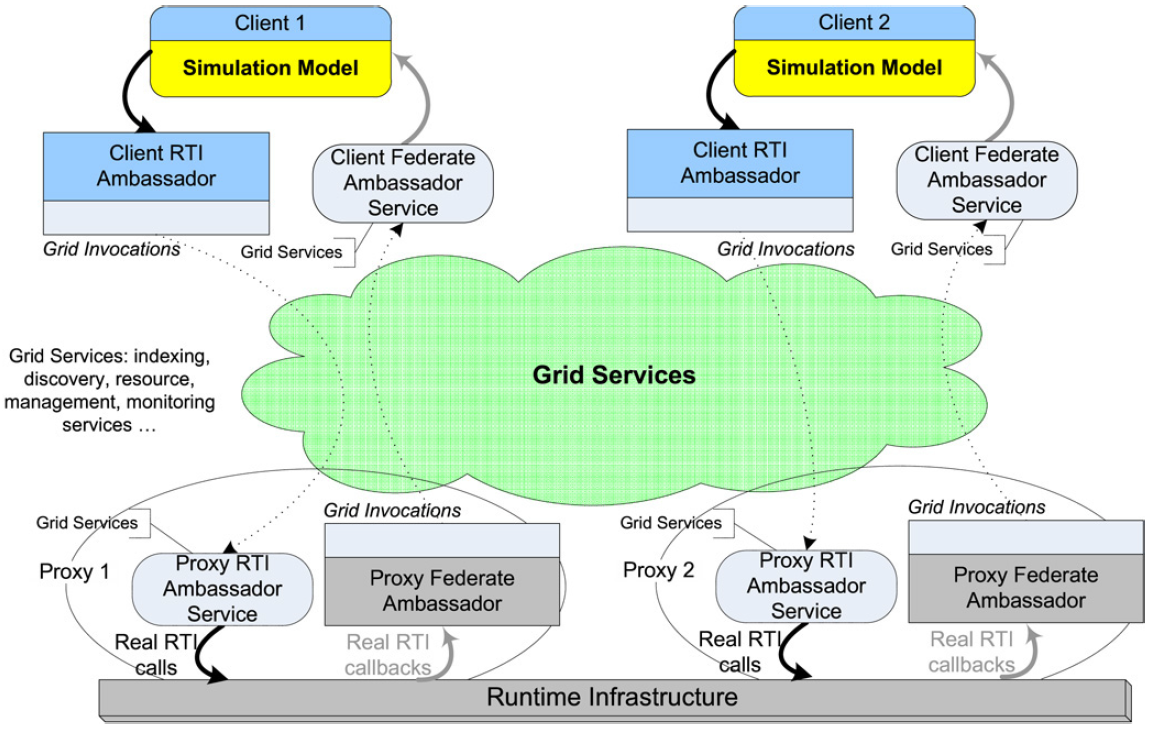


图6 HLA-Grid系统

该平台中每个成员包括一个仿真模型，从用户的角度，被称为Client以及一个proxy，作为Client和其他Client通过RTI交互的代表。而RTI和仿真对象之间通过proxy的远程通信技术，就可以实现横跨网格的扩展，网格则提供如资源管理、同步管理等基础服务。

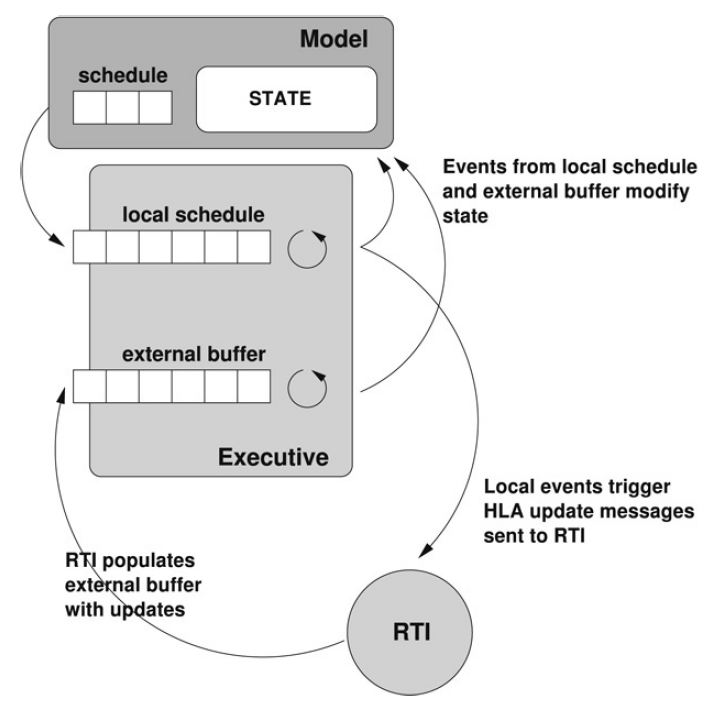


图7 仿真过程

而这个仿真过程与普通的基于HLA的仿真是一致的。仿真模型维持着本地任务和远程任务队列，发现RePast中的仿真事件并且以一种恰当的方式传递给RTI。

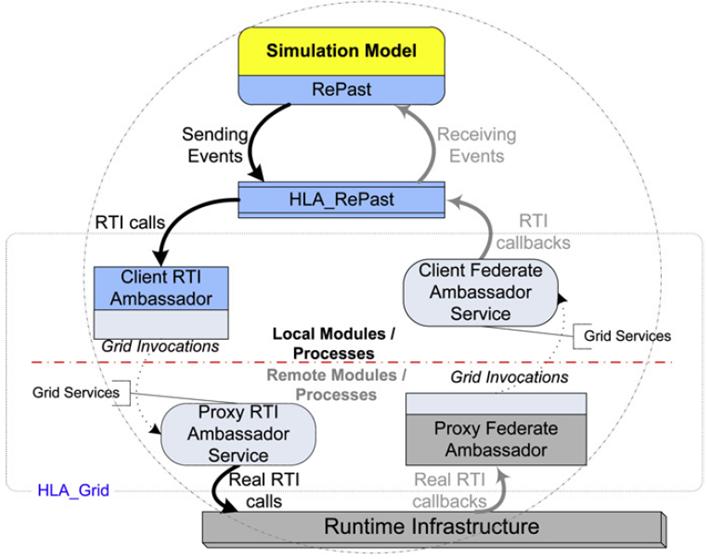


图8 系统结构

整体结构可以分为用户端和代理端。用户端中主要是利用HLA-repast接口将用户的仿真服务需求通过Client RTI接口传递给远程代理。远程代理通过网格，将需求提交该代理端的RTI平台进行处理，从而扩展了系统的规模。

## 5.2 基于网状结构的体系结构

以Agent-based Computational Grid, ACG[24]为代表，介绍这种类型的软件系统。该系统针对的是提升计算网格服务水平的问题。建立了一个基于Agent技术的计算网格，通过Agent技术解决计算网格中的资源发现、定位、调用等问题。主要研究：一种从更高层次对计算资源和网格服务进行管理的体系结构，在这种体系结构将服务和资源都视作Agent。

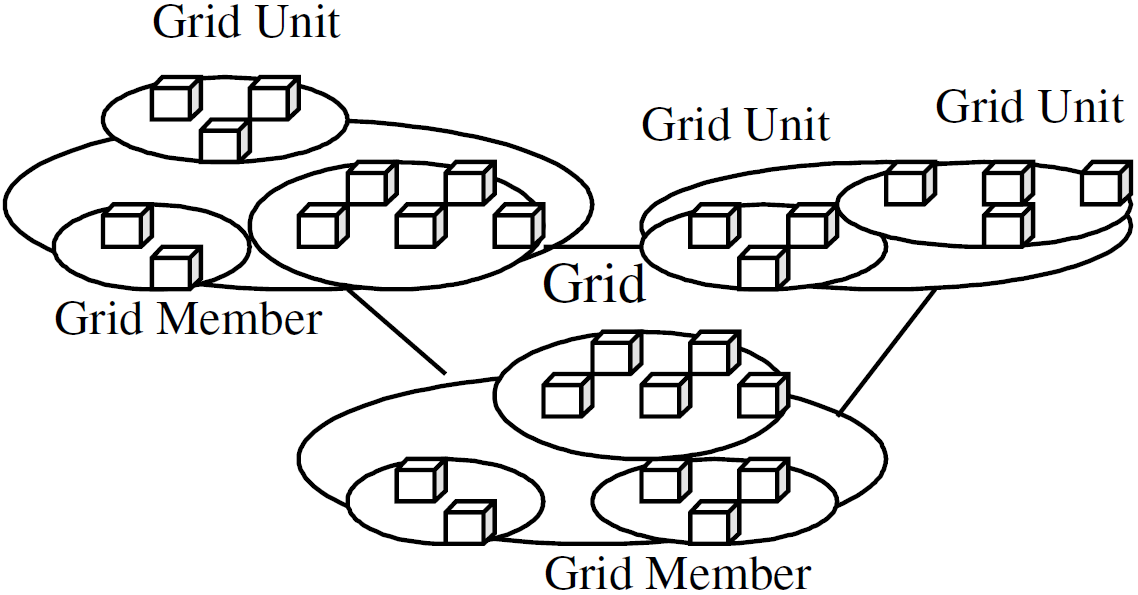


图9 ACG系统结构

如图所示，是该系统的体系结构。其中每个网格成员是对外提供的网格服务的基础单元，并可在损坏时被备份替换。他们是注册在网格系统中的Agent，可以独立异步的处理自己的事物，可自由加入并退出网格，并随时修改自己提供的服务。而若干网格成员将形成网格单元，即对拥有共同利益的member进行组织，实现unit内部自治，Member在unit中发布、查找、获取网格服务，目的在于保障数据传递的安全性。若干网格单元就将形成整体网格，它将提供网格基础服务：服务注册、服务描述、服务查询、服务获取。

整体体系结构主要如下三个模块组成：

普通Agent，封装了由各个成员提供的各种类型的网格服务。Service requestor Agent，作为网格用户的代理，提出用户对网格的使用需求，他们不需要了解网格系统目前的整体情况。基础管理服务，保证系统正常运行的各种基础功能，例如服务描述、注册、获取、查询。

## 5.3基于层次的体系结构

以Agent Grid Intelligent Platform[25]为代表，介绍这种类型的软件系统。该系统针对的是提针对分布式数据挖掘问题。利用多Agent系统解决分布式数据挖掘中的问题，能够快速建立基于Agent的分布式数据挖掘应用。主要研究：一个开放平台AGIP的结构和内容主要功能模块的设计，及一个集成工具VAStudio。

其中，分布式数据挖掘问题是指将数据挖掘技术与分布式计算技术相结合，提高对复杂问题的求解能力。而该平台基于MAGE软件工程开发方法学，即Multi-AGent Environment，一个多Agent系统的开发环境，包括一系列支持AOSE设计和开发过程的工具集。

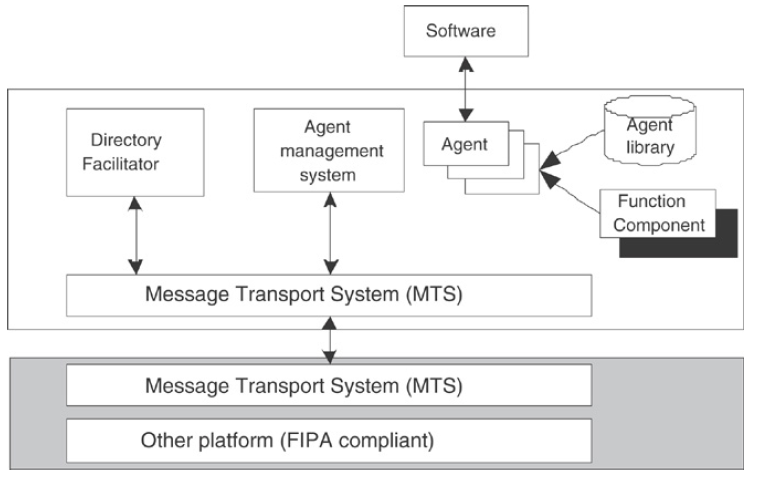


图10 MAGE系统结构

其中software是指所有非Agent的、外部的组件，将通过添加新的服务或必要的通信模块等封装为Agent；而Directory Facilitator是指Agent发布可供其他Agent使用的服务，并查找自己可以获取的服务；Agent management system是指管理Agent的定义，包括在系统中注册的Agent的地址等信息；而底层则以FIPA ACL作为标准通信语言。

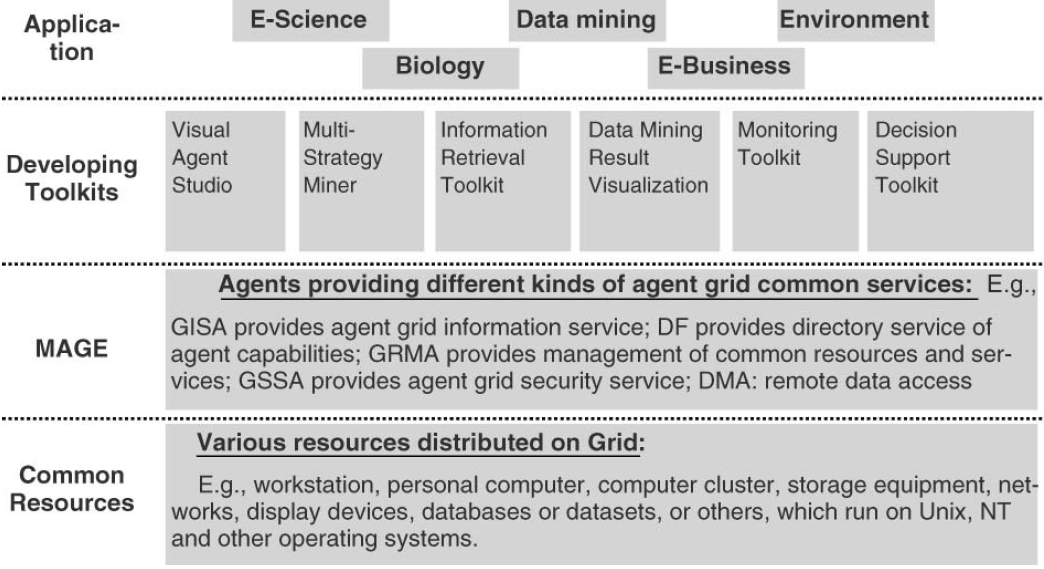


图11 AGIP系统结构

基于MAGE的AGIP体系结构如图是所示，是四层结构在物理资源层和MAGE层的基础上，研发了相关支持工具和平台以支持上层领域的数据挖掘需求。

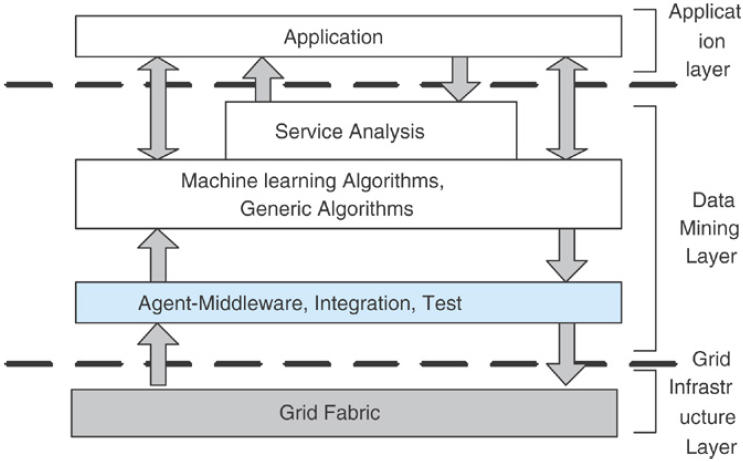


图12 任务处理结构

网格的存在，是为了将上述过程并行化，同时开展数据挖掘任务。因此需要提供一些核心服务，例如：服务查询，在MAGE传统的查询模块上加入了推理机，使得模块可以自主判断参与协作完成任务的agent种类；资源管理，建立一种网格资源管理Agent，对外提供任务的提交和撤销接口，对内管理网格基本计算和存储资源；数据管理，不同于一般意义的网格数据管理，是为了对计算过程中的中间结果进行缓存和一致性管理，避免传输浪费。

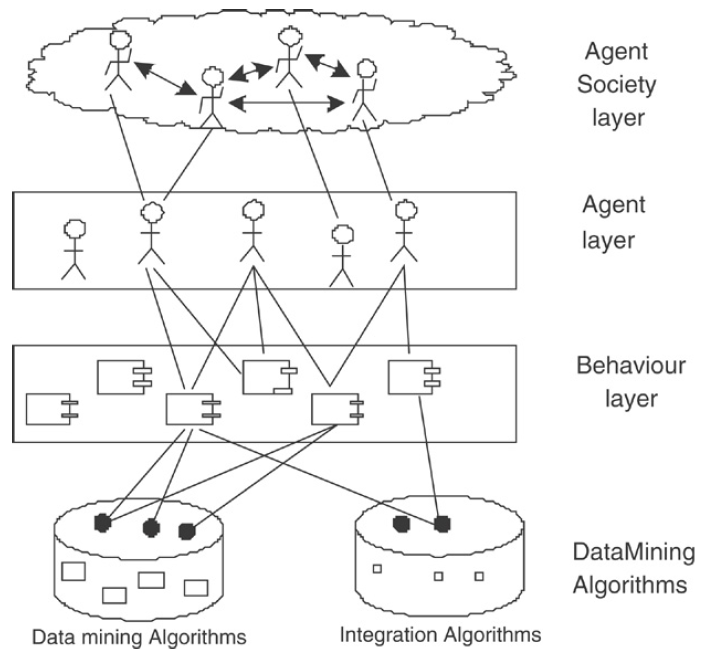


图13 数据挖掘过程

这样通过网格，进行的分布式数据挖掘过程如图所示，底层是相关的数据挖掘算法，以及将这些算法能够进行有效组合的连接算法。通过这些算法，将形成若干能力。一个Agent可以对应具有几个相应的能力。而若干Agent合作，就将有助于解决相关问题。

## 5.4 其他系统的体系结构

1. Agent-oriented Distributed System Architecture based on OGSA[26]

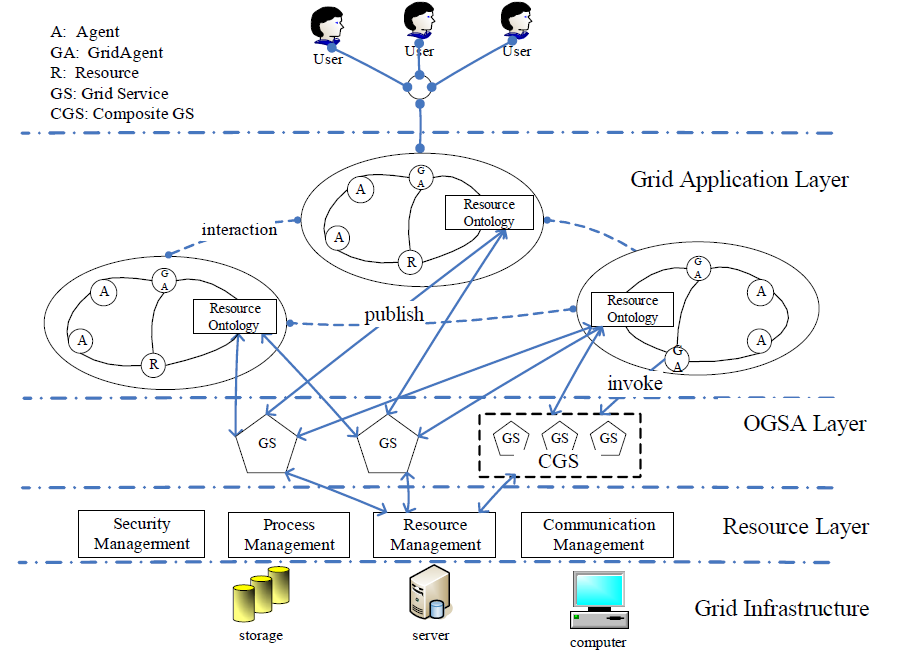


图14 体系结构

属于网状结构，将Agent作为网格中的基本单元，相关联合后提供网格服务。

1. Agent–enabled Grid node’s Architecture[27]

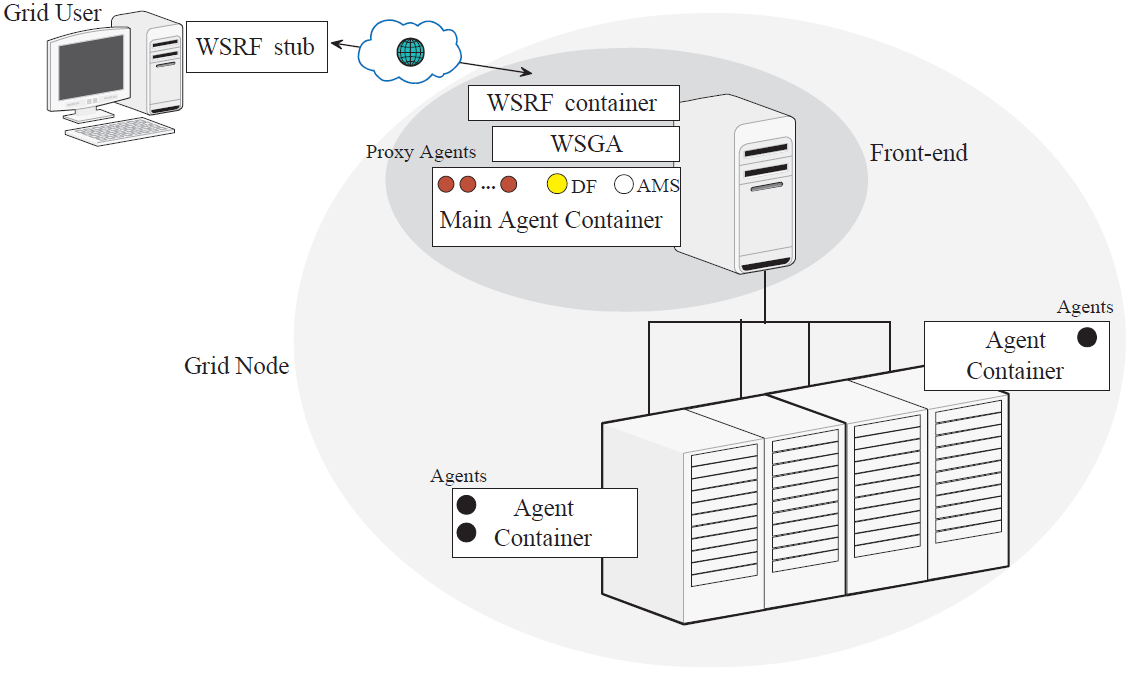


图15 体系结构

属于网状结构，内部分为多种Agent，网关AgentWSGA负责与外界传递消息。而每个机器由一个Agent控制，动态监控他们的能力和资源情况。

1. multi-agent approach to intrusion detection for Grid[28]

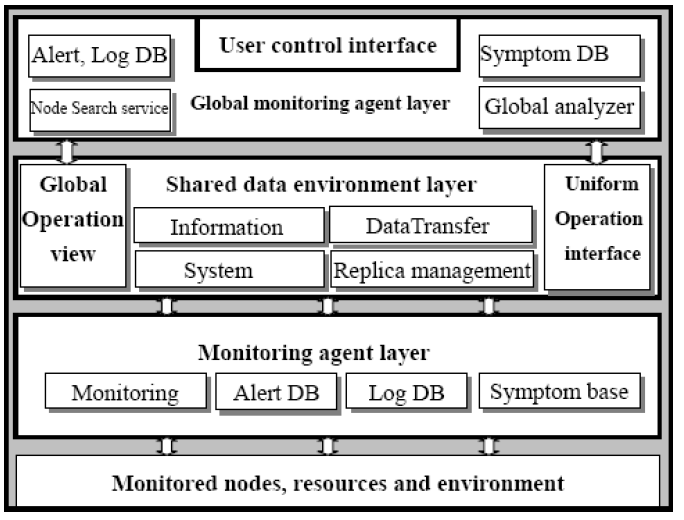


图16 体系结构

属于层次结构，底层是网格提供的相关服务，而上次是进行环境全局的Agent，他们在执行任务的工程中，将利用下次网格的相关资源。

1. Multi-agent parallel file system[29]

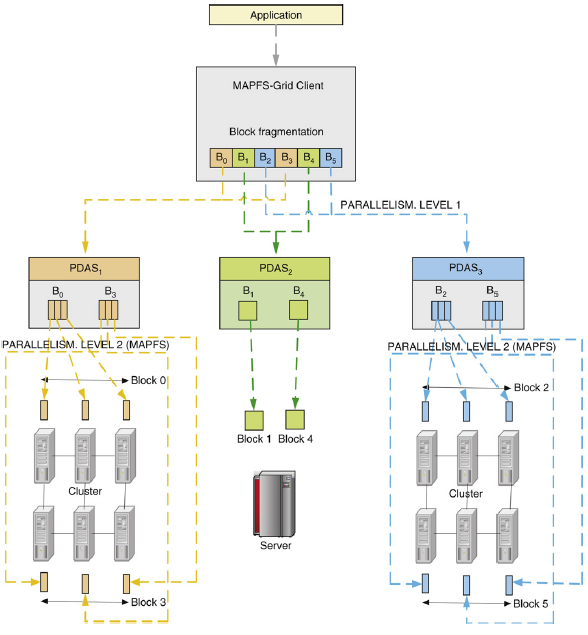


图17 体系结构

属于网状结构，每个Agent负责完成分配的任务并管理自己下属的相关资源。

1. agent-based storage management for Grid data resources[30]

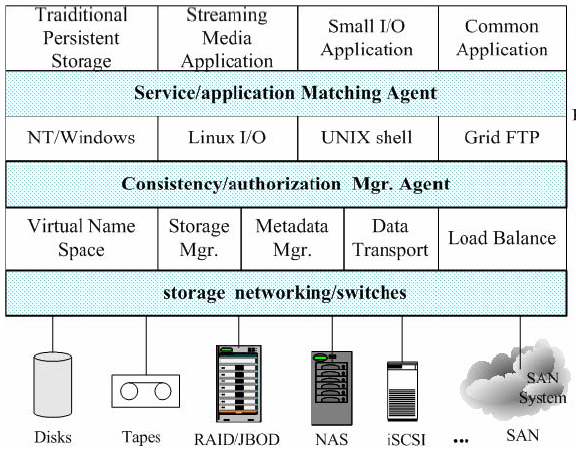


图18 体系结构

属于中间件结构，Agent分为两层，将上层用户的管理需求和服务与下层的网格资源进行联系。

1. Agent-Based Dynamic Grid Resource Management System[31]



图19 体系结构

属于层次结构，其中GRMA即Grid Resource Management Agent负责对分配下来的资源管理任务进行处理。

1. A knowledge grid architecture based on mobile agent[32]

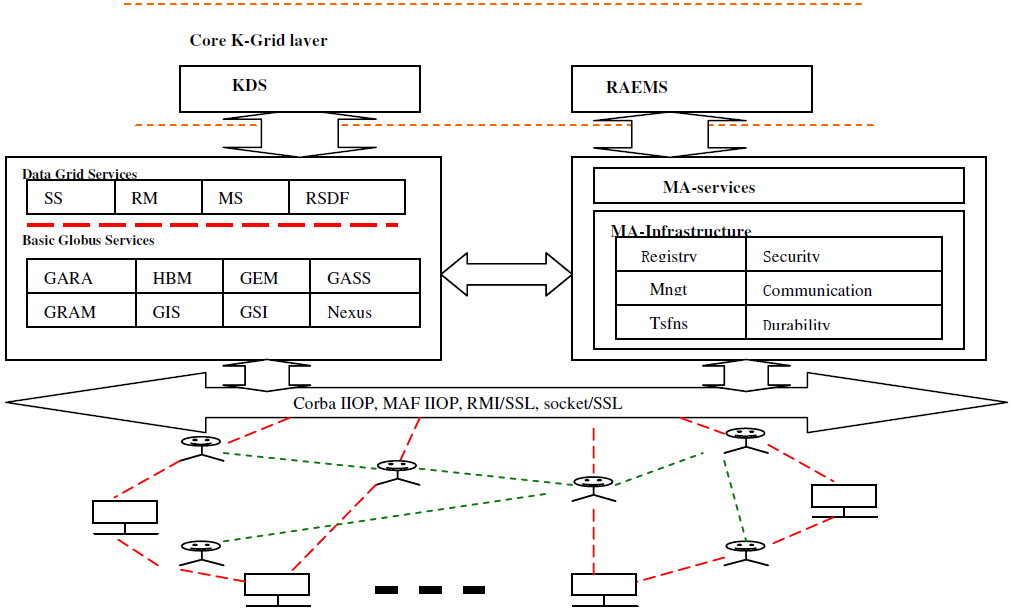


图20 体系结构

属于中间件结构，利用移动Agent在客户端和服务器数据挖掘服务之间传递数据。

1. Robust Agent-based Grid Scheduling Framework[33]



图21 体系结构

属于层次结构，上层的Task Partition Agent分离任务后，Task Control Agent将控制任务的执行，而下层Agent将利用网格资源完成任务。

1. Middleware architecture[34]

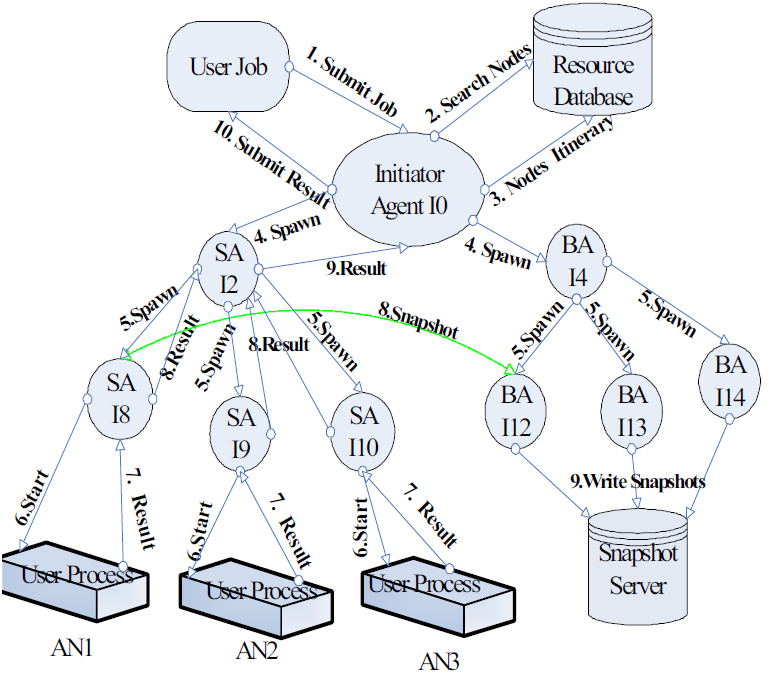


图22 体系结构

属于中间件结构，利用Agent对网格资源进行管理，隔离了用户与具体物理资源。

# 总结与展望

本次学术论坛集中介绍了网格技术与多Agent技术的结合，重点关注与基于Agent的网格系统的体系结构设计方法，主要可以分为以下三类。

图23 按照系统分类

中间结构中，Agent被作为沟通上下层的中间媒介；在层次结构中，MAS为系统中的独立一层管理网格，或者MAS内部各层Agent之间没有交互关系；而在网状结构中，Agent作为一种封装并提供网格能力的技术。

而在这些结构中，Agent技术的引入方式也存在如下区别：

图24 按应用方式分类

其中，应用类是指将基于Agent技术开发的软件应用到网格中，以提高软件的运行效率；而管理类是指将Agent技术作为管理网格资源、服务等的一种具体技术，以提高网格的能力；封装类是指将Agent技术作为封装网格节点或网格服务的一种具体技术，以提高网格的能力；管理与封装类是指分为两类Agent，一种作为网格服务的封装手段，一种作为网格基础服务的提供者。

因此可以看出，目前的研究热点是将Agent技术作为一种新的基础技术，用于解决网格本身的服务发现、资源定位与管理等问题。而并未过多考虑如何利用Agent技术将多Agent系统，甚至是非Agent软件进行改造，使之能够应用于网格环境，以提高这些软件的并行能力。因此我们在设计多Agent计算架构，应该借助现有的分布式计算技术，将Agent技术作为改造传统软件系统，使之具有分布式计算能力的重要手段。

# 参考文献

1. Foster, Ian, and Carl Kesselman, eds. The Grid 2: Blueprint for a new computing infrastructure. Elsevier, 2003.
2. Shah S N M, Haron N, Zakaria M N B, et al. Agent-based Robust Grid Scheduling Framework for High Performance Computing[J]. AASRI Procedia, 2012, 1: 554-560.
3. Berman, Fran, Geoffrey Fox, and Anthony JG Hey, eds. Grid computing: making the global infrastructure a reality. Vol. 2. John Wiley and sons, 2003.
4. Carvalho D, de Souza L R, Barbastefano R G, et al. Stochastic Product-Mix: A Grid Computing Industrial Application[J]. Journal of Grid Computing, 2015: 1-12.
5. Buyya R, Abramson D, Giddy J. Nimrod/G: An architecture for a resource management and scheduling system in a global computational grid[C]//High Performance Computing in the Asia-Pacific
6. Berman F, Wolski R, Casanova H, et al. Adaptive computing on the grid using AppLeS[J]. Parallel and Distributed Systems, IEEE Transactions on, 2003, 14(4): 369-382.
7. Foster I, Zhao Y, Raicu I, et al. Cloud computing and grid computing 360-degree compared[C]//Grid Computing Environments Workshop, 2008. GCE'08. Ieee, 2008: 1-10.
8. Foster, Ian, and Carl Kesselman, eds. The Grid 2: Blueprint for a new computing infrastructure. Elsevier, 2003.
9. Foster I, Kesselman C, Tuecke S. 17 CHAPTER The Open Grid Services Architecture[J]. The Grid: Blueprint for a New Computing Infrastructure, 2004.
10. Banks T. Web services resource framework (wsrf)–primer v1. 2[J]. OASIS committee draft, 2006.
11. Ciancarini, Paolo, and Michael Wooldridge, eds. Agent-Oriented Software Engineering: First International Workshop, AOSE 2000 Limerick, Ireland, June 10, 2000 Revised Papers. No. 1957. Springer Science & Business Media, 2001.
12. Balasangameshwara J, Raju N. A hybrid policy for fault tolerant load balancing in grid computing environments[J]. Journal of Network and Computer Applications, 2012, 35(1): 412-422.
13. Bazinet A L, Zwickl D J, Cummings M P. A gateway for phylogenetic analysis powered by grid computing featuring GARLI 2.0[J]. Systematic biology, 2014, 63(5): 812-818.
14. Rani S, Suri P K. Resource Management in Grid Computing: A Review[J]. Global Journal of Computer Science and Technology, 2014, 13(17).
15. Tsugawa M, Fortes J A B. A virtual network (ViNe) architecture for grid computing[C]//Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International. IEEE, 2006: 10 pp.
16. Mashayekhy L, Grosu D. A merge-and-split mechanism for dynamic virtual organization formation in grids[J]. Parallel and Distributed Systems, IEEE Transactions on, 2014, 25(3): 540-549.
17. Toporkov V, Toporkova A, Tselishchev A, et al. Core Heuristics for Preference-Based Scheduling in Virtual Organizations of Utility Grids[M]//Intelligent Distributed Computing VIII. Springer International Publishing, 2015: 321-330.
18. Souri A, Navimipour N J. Behavioral modeling and formal verification of a resource discovery approach in Grid computing[J]. Expert Systems with Applications, 2014, 41(8): 3831-3849.
19. Chang Y S, Shih P C. A resource-awareness information extraction architecture on mobile grid environment[J]. Journal of Network and Computer Applications, 2010, 33(6): 682-695.
20. Howard N. Intent-Based Ontology For Grid Computing Using Autonomous Mobile Agents: U.S. Patent 20,140,331,229[P]. 2014-11-6.
21. Saur D, Haque T R, Herzog R, et al. MAGiC: Multi-Agent Planning using Grid Computing concepts[C]//12th International Symposium on Artificial Intelligence, Robotics and Automation in Space. Montreal. 2014.
22. Leong P, Miao C, Lee B S. Agent oriented software engineering for grid computing[C]//Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on. IEEE, 2006, 2: 7 pp.-2.
23. Chen D, Theodoropoulos G K, Turner S J, et al. Large scale agent-based simulation on the grid[J]. Future Generation Computer Systems, 2008, 24(7): 660-671.
24. Li C, Li L Y. A multi-agent-based model for service-oriented interaction in a mobile grid computing environment[J]. Pervasive and Mobile Computing, 2011, 7(2): 270-284.
25. Shi Z, Huang H, Luo J, et al. Agent-based grid computing[J]. Applied Mathematical Modelling, 2006, 30(7): 629-640.
26. Liu W, Liu Z, Tong W Q. Agent-Oriented Modeling for Grid Service[C]//Machine Learning and Cybernetics, 2007 International Conference on. IEEE, 2007, 1: 77-82.
27. Micillo R A, Venticinque S, Aversa R, et al. A Grid Service for Resource-to-Agent Allocation[C]//Internet and Web Applications and Services, 2009. ICIW'09. Fourth International Conference on. IEEE, 2009: 443-448.
28. Zhu P Y, Gao J, Jiang B O, et al. A new flexible multi-agent approach to intrusion detection for Grid[C]//Machine Learning and Cybernetics, 2006 International Conference on. IEEE, 2006: 7-12.
29. Pérez M S, Sánchez A, Abawajy J H, et al. An agent architecture for managing data resources in a grid environment[J]. Future generation computer systems, 2009, 25(7): 747-755.
30. Liu Z. Autonomic Agent-based Storage Management for Grid Data Resource[C]//Semantics, Knowledge and Grid, 2006. SKG'06. Second International Conference on. IEEE, 2006: 54-54.
31. Li F, Qi D, Zhang L, et al. Research on Novel Dynamic Resource Management and Job Scheduling in Grid Computing\*[C]//Computer and Computational Sciences, 2006. IMSCCS'06. First International Multi-Symposiums on. IEEE, 2006, 1: 709-713.
32. Wang G, Wen T, Guo Q, et al. A knowledge grid architecture based on mobile agent[C]//Semantics, Knowledge and Grid, 2006. SKG'06. Second International Conference on. IEEE, 2006: 48-48.
33. Shah S N M, Haron N, Zakaria M N B, et al. Agent-based Robust Grid Scheduling Framework for High Performance Computing[J]. AASRI Procedia, 2012, 1: 554-560.
34. Jia C, Yue W, Bei H, et al. Multi-Agent-Based Middleware over Grid Computing[C]//Semantics, Knowledge and Grid, 2005. SKG'05. First International Conference on. IEEE, 2010: 86-86.