第34卷 第19期 计 算 机 工 程 2008年10月
Vol.34 No.19 Computer Engineering October 2008
·网络与通信· 文章编号：1000—3428(2008)19—0109—03 文献标识码：A 中图分类号：TP311

基于 Web Services 的网格数据传输子系统

陈海勇，武 林，王 娟
(解放军信息工程大学信息工程学院，郑州 450002)

摘 要：数据是网格环境中一类特殊的资源，它的流动、管理以及统一视图是网格系统的重要组成部分。该文分析数据的特点并设计实现一个基于FTP协议，支持多线程、断点续传、异常识别、连接维持以及第三方传输和“中转不落地传输”两种传输模式的数据传输子系统。该子系统提供了数据的动、静态部署能力，为在超级计算机和集群上运行作业提供数据支持，形成在各个网格资源上所部署的数据的统一视图，方便对数据的进一步操作。

关键词：网格；数据传输子系统；多线程；数据统一视图

Grid Datatransfer Subsystem Based on Web Services

CHEN Hai-yong, WU Lin, WANG Juan
(Institute of Information Engineering, PLA Information Engineering University, Zhengzhou 450002)

【Abstract】 Data is a kind of resources in grid environment, it has many differences with other resources. Its flow, management and uniform view play an important role in grid system. This paper analyzes the characteristic of data, designs and implements a grid datatransfer subsystem based on FTP protocol, which abets multithreading download, break point transfer, exception identify and connection maintain. Also the subsystem implements two transfer models which are “not landed” transfer model and third control transfer model. It not only offers grid the ability of deploying data, gives data help of running task with super computers and clusters, but also forms uniform data view of all data, making it convenient to take deeper work.

【Key words】 grid; datatransfer subsystem; multithread; uniform data view

1 概述

随着网格技术的发展，数据在网格中的重要性日益显现。数据传输子系统负责网格环境内的数据流动，它不但是网格中间件的基础设施，也是其重要的核心组成部分之一。网格系统中其他模块对数据有着很大的依赖性，如作业模块、应用模块等，它们的运行都需要数据的部署。数据传输子系统不但通过对网格环境下各种异构的资源、异构的文件系统进行集合，形成透明、统一的数据视图与数据使用环境，而且基于开源的Edtftpj软件包实现了网格系统内灵活、高效率的数据流动，向网格提供了数据的动、静态部署能力。

目前已经有了如Globus项目、Legion项目、Globe项目、NetSolve项目、Javalin项目等网格项目，但是由于它们的结构与实现各异，不同的网格项目需要符合其结构与功能要求的不同的子功能模块。本文根据国家某重点面向服务的网格系统项目，基于Web Services技术设计实现了一套较为实用的数据传输子系统。

2 系统总体设计

本系统参考OGSA架构，采用以服务为中心的思想，运用Web Services技术开发，使用WSDL[1]描述服务，UDDI[2]发布、查找服务，SOAP[3]用来执行服务调用。系统的结构参照五层沙漏结构，根据功能和部署位置划分为3层：门户层，系统层，核心层。门户层部署在网格门户层服务器，系统层部署在网格系统层服务器，核心层部署在网格节点上。这种层次的划分使得任何对网格节点的访问都必须经过系统层中转，不但有效地划分了功能层次，增强了易管理性，而且系统的层次明显、结构清晰、易于扩展。

2.1 基本定义

定义 1 存储资源：指网格辖域内通过VO(虚拟组织)发布后的存储资源，包括门户空间、VSFTP服务器和GridFTP服务器。它是包括3个属性的对象：存储资源的使用配额，存储空间的相对路径和允许使用的用户(群)。

定义 2 门户空间：指本系统所在的系统层服务器提供给网格用户的存储空间。它一方面将服务器冗余的存储资源提供给用户，另一方面为用户数据接入网格系统提供一个中转站。

定义 3 TJDL：TJDL指Transfer & Job Describe Language，用以描述本次数据传输作业的详细内容，该文本描述语言遵循XML规范。TJDL的设计目的主要在于使系统具有可扩展性。

定义 4 中转不落地传输：中转不落地传输指位于C点的数据传输服务能够将位于A点的数据传输到B点。与第三方传输的不同之处在于，传输过程中通过C点中转时，数据仅在内存中暂存而不经过硬盘。这种设计解决了某些巨型机防火墙要求数据只能通过可信服务器中转才能进入的特殊安全需求。

2.2 总体设计

数据传输子系统是网格系统中的核心部件之一，它接受用户请求，完成将数据文件从一个位置传输到另一个位置的

作者简介：陈海勇(1983－)，男，讲师、博士研究生，主研方向：网格计算；武 林、王 娟，硕士研究生

收稿日期：2008-04-20 **E-mail：**Cristal_83@163.com

传输任务。同时也接受批作业服务等单元的调用实现数据的动、静态部署，其总体设计如图 1 所示。

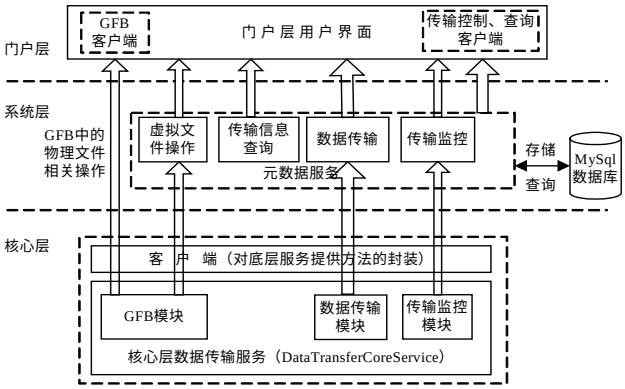

图 1　数据传输子系统总体设计

对各模块操作的详细介绍如下：

(1)GFB 即 Grid File Browser 网格文件浏览器，由 GFB 模块和 GFB 客户端组成，该模块提供对虚拟文件和物理文件的操作。GFB 客户端根据其所辖文件的特性分为 2 类：

1)虚拟文件客户端：负责虚拟文件操作，这些操作包括：用户目录下的文件列表，目录的创建，文件夹或文件的删除，重命名，依据关键字的文件检索、上传、下载、文件权限设置、文件刷新等。

2)物理文件客户端：由 GridFTP 客户端、VSFTP 客户端以及门户空间客户端组成，负责物理文件操作，这些操作包括：用户目录下的文件列表，目录的创建，文件夹或文件的删除、重命名、上传和下载等。通过 GFB 可以形成用户在各个计算结点上的统一数据视图，并可以进行相关操作。

(2)数据传输模块提供的操作主要有：数据传输(transfer)。

(3)传输监控模块提供的操作主要有：断点续传(resume)，重启传输(restart)，暂停传输(stop)，获取传输状态(getStatus)，取消传输(cancel)，传输删除(delete)，通过传输和监控操作可以完成数据从普通存储设备到计算结点之间的传输和控制工作。

(4)元数据服务中的传输信息查询模块提供的操作主要有：查询用户提交的所有作业号，查询某个作业号下所有的传输参数信息，查询某个作业号下某个传输的参数信息，这些操作主要是为了方便用户查询已提交的传输作业而提供的。

(5)核心层服务中的数据传输和传输监控模块主要是对核心层服务中对应方法的封装(权限检查和数据库操作)，元数据服务为访问本地数据传输服务中的统一视图操作进行权限检查，不进行其他封装。

3　系统实现

门户层是通过基于 RIA 技术使用 Macromedia 的 FLEX 开发，向用户提供友好的图形界面。系统层，即元数据服务。主要工作比较简单，包括对传输进行记录以供用户查询传输信息、调用核心层数据传输服务提交作业等。核心层是数据传输子系统的核心，也是传输的实际实现者。下面是对核心层几个关键问题的阐述。

3.1　多线程传输

用户通过登录门户层网站，使用图形界面中数据传输部分，通过拖拉操作生成作业描述文档 TJDL，然后将其交与系统层进行处理。数据传输时序图如图 2 所示。

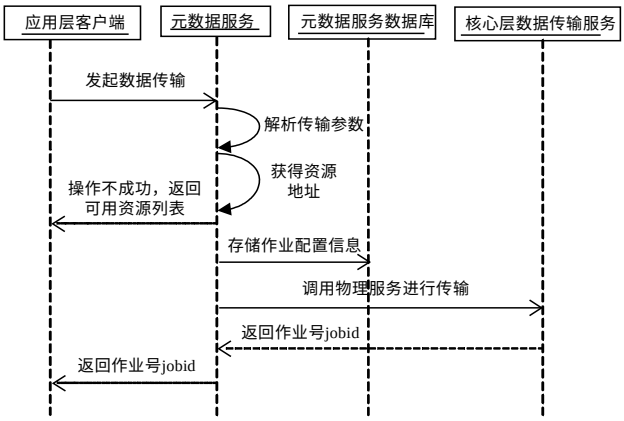

图 2　数据传输时序图

核心层服务的代码结构如图 3 所示，这种结构较好地保证了代码的层次性。

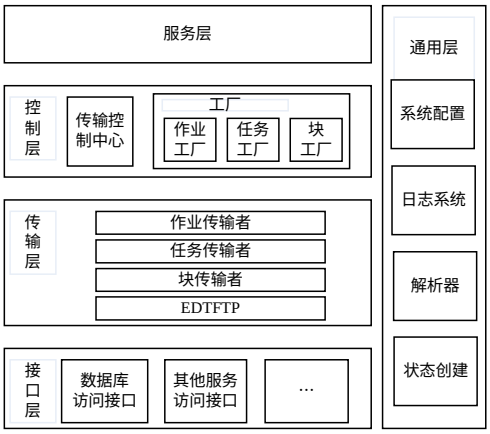

图 3　核心层服务代码结构示意图

作业工厂[4]采用Java设计模式中的工厂模式，由 1 个注册中心、2 个作业对象记录器、1 个edtftpj客户端、若干作业传输者组成。注册中心，即作业对象列表Hashtable RunningJobs存放作业对象和对应的作业号。2 个作业对象记录器，即***Vector***向量，1 个记录历史的作业号，1 个记录当前运行态的作业号，以便作业状态查询和检查点恢复使用。为了便于传输，设计了 3 类数据结构，即作业对象、任务对象、块对象。它们的划分是对应于一次传输任务、单个文件、单个线程，这样使得传输过程清晰和代码功能明确。作业对象描述的是一次传输请求，也就是对本次请求中所包含的传输任务的总体描述如并发数(同时进行传输的任务数)以及一个任务对象列表。一个任务对象代表了传输请求描述中的一个传输任务，也就是一个文件的传输描述，它包含了与此传输任务相关的所有需求信息，如数据源、传输目的、传输协议、文件类型、线程数等，此外还包含一个块对象列表。块对象描述了一个块的所有特征，它的创建数量为传输的线程数，包括块起始位置、块大小等。与作业有关的数据结构建立后，数据传输过程将被开启。与作业对象数据结构相对应系统分别实现了 3 类“传输者”，即作业传输者、任务传输者、块传输者。

传输的具体流程如下：

(1)核心层数据传输服务接口接收到传输请求后，先检查传输控制中心是否已存在，不存在则由服务层创建并根据检查点的备份初始化，再将传输描述文档 TJDL 交由控制中心处理。传输控制中心维护一个作业工厂用来创建作业对象。

(2)作业工厂调用通用层解析器解析 TJDL，创建作业对象，并向注册中心注册，然后将该作业对象提交到作业传输者进行传输。

(3)作业传输者首先对作业对象进行初始化，填充基本参数并通过 Mysql 实现作业对象的持久化保存。然后使用 edtftpj 客户端登录远程 ftp 服务器获取文件信息用以初始化任务传输者，根据作业对象中的并行数，即本次传输的文件数，通过任务工厂创建相应多个任务传输者初始化其参数。作业对象通过自身维护的任务传输者列表记录任务传输者总数和已完成的数量。进入休眠。

(4)任务传输者根据传输的线程数，通过块工厂创建相应多个块传输者。任务对象通过自身维护的块传输者列表记录块传输者总数和已完成传输的块数量；进入休眠。

(5)块传输者被创建后，调用 edtftpj 客户端发起传输，在该块完成传输后唤醒任务传输者。

(6)任务传输者检查块传输者是否都已完成传输。若没有继续休眠；若完成进入(7)。

(7)任务传输者合并文件，唤醒作业传输者。

(8)作业传输者检查任务传输者是否都已完成传输。若没有继续休眠；若完成进入(9)。

(9)作业传输者向作业工厂汇报传输完成。作业工厂修改对应的作业传输者状态。

(10)作业工厂结束本次作业传输。

3.2 多线程传输的状态控制

多线程传输控制根据其实现所在的层的不同分为 2 类。

第 1 类：系统层实现。用户对于传输的删除(delete)，即删除该层上数据库保存的作业信息。

第 2 类：核心层实现。断点续传(resume)、重启传输(restart)、暂停传输(stop)、获取传输状态(getStatus)、取消传输(cancel)均在核心层实现。其控制实现时序如下：

(1)服务层接收到控制请求，首先检查传输控制中心是否存在，若不存在则创建并根据检查点的备份初始化，然后将本次传输控制请求交于传输控制中心进行处理。

(2)控制中心对本次控制请求做 2 个检查。查询作业工厂的注册中心和 2 个作业对象记录器，检查是否都存在指定的作业号，存在进入(3)，否则进入(11)。

(3)对用户进行合法性检查，看本次请求是否符合安全规范；合法进入(4)，否则进入(11)。

(4)控制中心调用作业传输者相应接口实施对传输任务的控制。

(5)作业传输者首先修改注册中心和 2 个记录器里关于该作业对象的信息，然后通过自身维护的任务对象列表分别调用任务传输者控制接口；进入休眠。

(6)任务传输者通过自身维护的块对象列表，调用所属的块对象暂停接口；进入休眠。

(7)块传输者通过调用 edtftpj 的传输控制接口，实施对传输的真正控制，在完成后唤醒任务传输者。

(8)任务传输者检查是否全部块对象均完成操作，若完成，唤醒作业传输者。

(9)作业传输者向传输控制中心汇报传输完成。

(10)传输控制中心检查是否本次传输请求的所有作业传输者完成指定操作；若完成，进入(11)。

(11)根据结果，调用通用层状态创建工具，生成符合 XML 文档规范的字符串反馈给上层服务。

3.3 异常识别与连接维持

由于网络中断等网络异常导致传输作业终止的情况时有发生。有时网络的中断是暂时的，针对这种情况，系统设计实现了传输过程中的异常识别与连接维持功能。

该功能通过块传输者完成，当块传输者进行文件块传输过程中时，对捕获到的所有异常进行识别，如发现是网络连接超时异常或者 Socket 异常，即在传输配置文件中设定的异常容忍时间的范围内暂时维持传输状态并不断重试传输直到连接恢复。

经测试在传输动作进行中拔掉网线，传输状态进入连接维持，恢复后传输继续。该功能不但提高了传输过程的稳定性，而且使得系统的智能化大大提高。

4 系统测试

测试环境如下：网络带宽 10 Mb/s；客户端运行环境 CPU 400 MHz，内存 64 MB；待下载文件大小约 2.4 MB；地址 ftp://ftp.gui.uva.es/pub/ISOS/fedoral/3/i386/iso/FC3-i386-rescued.iso。

测试结果如图 4 所示。

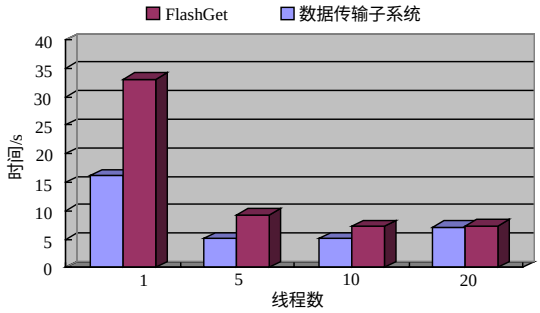

图 4 测试结果

由图 4 可以得出以下结论：

(1)本系统传输速率要高于 FlashGet。

(2)由于测试时网络状况和测试主机不理想，从测试数据来看，子系统的传输效能一般。

(3)当线程数增加时，子系统的传输效率先升后降。这是线程数的增加给系统带来的开销增大所致。

因为网络环境和测试主机的性能对 flashget 和子系统都是均等的，所以数据是可信的，测试结论是正确的。

5 结束语

设计实现的数据传输子系统，遵循通用的 OGSA 架构和相关 Web Services 技术规范，可以实现网格环境中的数据动、静态部署功能和分散在广域范围内的异构资源上数据文件的统一视图，并通过测试验证了系统符合设计要求。随着网格技术的发展，如何将新技术运用到子系统内以提高其性能，是笔者目前研究的内容。

参考文献

[1] Christensen E, Curbera F, Meredith G, et al. Web Services Description Language(WSDL)1.1[EB/OL]. (2001-04-15). http://www.w3.org/TR/wsdl.

[2] Jewell T, Chappell D. Java Web Services, Universal Description, Discovery and Integration[EB/OL]. (2002-03-20). http://www.uddi.org/.

[3] W3C. SOAP 1.1[EB/OL]. (2000-05-10). http://www.w3.org/TR/SOAP.

[4] Cooperj J W. Java 设计模式[M]. 王 宇, 林 琪, 杜志秀, 译. 北京: 中国电力出版社, 2002: 15-31.