Desenvolvimento de um componente para Transferência de Arquivos em Ambientes de Alto Desempenho Heterogêneos

Izaias de Faria e M.A R. Dantas

Universidade Federal de Santa Catarina (UFSC)

Departamento de Informática e Estatística (INE)

{izaiasdf, mario}@inf.ufsc.br

Abstract. *Different types of grids have been used in environments where it is required a large processing power and data storage. In this context, it is observed different grid software managers that use different communication protocols. Regardless of the software manager in use file exchange between grids is a challenge to be tackle. In this paper, it is presented a research work in developing a component that can communicate with different software managers, making it possible to transfer files regardless of the protocol used. Experimental results indicate that this component can successfully support its goals.*

Resumo. *Diferentes tipos de grades vêm sendo utilizadas em ambientes onde é necessário grande capacidade de processamento e armazenamento de dados. Neste contexto, foram observados diferentes gerenciadores de grades que utilizam diferentes protocolos de comunicação. Independente do gerenciador em uso a troca de arquivos entre grades é um desafio a ser enfrentado. Neste artigo, é apresentado um trabalho de pesquisa onde foi desenvolvido um componente capaz de se comunicar com diferentes gerenciadores de grade, tornando possível transferir arquivos independentemente do protocolo utilizado. Resultados experimentais indicam que este componente atende com sucesso seus objetivos.*

1. Introdução

Sistemas de grade científica são ambientes importantes para pesquisadores analisarem, armazenarem e compartilharem dados. A maioria destes sistemas fornece não apenas os serviços computacionais, mas também serviços de dados. Entre os sistemas existentes de grade científica, muitos dos serviços computacionais são construídos em Globus Toolkit [Foster 2003], ou suas variantes, tais como gLite [Cern, 2011]. Alguns outros grupos adotam Condor (2011), um sistema de agendamento diferente, que pode funcionar com Globus via Condor-G. No entanto, o mecanismo usado para armazenamento de dados e acesso não é único.

Alguns sistemas de dados da rede são projetados para serem sistemas de arquivos virtuais, oferecendo espaço de armazenamento a pesquisadores sem que usuários saibam o exato lugar de dados, como SRB [Baru 1998] e seu sucessor, iRODS [Rajasekar 2006], bem como dCache [Ernst 2001]. A maneira de acessar esses sistemas de grades de dados varia. Por exemplo, dCache fornece uma interface GridFTP e uma

interface HTTP, de modo que sistemas de submissão a grade possam facilmente acessar dados a partir dele. Alguns outros, como iRODS, implementam os seus próprios protocolos proprietários e sistemas de envio de trabalhos a grade não têm como obter dados diretamente fora dele.

Este artigo apresenta o trabalho de pesquisa relativo ao desenvolvimento de um componente para transferência de arquivos em arquiteturas de grades computacionais heterogêneas. O componente tem como diferencial ser capaz de adaptar-se a diferentes tipos de fontes de dados, para tal utiliza o protocolo padrão para comunicação com grades, o GridFTP. Visando obter um sentimento da eficácia do componente desenvolvido, o mesmo foi testado e seu desempenho avaliado comprovando-se sua viabilidade com relação a alternativas utilizadas atualmente no mercado.

O artigo está organizado da seguinte maneira. Na segunda seção apresentam-se alguns trabalhos relacionados ao desafio de transferência de grandes arquivos. A abordagem desenvolvida é descrita na terceira seção. O ambiente e resultados experimentais são apresentados na quarta seção. Na quinta seção finaliza-se este artigo com as conclusões e trabalhos futuros relacionados ao tema desta pesquisa.

2. Trabalhos Relacionados

2.1. TCP

Transmission Control Protocol (TCP) tem sido amplamente utilizado como um protocolo de comunicação em nível de transporte na Internet [Postel 1981]. GridFTP foi concebido para utilizar TCP como seu protocolo de comunicação a nível de transporte [Allcock 2003]. No entanto, o TCP é um protocolo de comunicação muito antigo que foi projetado na década de 1970. Vários problemas foram relatados sobre TCP, tais como a sua incapacidade para suportar a velocidade rapidamente crescente de redes recentes. Como exemplo, o atual TCP Reno (TCP versão Reno) não consegue detectar o congestionamento em uma rede até que ocorra perda de pacotes, portanto, um grande número de pacotes é perdido [Postel 1981]. Com as velocidades mais rápidas de redes e tamanhos maiores de buffer de roteadores em uma rede, a quantidade de pacotes perdidos e o throughput do TCP deterioram significativamente. Para resolver os problemas existentes no TCP, GridFTP tem características como o estabelecimento de múltiplas conexões TCP em paralelo para acelerar o início lento na fase inicial do TCP e negociando o tamanho do TCP socket buffer entre o servidor e o cliente GridFTP de acordo com o atraso da largura de banda de um rede [Allcock 2003].

No entanto, a eficácia desses recursos não foi totalmente investigada. Em outras palavras, as configurações ideais para o número de conexões TCP paralelos e tamanhos do TCP socket buffer não foram investigados. Existem alguns trabalhos relacionados que tratam o assunto de tamanho de TCP socket buffer e conexões TCP paralelas, como por exemplo, em [Semke 1998] e [Lu 2005].

2.2. Computação Virtual

Os sistemas de transferência de arquivos mais utilizados atualmente são aplicações que se apoiam em protocolos de nível de transporte, ou seja, TCP e UDP, para transferir dados de sua origem para o destino. Embora estes sistemas explorem técnicas altamente

sofisticadas para aumentar sua taxa de transferência, eles sofrem de uma desvantagem comum, nomeadamente a sua dependência sobre os protocolos de roteamento IP, que tem como consequência a fato de que o throughput que eles conseguem é limitado pela largura de banda disponível no caminho de rede escolhido pela camada de roteamento IP.

Infelizmente, os protocolos de roteamento IP notoriamente produzem rotas sub-ótimas [Savage 1999], uma vez que sua escolha de caminhos da rede não é guiada por considerações de desempenho como estão principalmente preocupados com a troca de conectividade da informação. Como consequência, não é raro o caso onde os tempos mais curtos de transferência podem ser obtidos escolhendo um caminho de rede diferente do escolhido pelos algoritmos de roteamento IP. Por exemplo, Savage (1999) observa que para 30 a 80 por cento dos caminhos escolhidos pelos algoritmos de roteamento IP entre pares de hosts da Internet, tomadas a partir de um conjunto relativamente grande de máquinas, foi possível encontrar caminhos alternativos com melhores características de desempenho.

Anglano (2004) descreve o File Mover, um software que atende os problemas acima através da exploração de uma arquitetura overlay network (Rede de sobreposição). Uma rede de sobreposição é uma rede virtual, em camadas em cima da Internet existente, cujos membros nós são colocados nas bordas da rede física subjacente e se comunica por meio de um protocolo de nível de transporte (por exemplo, TCP ou UDP). Cada par de nós membros de uma rede sobreposta se comunica por meio de um link virtual, que corresponde ao caminho de rede escolhido pela camada IP para transferência de dados de um membro para o outro. Os Relays de uma rede de sobreposição concordam em seguir em frente o tráfego ao longo de um ou mais links virtuais, até que o host de destino seja atingido.

2.3. Protocolo GridFTP

GridFTP é um protocolo de transferência de dados, que é projetado para transferir efetivamente grande volume de dados em Grid Computing [Globus 2002, Allcock 2003]. GridFTP é uma extensão de FTP (File Transfer Protocol) [Postel 1985] que tem sido amplamente utilizado e esta atualmente sob a normalização do GGF (Global Grid Forum) (GGF, 2011). GridFTP, que usa TCP como seu protocolo de comunicação na camada de transporte, foi concebido para resolver vários problemas do TCP. Por exemplo, além das características dos atuais FTP, ele tem funcionalidades adicionais, tais como negociação automática do tamanho do TCP socket buffer, transferência de dados paralela, controle de transferência de arquivos de terceiros, transferência de arquivos parciais, segurança, e transferência de dados confiável [Allcock 2003]. A maioria destas características específicas de GridFTP são realizadas por um novo modo de transferência chamado Modo de Bloqueio Estendida [Allcock 2003].

Atualmente, o servidor GridFTP e software cliente em conformidade com GridFTP versão 1 (GridFTPv1) é implementado no Globus Toolkit [Globus 2011], que é o middleware padrão de fato para a computação em Grid. No entanto, nesta implementação GridFTP específica, o recurso da negociação automática do tamanho do TCP socket buffer não é implementado e, portanto, o usuário deve especificar manualmente o número de conexões TCP paralelas para transferência de dados paralelo.

Além disso, GGF tem discutido os problemas com GridFTP v1 e também realizado um estudo sobre GridFTP v2 (versão 2) como uma solução para esses problemas [Mandrichenko 2005].

3. Abordagem Desenvolvida

Conforme o aumento de dados presentes em grades computacionais, têm se tornado bastante comum a prática de transferência de dados entre fontes de dados heterogêneas. Muito trabalho vem sendo realizado para integrar tais fontes de dados. Tipicamente, usuários precisam utilizar um cliente para copiar dados de uma fonte origem para uma fonte intermediária, como, por exemplo, uma fonte de dados local, e em seguida utilizar outro cliente para mover os dados a partir da fonte intermediária para a fonte destino. Isso significa um passo extra na transferência de dados, o que é errado e ineficiente, considerando que existe a necessidade de interação humana para efetuar a transferência.

A proposta de Zhang (2010) descreve uma abordagem diferente, que é capaz de converter qualquer interface de fonte de dados para a interface do GridFTP, que por sua vez é compatível com sistemas grade, e é capaz de potencializar as vantagens do protocolo com relação a transferência de dados. Ele resolve dois problemas com uma única solução. Primeiro, ele possibilita sistemas grade acessarem dados de qualquer fonte de dados. Segundo, transferir arquivos entre duas fontes de dados com protocolos distintos se torna possível sem a utilização de um ponto intermediário. A Figura 1 representa o funcionamento da mesma.

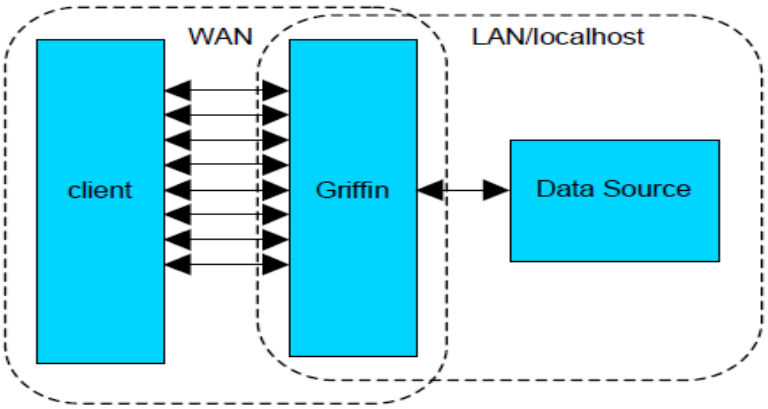

Figura 1. Modelo lógico do Griffin

A proposta deste trabalho consiste na modificação do código fonte do Griffin [Zhang 2010] de forma que a adição de adaptadores seja o mais simples possível. Conforme pode ser observado na Figura 2, a proposta modifica a arquitetura do Griffin adicionando uma nova camada de abstração ao seu código denominada Service Adaptor, cuja função é tornar o Griffin o mais flexível possível, ainda podemos destacar que tal camada implementa a interface Generic file system framework (Framework de sistemas de arquivos genérico) e os adaptadores de fontes de dados.

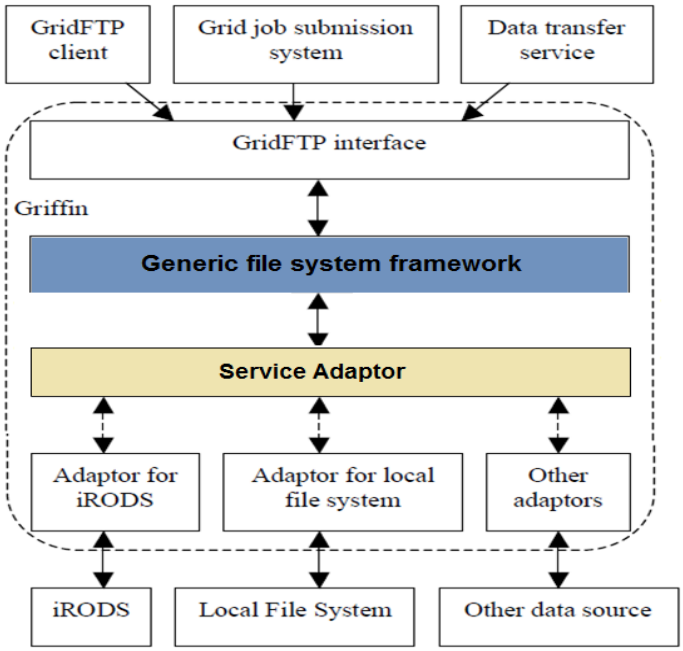

Figura 2. Arquitetura do Griffin

A implementação, ilustrada na Figura 2, foi desenvolvida em Java e baseado no framework Spring [Arthur 2005], que oferece uma arquitetura para desenvolvedores facilmente desenvolver aplicações modulares. A interface do GridFTP é a base da aplicação. Ele gerencia todas as conexões do cliente com um parser para analisar comandos GridFTP e invocá-los com um classe de comando relevantes. A implementação da autenticação GSI é baseada em jGlobus [Laszewski 2001]. Adaptadores para fontes de dados dependem da biblioteca de cliente relevante. Por exemplo, o adaptador para iRODS depende da biblioteca Java do iRODS, a Jargon. Tendo o Framework genérico torna-se possível desenvolver adaptadores para bibliotecas de outros sistemas de arquivos, como commons-vfs e JSAGA, de modo que o Griffin pode ser usado para acessar todas as fontes de dados que são suportados por essas bibliotecas. O binário de Griffin é leve, stand-alone e autônomo, com todas as bibliotecas java dependentes. Por isso, não depende de quaisquer componentes Globus, o que torna fácil de instalar e manter. Além disso, sendo uma aplicação Java, é possível ser executado na maioria dos sistemas operacionais sem recompilação.

Com o design modular e com a ajuda do framework de injeção de dependências Spring [Fowler 2004], mudar o sistema de dados subjacente requer apenas uma mudança em um arquivo XML de configuração, sem a necessidade de recompilar todo o do sistema. A Figura 3 é uma amostra de uma parte do arquivo de configuração do Griffin com o adaptador iRODS. Para executar Griffin com um sistema de arquivos local, só é preciso substituir a parte <bean id = "filesystem"> parte com a Figura 4, e reiniciar o Griffin.

```
<bean id="server"
class="au.org.arcs.griffin.server.impl.GsiFtpServer"
singleton="true">
   <property name="name" value="GSI FTP Server" />
   <property name="options" ref="options" />
   <property name="resources" value="griffin-resources"/>
   <property name="fileSystem" ref="fileSystem" />
</bean>
<bean id="fileSystem"
class="au.org.arcs.griffin.filesystem.impl.jargon.JargonFileSystemI
mpl" singleton="true">
   <property name="serverName" value="localhost" />
   <property name="serverPort" value="1247" />
   <property name="serverType" value="irods" />
</bean>
```

Figura 3. Exemplo de configuração com adaptador para iRODS

```
<bean id="fileSystem"
class="au.org.arcs.griffin.filesystem.impl.localfs.LocalFileSystemI
mpl" singleton="true">
   <property name="rootPath" value="/data/data-fabric" />
   <property name="userManager" ref="userManager" />
</bean>
```

Figura 4. Exemplo de configuração com adaptador para sistema de arquivos local

Para acessar sistemas de arquivos arbitrários de uma maneira uniforme, foi criada a interface Generic file system framework e em seguida a camada que implementa seus métodos a Service Adaptor que funciona como back-end para a interface GridFTP. Por meio da camada Service Adaptor, qualquer pedido proveniente da interface GridFTP será traduzido para uma operação padrão do sistema de arquivos. Para torná-lo genérico, a camada é leve e simples, suportando apenas operações básicas e comumente implementadas pela interface GridFTP. A concepção desta camada é específica para as necessidades do protocolo GridFTP, especialmente recursos avançados para transferência de dados, a fim de minimizar a sobrecarga entre a interface front-end do GridFTP e o sistema de dados no back-end. Como o protocolo GridFTP é usado principalmente para transferência de dados, esta camada não requer outras funções, tais como manipulação de meta-dados.

A estrutura top-down simples é ilustrada na Figura 5, com quatro grandes objetos.

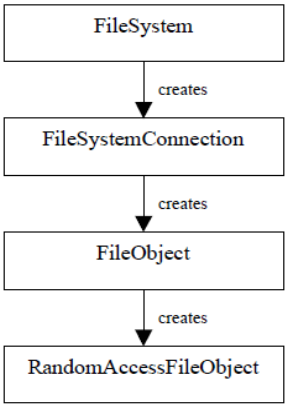

Figura 5. A estrutura da camada Service Adaptor

O objeto FileSystem é a raiz da hierarquia. Este objeto apresenta o sistema subjacente de dados na interface GridFTP e é chamado na inicialização e desligamento. Em particular, o método init () é chamado para examinar as configurações e verificar a conectividade com a fonte de dados durante a inicialização. Se houver um erro, por exemplo, a configuração não está correta ou o sistema subjacente de dados não está acessível, uma exceção será lançada, assim o servidor GridFTP pode decidir se ele pode continuar a ser executado. Da mesma forma, o método exit() é chamado durante o desligamento para liberar recursos e encerrar com o sistema de dados se houver.

Correspondentemente, uma conexão com o sistema de dados subjacente será criada, de acordo com a credencial GSI. Este método retorna um objeto FileSystemConnection, que representa uma sessão para o sistema de dados. Se o sistema de dados subjacentes suporta autenticação GSI, a mesma credencial GSI utilizada para autenticar com a interface GridFTP será usada para autenticar com o sistema de dados subjacente. No entanto, se o sistema de dados subjacente não suportar à autenticação GSI, algum tipo de mecanismo de mapeamento será empregado.

O objeto FileSystemConnection mantém uma conexão com o sistema de dados subjacente para o usuário. Ela consiste de alguns métodos para todo o sistema e relacionadas ao usuário.

O objeto FileObject representa um objeto de dados na fonte de dados. Ele é projetado para refletir a classe java.io.File, fornecendo métodos para obter o nome do arquivo, caminho, caminho canônico, comprimento, hora da última modificação, o seu pai ou seus filhos. Além disso, ele permite aos usuários verificar se esse arquivo existe, ou se este objeto é um arquivo ou um diretório. Os usuários também podem usar esse objeto para fazer um novo diretório, apagar-se, mudar o nome em si, ou definir a hora da última modificação.

O objeto RandomAccessFileObject fornece métodos para ler e escrever conteúdo no FileObject correspondente. O design é semelhante ao java.IO.RandomAccessFile, para os desenvolvedores. Ao contrário dos objetos streaming, que podem apenas ler ou escrever em seqüência, o RandomAccessFileObject oferece um caminho para o front-end da interface GridFTP que permite facilmente ler ou gravar dados de qualquer posição no arquivo, que é útil para o modo de bloqueio estendido. Consequentemente, a fonte de dados subjacente deve suportar acesso aleatório para que ele possa ser conectado a este framework.

4. Ambiente e Resultados Experimentais

O ambiente experimental consiste em máquina desktop (PC) e um netbook; para fins do experimento no desktop foi instalado o Ubuntu server versão 9.04 (Processador: Core intel i5, 2,90 GHz, Disco Rígido: 500 GB, Memória RAM: 4 GB), um simulador de ambiente grade, o GridSim (2010), e rodando sobre ele o Globus Toolkit. Enquanto que no netebook foi instalada a versão mais atual do Ubuntu a 11.10 (Processador: Atom 1,33GHz de CPU, Disco Rígido: 320 GB, Memória RAM: 2 GB) Os testes foram conduzidos em um ambiente controlado, e as transferências realizadas em uma rede residencial via intranet, com link de velocidade de 100,00 MBytes.

Estes experimentos têm por objetivo demonstrar que o desempenho das transferências não é afetado pela utilização do Griffin. No primeiro teste, foi comparado a taxa de transferência do protocolo original com GridFTP via Griffin.

IRODS 2.1 foi instalado como fonte de dados. Griffin foi configurado no servidor com 786MB de tamanho de pilha como a interface GridFTP para este iRODS. Com esta configuração, os dados em iRODS podem ser acessado via interface nativa iRODS ou interface GridFTP. Foram comparamos a taxa de upload e taxa de download entre Servidor e cliente, com tamanhos de arquivo de 256MB, 512MB, 1GB, 2GB, 4GB, 8GB a 16GB, e oito threads são usadas tanto para transferências de iCommands quanto Griffin.

Tendo cada teste sido executado 5 vezes, o resultado na Figura 6 mostra a taxa de transferência média em Kbytes por segundo. A partir da figura, ao fazer upload de um arquivo, iCommands e Griffin executam a uma taxa semelhante, para arquivos menores que 8 GB, no entanto, se o arquivo é de 8GB ou mais, Griffin é mais rápido que iCommands.

Ao fazer o download, Griffin funciona de forma constante para os diversos tamanhos no teste, enquanto iCommands executa mais rápido para arquivos pequenos, mas fica mais lento conforme o tamanho do arquivo aumenta.

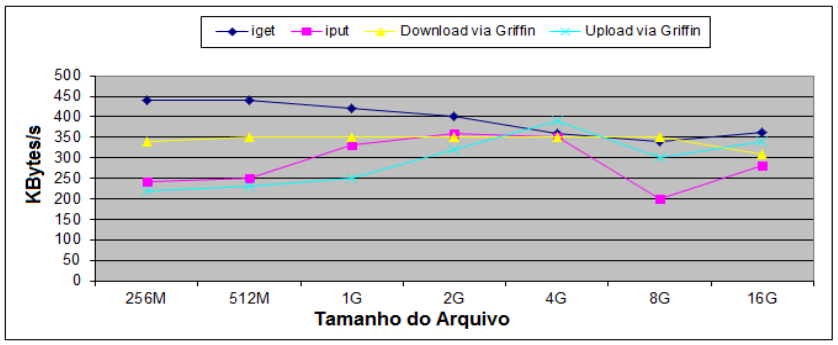

Figura 6. Comparação entre iCommands e Griffin

No segundo teste, é comparado o Griffin com o servidor Globus GridFTP, no mesmo teste ambiente, como os testes acima. O servidor Globus GridFTP foi instalado no servidor a partir VDT 1.10.1. Griffin foi reconfigurado com um adaptador de sistema de arquivos local, de modo que lê e grava dados no sistema de arquivos local.

Neste teste, os dados de testes são armazenados na mesma partição que o recurso iRODS. Semelhante ao primeiro teste, foram transferidos arquivos com tamanhos de 256M, 512M, 1G, 2G, 4G, 8G e 16G, via Griffin e Globus GridFTP, com 16 threads. Cada teste de transferência foi feito 5 vezes e os valores médios são mostrados na Figura 7. O resultado mostra que Griffin executa muito próximo do servidor Globus GridFTP em upload e download de dados.

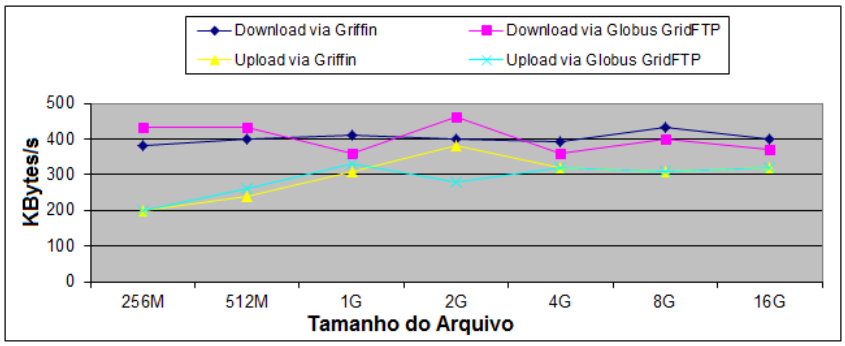

Figura 7. Comparação entre Servidor Globus GridFTP e Griffin

5. Conclusão e Trabalhos Futuros

Neste artigo foi apresentado à proposta e a implementação de um componente para transferência de arquivos com grande tamanho, independente do protocolo utilizado no ambiente de grid. A implementação foi baseada em Java sendo uma alternativa para o servidor Globus GridFTP. Algumas características diferenciais do componente são sua autonomia, não dependência da pilha de software Globus e pode ser executado na maioria dos sistemas operacionais.

Como trabalho futuro, pretende-se pesquisar o suporte UDT como um elemento a ser adicionado ao pacote de software Griffin, em adição a mecanismos de verificação deverão ser implementados para verificar a integridade de arquivos copiados. Outro caminho de pesquisa interessante é a investigação de múltiplos fluxos do Griffin para o fonte de dados será em casos onde a fonte de dados suporte múltiplos fluxos. A segmentação de transferências é outro recurso útil a ser considerado para um futuro projeto, posto que este permite o envio de um arquivo grande a partir de vários servidores GridFTP com cada um enviando uma parte do arquivo para obter melhor desempenho do que o uso de apenas um servidor GridFTP.

Referências

Allcock, W. (2003) “GridFTP: Protocol extensions to FTP for the Grid,” GGF Document Series GFD.20, Apr. 2003. Disponível em http://www.gridforum.org/GFD.20.pdf.

Anglano, C., Canonico, M. (2004). The File Mover: an efficient data transfer system for Grid applications, 27 set. 2004.

Arthur, J., Azadegan, S. (2005). "Spring Framework for rapid open source J2EE Web Application Development: A case study," in Proc of the Sixth Int Conf on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing(SNPD’05), 2005.

Baru, C., Moore, R., Rajasekar, A. (1998) "The SDSC Storage Resource Broker," in Proc of the Centre for Advanced Studies on Collaborative Research (CASCON), 1998.

Cern. (2011) glite. Disponível em http://glite.web.cern.ch/glite/. Acessado em Junho de 2011.

Condor.(2011). Disponível em http://www.cs.wisc.edu/condor/. Acessado em Maio de 2011.

Ernst, M., Fuhrmann, P., Gasthuber, M. (2001) "dCache, a distributed data storage caching system," in Computing in High Energy and nuclear Physics (CHEP2001), Beijing, 2001.

Foster, I., Kesselman, C. (2003). Globus: A Metacomputing Infrastructure Toolkit. International Journal of Supercomputer Aplications 11, 2, 115-128. Disponível em: ftp://ftp.globus.org/pub/globus/papers/globus.pdf. Acesso em: Novembro de 2003.

Fowler, M. (2004). Inversion of Control Containers and the Dependency Injection pattern.(29 April 2010). http://www.martinfowler.com/articles/injection.html.

Globus, The Project. (2002). GridFTP update January 2002, 2002. Disponível em: http://www.globus.org/datagrid/deliverables/GridFTP-Overview-200201.pdf.

Globus Toolkit. (2011). Disponível em http://www.globus.org/. Acessado em Junho de 2011.

GridSim. (2011). Disponível em http://www.buyya.com/papers/gridsim.pdf. Acessado em Outubro de 2011.

Laszewski, G. V., Foster, I., Gawor, J. (2001). "A Java Commodity Grid Toolkit," Concurrency: Practice and Experience, vol. 13, 2001.

Lu, D.; Quao, Y., Dinda, P., Bustamente, F. (2005) “Modeling and taming parallel TCP on the wide area network,” in Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, Apr. 2005.

Mandrichenko, I., Allcock, W., Perelmutov, T. (2005). GridFTP v2 protocol description. GGF Document Series GFD.47, May 2005. Disponível em http://www.ggf.org/documents/GFD.47.pdf.

Postel, J. (1981) “Transmission control protocol,” Request for Comments (RFC) 793, Sept. 1981.

Postel, J., Reynolds, J. (1985). File transfer protocol (FTP). Request for Comments (RFC) 959, Oct. 1985.

Rajasekar, A., Wan, M., Moore, R. (2006) "A Prototype Rule-based Distributed Data Management System," in High Performance Parallel and Distributed Computing (HPDC), Paris, França, 2006.

Savage, S. (1999). The End-to-End Effects of lntemet Path Selection. In Proc. of ACM SIGCOMM, pages 289-299, Boston, MA, 1999b.

Semke, J., Mahdavi, J., Mathis, M. (1998) “Automatic TCP buffer tuning,” in Proceedings of ACM SIGCOMM ’98, vol. 28, Oct. 1998.

Zhang, S., Coddington, P., Wendelborn, A. (2010). Connecting arbitrary data resources to the grid. In Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on Oct. 2010.