UNIVERSIDADE SAGRADO CORAÇÃO

JÔNATAS MINUCCI BALDERRAMAS

TRANSPARÊNCIA EM SISTEMAS DE ARQUIVOS DISTRIBUÍDOS.

BAURU
2012

JONATAS MINUCCI BALDERRAMAS

TRANSPARÊNCIA EM SISTEMAS DE ARQUIVOS DISTRIBUÍDOS.

Trabalho de Conclusão de Curso apresentado à Universidade Sagrado Coração, para a obtenção do título de Bacharel em Ciência da Computação, sob orientação do Professor Esp. Henrique Pachioni Martins.

BAURU

2012

Balderramas, Jonatas Minucci
B176t

Transparência em sistemas de arquivos distribuídos / Jonatas Minucci Balderramas -- 2012.
70f. : il

Orientador: Prof. Esp. Henrique Pachioni Martins.

Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade do Sagrado Coração - Bauru - SP

1. DFS. 2. Linux. 3. Sistemas de arquivos distribuídos. 4. Transparência. 5. Windows. I. Martins, Henrique Pachioni. II. Título.

JONATAS MINUCCI BALDERRAMAS

TRANSPARÊNCIA EM SISTEMAS DE ARQUIVOS DISTRIBUÍDOS

Trabalho de Conclusão de Curso apresentado ao Centro de Ciências Exatas e Sociais Aplicadas da Universidade Sagrado Coração como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação, sob orientação do Prof. Esp. Henrique Pachioni Martins.

Banca examinadora:

_____

Prof. Esp. Henrique Pachioni Martins
Universidade Sagrado Coração

_____

Prof. Esp. Andre Luiz Ferraz Castro
Universidade Sagrado Coração

_____

Prof. Dr. Elvio Gilberto da Silva
Universidade Sagrado Coração

Bauru, 19 de novembro de 2012.

AGRADECIMENTOS

Agradeço primeiramente a Deus por me conceder a oportunidade de concluir o curso de Ciência da Computação, aos meus pais Antônio Carlos Balderramas Afonso e Sueli Minucci Balderramas que além de financiarem meus estudos com muitos sacrifícios também sempre me apoiaram e admoestaram com imenso amor, aos meus avós Olavo Minucci e Maria de Lourdes Devidé Minucci pelo enorme carinho, apoio e por suas constantes orações, ao Prof. Esp. Henrique Pachioni Martins por me orientar com sabedoria, paciência e dedicação, aos demais professores, mestres e doutores do corpo docente da Universidade do Sagrado Coração que me auxiliaram prontamente sempre que necessário, aos meus amigos pelo incentivo e pela paciência em entender minha ausência em diversas ocasiões e por fim a minha amada namorada, por ser a pessoa mais carinhosa e companheira do mundo.

RESUMO

O Sistema de Arquivos é responsável pela organização, armazenamento, recuperação, denominação, compartilhamento e proteção de arquivos. Por muito tempo o Sistema de Arquivos tinha abrangência local, mas a presença constante das Redes de Computadores propiciou o surgimento e a disseminação de Sistemas de Arquivos Distribuídos (SAD), que tem a mesma incumbência do Sistema de Arquivos, mas agrega abrangência local e externa. Os SAD são fortemente explorados por grandes empresas como Google, Microsoft e IBM. Estas investem pesado em projetos de sistemas que objetivam o armazenamento de arquivos e processamento distribuídos. Estes projetos de SAD individualmente são direcionados à resolução de problemas específicos que vão desde o reaproveitamento de espaço ocioso em discos até a possibilidade do cliente trabalhar com seus arquivos tendo sua estação de trabalho desconectada da rede. Para que um SAD possibilite acesso a arquivos armazenados em um servidor com desempenho e confiabilidade semelhante aos arquivos armazenados em computadores locais é preciso enfrentar um dos maiores desafios presentes na implantação destes sistemas, a Transparência. Esta pesquisa tem o intuito de identificar, através da aplicação de experimentos, o grau da transparência de replicação, transparência de acesso e transparência de localização existente nas ferramentas de SAD em diferentes sistemas operacionais.

Palavras Chave: Armazenamento Distribuído, Sistemas de Arquivos distribuídos (SAD), Transparência.

.

ABSTRACT

The file system is responsible for organizing, storing, retrieving, naming, sharing and protecting files. For a long time File System coverage was local, but the constant presence of Computer Networks fostered the emergence and spread of Distributed File Systems (DFS), which has the same task File System, but aggregates breadth local and foreign. The SAD are heavily exploited by large companies like Google, Microsoft and IBM. They invest heavily in systems projects aimed file storage and distributed processing. These projects are targeted DFS individually to solve specific problems ranging from the reuse of idle space on disks to the client's ability to work with your files and your workstation disconnected from the network. For a DFS allows access to files stored on a server with performance and reliability similar to the files stored on local computers must face one of the biggest challenges in the deployment of these systems, Transparency. This research aims to identify, through the application of experiments, the degree of replication transparency, access transparency and location transparency in existing tools DFS on different operating systems.

Key words: Distributed Storage, Distributed File Systems (DFS), Transparency.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

AFS – Andrew File System
AVSG - Accessible Volume Storage Group
CPU – Central Processing Unit
DFS – Distributed File system
FTP - File Transfer Protocol
IBM – International Business Machines
ISO - International Organization for Standardization
LAN – Local Area Network
MAN – Metropolitan Area Network
NFS – Network File System
RAM - Random Access Memory
RPC - Remote Procedure Call
RPC2 - Remote Procedure Call (2ª versão)
SAD – Sistemas de Arquivo Distribuídos
UDP - User Datagram Protocol
VFS – Virtual File System
VSG - Volume Storage Group
WAN – Wide Area Network

SUMÁRIO

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

No princípio da informática os discos flexíveis eram a única forma de compartilhar dados entre computadores pessoais, mas estas mídias não ofereciam alto nível de confiabilidade, podendo ser facilmente corrompidas ou perdidas. Com o passar do tempo surgiram outras mídias mais confiáveis que os discos flexíveis, a exemplo do CD, DVD ROM e derivados, mas estes também se tornaram obsoletos após alguns anos. Com o advento de tecnologias como o *Universal Serial Bus* (USB) e IEEE-1394 (*Firewire*), o armazenamento em mídia removível foi revitalizado. Entretanto, apesar de sua robustez e elevado grau de confiabilidade em relação aos antigos discos flexíveis, estes dispositivos também não garantem a total disponibilidade dos dados, sendo suscetíveis a problemas eletrônicos, furtos e perdas.

De acordo com Dabek (2001) a vulnerabilidade das mídias de armazenamento portátil aliadas a crescente oferta de largura de banda e a redução acentuada do custo para utilização de meios de transmissão, os protocolos HTTP, FTP e serviços *Peer-to-Peer* (P2P) tornaram-se alternativas acessíveis para o armazenamento e transferência de arquivos entre computadores pessoais.

Alguns analistas prevêem que, no futuro, os dados que ainda são armazenados localmente, residirão em servidores existentes na Internet. Diversas empresas já possuem serviços que exploram esta forma de armazenamento, como por exemplo, o *BeInSync* e o *MediaMax* da Streamload (BEINSYNC, 2007; MEDIAMAX, 2007).

Alguns sistemas operacionais também possuem ferramentas nativas que possibilitam o armazenamento distribuído de arquivos, a exemplo do Microsoft Windows Server, que em algumas versões é integrado pelo DFS. Esta ferramenta possibilita acessar recursos compartilhados na rede de uma forma centralizada, sem que o usuário precise saber em qual servidor ou computador o recurso compartilhado se encontra.

De acordo com Chow (1998) os papeis de um Sistema de Arquivos Distribuído são semelhantes aos dos Sistemas de Arquivos convencionais, ou seja, ambos são Responsáveis pela organização, armazenamento, recuperação, denominação, compartilhamento e proteção de arquivos. Um sistema de arquivos distribuído é a implementação de um sistema de arquivos cujos locais de armazenamento estão dispersos fisicamente, mas provê ao usuário uma visão centralizada, semelhante aos sistemas de arquivos locais.

Silberschatz (2000) complementa que a diferença básica entre eles é que os SAD podem ler e gravar arquivos não apenas em unidades de armazenamento locais como também em outras unidades em computadores espalhados através de uma rede, seja esta de abrangência local (LAN) ou global (WAN). Silberschatz complementa que o sistema de arquivos distribuído é composto de clientes, servidores e dispositivos de armazenamento dispersos na rede, onde não há um único repositório de arquivos, mas sim diversos, sendo eles independentes.

É evidente que os sistemas distribuídos representam um grande avanço para a informática, oferecendo um conjunto de novas possibilidades e abrindo caminho para novas tecnologias, mas existem também alguns conceitos implícitos a estes sistemas que devem ser consideradas ao debatermos sobre um SAD, a exemplo da transparência. O grau de importância destes conceitos na implementação dos sistemas de arquivos distribuídos, além de depender do ambiente a que são aplicados, recebem tratamentos diferentes entre os autores da área de sistemas distribuídos. De acordo com Singhal (1994), os dois serviços de maior importância presentes nos sistemas de arquivos distribuídos são o servidor de nomes e o gerenciador de *Cache*. Estes serviços são, respectivamente, responsáveis pelo mapeamento de arquivos e diretórios, e pelo armazenamento de cópias dos dados de usuários em seus computadores.

Chow (1998), por sua vez, ressalta a importância da dispersão e da multiplicidade, onde múltiplos clientes dispersos fisicamente podem acessar diversos arquivos residentes em servidores também espalhados em diversas localizações.

Nos capítulos seguintes deste trabalho de conclusão de curso é apresentada uma revisão bibliográfica dos desafios, conceitos e problemas presentes na

implementação de sistemas distribuídos e sistemas de arquivos distribuídos, focando-se basicamente na questão da transparência de acesso, transparência de localização e transparência de replicação. Na segunda parte deste trabalho de conclusão de curso são realizados experimentos com o DFS, que é um sistema de arquivos distribuído padrão utilizado pelo Microsoft Windows para identificar o grau de sua transparência de replicação, transparência de acesso e transparência de localização e compará-las com a implementação existente do DFS para a plataforma Linux.

1.2 PROBLEMA

Como identificar e comparar os níveis de transparência de replicação, transparência de acesso e transparência de localização oferecidos pelo DFS operando no Microsoft Windows Server e no Linux.

2 OBJETIVOS

2.1 OBJETIVOS GERAIS

Definir um método para identificar e comparar os níveis da transparência de acesso, localização e replicação do DFS no Microsoft Windows Server e no Linux.

2.2 OBJETIVOS ESPECÍFICOS

- Estudar o funcionamento dos sistemas de arquivos distribuídos.
- Estudar o funcionamento do DFS.
- Desenvolver os testes na plataforma Microsoft Windows.
- Desenvolver os testes com o DFS na plataforma Linux.
- Relacionar os resultados e realizar a comparação entre as plataformas.

3 JUSTIFICATIVA

Os sistemas distribuídos são sem dúvidas um dos temas mais debatidos no âmbito da informática, os maiores nomes da indústria de *Softwares* do mundo, como Google, Microsoft, Apple, IBM e diversos outros tem investido fortemente nestes sistemas que, de um grosso modo, objetivam a computação em geral de maneira distribuída.

O conceito de *Cloud Computing* (Computação em Nuvem) que se refere à utilização da memória e das capacidades de armazenamento e cálculo de computadores e servidores compartilhados e interligados por meio da Internet e a *Grid Computing* (Computação em Grade) que é um modelo computacional capaz de alcançar uma alta taxa de processamento dividindo tarefas entre diversas máquinas interconectadas através de uma rede local ou redes de longa distância, formando uma espécie de máquina virtual são exemplos de sistemas que operam de maneira distribuída.

Kirner (1988) afirma que estes conceitos de computação distribuída tiveram êxito graças às redes de computadores, que já estão presentes em praticamente todos os lugares. As primeiras evidências de redes de computadores apareceram em meados da década de 60 com o projeto e implementação de sistemas centralizados, contendo um grande número de terminais espalhados.

A evolução tecnológica das redes de computadores juntamente com as reduções dos custos de *hardware* necessários para sua implementação consequentemente ocasionaram reduções significativas nos custos destas comunicações, propiciando então as condições favoráveis à disseminação das redes de computadores.

Segundo Kon (1994) inicialmente, as redes de computadores eram aplicadas a problemas simples, como a troca de mensagens entre usuários de diferentes máquinas e o compartilhamento de impressoras ou de discos rígidos apenas para leitura, entretanto, com o passar do tempo, foi surgindo sistemas mais complexos que ofereciam ao usuário um maior aproveitamento dos recursos distribuídos pela rede, a exemplo dos sistemas distribuídos e sistemas de arquivos distribuídos.

O predomínio de sistemas acessíveis pela internet, denominados de sistemas *on-line*, a presença constante da internet no cotidiano de grande parte da população mundial e o surgimento de novas ideologias relacionadas a computação distribuída são alguns dos fatos que enfatizam o domínio dos sistemas distribuídos e sistemas de arquivos distribuídos em um futuro gradativamente menos distante. Contudo, existem diversos problemas e particularidades relacionados aos sistemas distribuídos e seus derivados.

Uma das características mais marcantes sobre os sistemas distribuídos é sem dúvidas a transparência. Muitos autores a consideram um dos maiores desafios encontrados ao se implantar sistemas distribuídos. Tanenbaum define os sistemas distribuídos como sendo uma coleção de computadores independentes que aparecem para os usuários do sistema como um único computador, ou seja, se os usuários perceberem de alguma forma que o sistema é gerenciado, processado ou até armazenado por diversos nós (computadores ou servidores), o sistema distribuído não foi capaz de alcançar seu objetivo (TANENBAUM, 2002, p. 2).

Kirner (1988) complementa que o compartilhamento de recursos computacionais pode existir de forma explícita ou implícita. No compartilhamento explícito, a rede oferece a possibilidade dos usuários acessarem e manipularem diretamente os recursos remotos, exigindo que os clientes conheçam os detalhes dos recursos, enquanto que no compartilhamento implícito, o acesso é feito automaticamente pelo sistema de forma transparente para o usuário. Sistemas com compartilhamento de recursos de comunicação e compartilhamento implícito de recursos acabaram, em grande parte, constituindo-se nos sistemas distribuídos.

De acordo com Ribeiro (2005) os sistemas distribuídos foram criados para distribuir as tarefas e aumentar o poder computacional através do uso de vários processadores como também promover o compartilhamento de recursos. Cada processador possui sua própria memória e a comunicação entre os processadores é feita através de linhas de comunicação. Esses sistemas objetivam melhorar a comunicação entre os computadores, propiciando a integração destes num sentido amplo, que pode envolver a facilidade de mudanças futuras, rapidez nas trocas de informações e confiabilidade na execução dos processos. Eles permitem que uma

aplicação seja dividida em diferentes partes que podem ser processadas em sistemas independentes que se comunicam através das linhas de comunicação. Ele cria a ilusão de que a rede de computadores seja vista como um único sistema de tempo compartilhado, em vez de um conjunto de máquinas distintas.

É notável que os sistemas distribuídos tenham uma relação muito estreita com a transparência, como já enfatizado nas citações apresentadas anteriormente neste capítulo, Kirner (1988) afirma que se um usuário necessitar de conhecimentos detalhados dos recursos da rede este compartilhamento passa a ser explícito e que os sistemas distribuídos por sua vez derivam de compartilhamentos implícitos. Uma coleção de computadores deve aparecer ao usuário como um único computador para que possa ser denotado como um sistema distribuído.

Após tantas discussões sobre sistemas distribuídos tona-se claro que esta tecnologia já proporcionou grandes inovações para a informática e ainda tem muito a oferecer se continuar a ser explorada, porém uma questão importante a ser observada é a aplicação das ferramentas de sistemas de arquivos distribuídos em plataformas distintas. Grandes redes de dados com abrangência global (WAN) abrigam diversos equipamentos distintos, a exemplo dos servidores que executam sistemas operacionais distintos e protocolos de comunicação distintos. Por este fato muitos *Softwares* precisaram receber tratamentos para tolerar as diferenças de arquitetura dos equipamentos com os quais precisavam trabalhar, estas modificações de *Software* permitiram a heterogeneidade destas ferramentas e protocolos em diferentes sistemas operacionais.

A questão da heterogeneidade também pode afetar as ferramentas de sistemas de arquivos distribuídos quando estas são aplicadas em redes globais ou em topologias em que existam diferenças de *hardware* e *software*. Por este fato algumas ferramentas já possuem implementações para diferentes sistemas operacionais, no entanto existem casos em que ferramentas adaptadas perderam algumas de suas características originais, causando assim alguns inconvenientes e reduzindo sua viabilidade.

Neste trabalho de conclusão de curso foram realizados experimentos com o DFS em duas plataformas distintas, o Microsoft Windows Server 2008 R2 e o Linux

Ubuntu, com o intuito de testar e comparar as transparências de Acesso, Replicação e Localização e utilizá-las como meio para comparar a heterogeneidade do DFS para as plataformas testadas de modo que se possa então avaliar se os usuários do SAD podem notar alguma diferença durante a utilização do DFS sendo executado em diferentes plataformas.

4 REVISÃO DA LITERATURA

4.1 SISTEMA DE ARQUIVOS

No âmbito da informática, para que informações possam ser salvas, elas são armazenadas em forma de arquivos. De acordo com Tanenbaum (2000) os arquivos podem ser definidos como uma seqüência de itens de dados, ou seja, um arquivo é uma sequência finita de dados organizados e armazenados em blocos acessíveis pelas operações de leitura e escrita de qualquer parte dessa sequência. Um arquivo pode representar qualquer tipo de informação, como por exemplo, textos, músicas, imagens, vídeos e muitos outros .

De acordo com Coulouris (2005) o sistema de arquivos tem um papel fundamental para a informática por permitir que o sistema operacional leia e armazene informações em forma de arquivos nas unidades de armazenamento disponíveis. Tecnicamente os sistemas de arquivos são responsáveis pela organização, armazenamento, recuperação, atribuição de nomes, compartilhamento e proteção de arquivos. Eles fornecem uma *interface* de programação que caracteriza a abstração de arquivo, liberando os usuários da preocupação com os detalhes da alocação e do armazenamento físico no disco.

Segundo Silberschatz (2000) os sistemas de arquivos são projetados para armazenar e gerenciar um grande número de arquivos, com recursos para criação, atribuição de nomes e exclusão de arquivos. Os sistemas de arquivos também assumem a responsabilidade pelo controle de acesso aos arquivos, restringindo o acesso de acordo com as autorizações dos usuários e com o tipo de acesso solicitado (leitura, atualização, execução etc.).

4.1.1 NOMEAÇÃO TRANSPARENTE

Os usuários de um sistema de arquivos trabalham com arquivos de maneira lógica, ou seja, os dados são representados por nomes de arquivos e estes arquivos geralmente estão organizados em diretórios e pastas. O sistema, por sua vez,

manipula blocos físicos de dados, armazenados em trilhas de discos. A nomeação é um mapeamento entre objetos lógicos e físicos capaz de fornecer ao usuário uma abstração de um arquivo que oculta os detalhes de como e onde ele está armazenado de fato no disco físico.

Dada a localização lógica de um arquivo, ou seja, o caminho até ele, é necessário analisar os componentes deste caminho a fim de encontrar sua localização física. É necessário descobrir em quais blocos de quais discos e de quais servidores se encontra o arquivo em questão. Quando os arquivos de um sistema estão distribuídos entre vários servidores localizados em diferentes máquinas, é desejável que a localização destes dados seja transparente aos usuários do sistema.

Em um sistema transparente, uma nova dimensão é adicionada a abstração, a de ocultar o local na rede onde o arquivo esta armazenado. Idealmente, um serviço de sistema de arquivo distribuído deve aparecer aos seus clientes como um sistema de arquivos centralizado-convencional. A multiplicidade e a dispersão de seus servidores e dispositivos de armazenamento devem ser transparentes. Ou seja, a *interface* cliente desse serviço não deve fazer distinção entre arquivos locais e remotos. Cabe ao serviço localizar os arquivos e organizá-lo para o transporte dos dados. Um sistema de arquivos distribuído transparente facilita a mobilidade do usuário trazendo o ambiente do usuário (ou seja, seu diretório inicial) para onde quer que o usuário efetue login (SILBERSCHATZ, 2000, p. 542).

Diversas técnicas de transparência, adotadas em sistemas distribuídos são empregadas parcialmente ou diretamente na implementação de serviços de arquivos.

4.1.2 CACHE

De acordo com Singhal (1994) o cache consiste em uma técnica para aumentar o desempenho na execução de processos em diversos sistemas. O cache armazena na memória fragmentos de dados que estão sendo constantemente utilizados pelo sistema, ou seja, os dados que possuem o maior número de

solicitações. Dessa forma a informação precisa ser lida diretamente do disco uma única vez. Depois de armazenada na memória, o acesso a esta informação é significativamente mais rápido, comparando-se com a velocidade de acesso aos meios de armazenamento, visto que a velocidade de acesso à memória é superior a velocidade de acesso a um disco rígido, por exemplo, que necessitaria mover o braço do leitor até a trilha em que se encontram os dados e esperar até que a rotação do disco traga-os à cabeça de leitura, evitando desta forma uma possível sobrecarga para obter a informação do sistema novamente.

O Cache também é aplicado em sistemas de arquivos distribuídos por auxiliar na economia do tempo de processamento. Para acessar informações remotas, por exemplo, o sistema estará limitado a velocidade da rede, que, mesmo rápida, estará limitado a velocidade do meio físico do servidor remoto, pois este ainda necessitará procurar e processar os dados solicitados, carregando-os na memória e enviando-os para o cliente.

De acordo com Carvalho (2003) em sistemas de arquivos distribuídos, o cache pode estar presente tanto no cliente, como no servidor, evitando assim que o usuário faça uso desnecessário da rede para obter informações antes já solicitadas, enquanto que o servidor diminui o acesso ao meio físico de armazenamento dos dados para enviá-los ao cliente.

4.2 SERVIÇOS DO SISTEMA DE ARQUIVOS DISTRIBUÍDOS

Nesta sessão são apresentados três serviços intrínsecos aos sistemas de arquivos distribuídos que facilitam o acesso às informações para os usuários, escondendo toda a complexidade existente na nomeação global, no acesso aos arquivos dispersos pela rede e na organização destes que são o serviço de nomes, o serviço de arquivos e o serviço de diretórios, respectivamente.

4.2.1 SERVIÇO DE NOMES

Segundo Galli (2000), o serviço de nomes tem a função de fornecer a localização de um determinado arquivo no conjunto de computadores do sistema de

arquivos distribuído. Caso o nome do arquivo contiver o nome do computador aonde ele esta localizado, como por exemplo, “apolo:/tmp/arquivo”, então este serviço não provê transparência de localização. Para prover esta transparência, o nome de um arquivo não deve possuir indícios de sua localização física, pois caso ele mude de lugar ou possua várias cópias, o seu nome ou caminho não precisarão ser alterados.

4.2.2 SERVIÇO DE ARQUIVOS

Conforme Galli (2000) o serviço de arquivos é responsável por fornecer os mesmos serviços e recursos de um sistema de arquivos tradicional. A diferença é que um sistema de arquivos distribuído pode ser acessado de qualquer estação de trabalho dentro da rede. Esse serviço também cuida das propriedades dos arquivos, como data de criação, data de alteração, tamanho, proprietário, permissões de acesso e outras informações relevantes, além de manter a integridade das operações realizadas nos arquivos.

4.2.3 SERVIÇO DE DIRETÓRIOS

Segundo Galli (2000) o objetivo do serviço de diretórios é manter a organização dos arquivos armazenados no sistema, fornecendo ao usuário uma *interface* para que eles possam organizar seus arquivos em uma estrutura hierárquica, através de diretórios e subdiretórios.

4.3 CARACTERÍSTICAS DESEJÁVEIS EM SISTEMAS DE ARQUIVOS DISTRIBUÍDOS

Muitos dos conceitos encontrados na implementação de sistemas distribuídos devem ser levados em consideração no desenvolvimento de sistemas de arquivos distribuídos. Coulouris (2005) afirma que inicialmente, os primeiros sistemas de arquivos distribuídos ofereciam recursos de transparência de acesso e transparência de localização, emergindo subsequentemente à preocupação no desenvolvimento

de recursos como desempenho, escalabilidade, controle de concorrência, tolerância a falhas e segurança.

4.3.1 ATUALIZAÇÃO CONCORRENTE

As alterações feitas em um arquivo por um único cliente não devem interferir na operação de outros clientes que estejam acessando, ou alterando, o mesmo arquivo simultaneamente. Em muitos aplicativos, a necessidade de controle de concorrência para acesso a dados compartilhados é amplamente aceita, e as técnicas de implementação muitas vezes são dispendiosas.

O maior problema encontrado nas implementações desse tipo de solução é quanto a sincronização dos arquivos, o que inclui leitura e escrita concorrente. Carvalho (2003) afirma que a leitura concorrente pode ser implementada facilmente caso não haja escrita concorrente, pois quando um arquivo estiver sendo lido, certamente ninguém poderá alterá-lo.

Para implementar a escrita concorrente, deve-se levar em conta que, quando um cliente escreve em um arquivo, todos os leitores devem ser avisados que o documento foi alterado, tendo-se que tomar cuidado também para que estas alterações não desfaçam as alterações realizadas por outros usuários (COULOURIS, 2005, p. 139).

4.3.2 REPLICAÇÃO

A exigência básica de um esquema de replicação é que diferentes réplicas do mesmo arquivo residam em servidores independentes, ou seja, a disponibilidade de uma réplica não é afetada pela disponibilidade das demais.

Em um serviço de arquivos que suporta replicação, um arquivo pode ser representado por várias cópias de seu conteúdo em diferentes locais, trazendo como benefício a possibilidade dos servidores, que hospedam um mesmo conjunto de arquivos, compartilharem a carga do fornecimento de um serviço para clientes que acessam esses arquivos, melhorando assim a escalabilidade do serviço e também a

tolerância a falhas, permitindo que, em caso de falhas, os clientes localizem outro servidor que contenha uma cópia do arquivo. Kon (1994) menciona algumas vantagens importantes que podem existir com a replicação de arquivos:

- Caso um disco seja danificado, as informações nele contidas não são perdidas, podendo ser recuperadas de outros discos em outros servidores;
- Se um servidor está momentaneamente inoperante ou inacessível, os seus arquivos podem ser acessados em servidores alternativos, pois haverá uma maior disponibilidade do serviço de arquivos;
- Arquivos ou diretórios muito lidos podem ser oferecidos por vários servidores, distribuindo-se a demanda eqüitativamente entre os diversos servidores e aumentando o desempenho global do sistema.

Segundo Coulouris (2005) o principal desafio associado a replicação é a sua atualização e manutenção da consistência entre as réplicas dos arquivos nos diversos servidores. Para o usuário, as réplicas de um arquivo denotam a mesma entidade lógica e, dessa forma, uma atualização a qualquer réplica deve refletir nas demais. Os detalhes da replicação devem ser transparentes para os usuários, sendo tarefa do esquema de nomeação mapear um nome de arquivo replicado em um computador específico sem o conhecimento de sua localização pelo cliente.

Conforme Silberschatz (2000) poucos serviços de arquivos suportam replicação completa, mas a maioria suporta o armazenamento de arquivos, ou de porções de arquivos em caches locais, que é uma forma limitada de replicação.

4.3.3 HETEROGENEIDADE

Segundo Coulouris (2005) a Heterogeneidade possibilita que as *interfaces* de serviço sejam definidas de modo que o *software* cliente e servidor possa ser implementado para diferentes sistemas operacionais e computadores.

Uma das grandes deficiências dos sistemas computacionais tem sido a incompatibilidade tanto de *hardware* quanto de *software* fabricado por diferentes companhias.

Cada vez mais, buscam-se padronizações que permitam que máquinas de diferentes fabricantes se comuniquem sem grandes dificuldades. Segundo Kon (1994) O sistema de arquivos distribuído deve ser implementado de forma que o servidor e o cliente não necessitem da mesma arquitetura de *hardware* e da mesma solução de *software* para a correta execução do serviço.

4.3.4 TOLERÂNCIA A FALHAS

Dentre as características desejáveis de um sistema distribuído, a tolerância a falhas certamente possui grande destaque, sendo um forte motivo para a crescente procura pela implementação de sistemas distribuídos. É essencial que o serviço de arquivos distribuídos continue a funcionar diante de falhas de clientes e servidores.

Coulouris (2005) afirma que falhas de *hardware* ou de *software* podem ocasionar uma interrupção momentânea no funcionamento de uma máquina. A queda de um nodo, ou uma falha em um canal de comunicação pode levar a uma partição na rede, ou seja, a conversação entre dois pontos da rede é interrompida durante certo intervalo de tempo.

Segundo Kon (1994) a grande maioria dos sistemas de arquivos distribuídos é sensível a partições na rede quando estão em operação, e caso o cliente não consiga estabelecer contato com alguns dos servidores, os processos que fizeram solicitações aos serviços de arquivos serão notificados de que as informações solicitadas não estão disponíveis ou simplesmente são bloqueados até que a conexão se estabeleça.

O método mais utilizado para aumentar a disponibilidade de um serviço de arquivos tem sido a replicação de dados, onde os arquivos são armazenados em dois ou mais servidores e caso um deles não esteja disponível, outro servidor poderá fornecer os serviços solicitados.

4.3.5 CONSISTÊNCIA

Segundo Coulouris (2005) um sistema de arquivos distribuídos deve garantir a consistência dos arquivos de seus usuários.

Quando um arquivo é replicado ou recuperado do cache de uma estação cliente, por exemplo, podem ocorrer demoras inevitáveis na propagação das modificações devido a latências de rede, podendo resultar na geração de inconsistências ou até perda das informações caso o sistema seja frágil a esse tipo de cenário.

4.3.6 SEGURANÇA

A segurança é um assunto muito debatido no âmbito dos sistemas de arquivos distribuídos devido a sua arquitetura, que é quase que completamente dispersa por uma rede de comunicação local e, em alguns casos, global. Essa preocupação existe pelo fato de que uma comunicação estabelecida entre nodos do sistema pode ser interceptada por um usuário não autorizado, afinal compartilhar arquivos entre vários ambientes e usuários é uma das vantagens que os sistemas de arquivos distribuídos oferecem, entretanto, deixar que outros usuários possuam acesso a arquivos confidenciais é um grande problema, sendo necessário adotar mecanismos que garantam que pessoas não autorizadas não tenham acesso a tais informações.

Segundo Carvalho (2003) uma das maneiras mais comuns adotada por todos os sistemas de arquivos para assegurar a confidencialidade dos dados é o controle de acesso. Os mecanismos de controle de acesso fazem uso de listas de controle de acesso. Nos sistemas de arquivos distribuídos, há a necessidade de autenticar as requisições dos clientes para que o controle de acesso no servidor seja baseado nas identidades corretas de usuário e para proteger o conteúdo das mensagens de requisição e resposta com assinaturas digitais e, opcionalmente, criptografia de dados secretos.

4.3.7 EFICIÊNCIA

Os sistemas de arquivos distribuídos não dependem apenas da velocidade de transferência de dados dos meios de comunicação, a exemplo das redes de computadores. Mesmo que estas sejam demasiadamente velozes, uma requisição, além do tempo de transferência de pacotes imposto pelo meio, terá de aguardar que os pacotes de dados da requisição e da resposta obtida pelos servidores sejam processados e encapsulados antes de poder entregar as informações ao usuário.

Coulouris (2005) afirma que um sistema de arquivos distribuído deve fornecer um serviço que seja comparável (ou melhor) aos sistemas de arquivos locais, em termos de desempenho e confiabilidade. Para isso foram desenvolvidas técnicas para contornar os atrasos ocorridos com a comunicação entre os nodos, entre elas destacam-se o cache, que acessa a rede o mínimo possível, armazenando as informações acessadas constantemente na memória e também o balanceamento de carga, que graças a existência de replicas dos arquivos espalhadas em diversos servidores, pode optar por requisitar um arquivo de um ou mais servidores que estejam menos sobrecarregados, ganhando assim em tempo de processamento.

4.3.8 TRANSPARÊNCIA

A transparência é definida como sendo o encobrimento do usuário e do programador da aplicação da separação dos componentes de um sistema distribuído, de modo que o sistema seja percebido como um todo ao invés de uma coleção de componentes independentes, ou seja, todos os processos do sistema que não forem relevantes ao usuário final deverão ser encobertos (COULOURIS, 2005, p. 23).

Estes processos incluem a acessibilidade, localização, concorrência de processos, replicação de recursos e outros. Para que a transparência completa possa ser alcançada diversos aspectos, identificados pela *International Organization for Standardization* (ISO), devem ser levados em consideração durante o desenvolvimento de um sistema distribuído, tarefa que torna-se complexa devido a

quantidade de requisitos a ser considerados. Existem outros tipos de transparência, mas este trabalho explorará as características e os conceitos individuas da transparência de acesso, da transparência de replicação e da transparência de localização para os sistemas distribuídos:

- *Transparência de Acesso*: Um único conjunto de operações fornece acesso a arquivos locais e remotos, permitindo que estes recursos sejam acessados com o uso de operações idênticas para todo o sistema.
- *Transparência de Localização*: Permite que os recursos sejam acessados sem conhecimento de sua localização física na rede. A transparência de localização permite também que recursos sejam movidos entre nodos de um sistema distribuído, como por exemplo, um determinado arquivo, o cliente não deve perceber qualquer alteração no mesmo, mesmo que este cliente esteja usando o arquivo quando este for realocado.
- *Transparência de replicação*: Permite que várias instâncias dos recursos sejam usadas simultaneamente visando aumentar a confiabilidade e o desempenho das tarefas do sistema, sem que o cliente possua qualquer conhecimento das réplicas existentes e de suas localizações.

São consideradas mais importantes as transparências de acesso e a de localização, pois o impacto ao sistema ocasionado por sua ausência ou presença é maior que as demais, visto que afetam mais fortemente a utilização de recursos distribuídos. Às vezes, elas são referidas em conjunto como transparência de rede.

Um exemplo da transparência de acesso seria uma *interface* gráfica, capaz de reunir diretórios locais e remotos de forma homogênea, como por exemplo, exibir todos os diretórios para o usuário em forma convencional de estruturas de pastas de arquivos, sem que este seja capaz de diferenciar diretórios locais e remotos. Na

ausência da transparência de acesso, os usuários do sistema seriam obrigados possuir conhecimentos profundos da infraestrutura de rede para ter acesso aos arquivos remotos, fazendo uso de ferramentas como, por exemplo, de aplicativos que operam sobre o protocolo FTP.

A transparência de localização permite que os recursos, dados e dispositivos que são compartilhados e acessados, não precisem e não devam conter nenhuma informação relevante que forneça ao usuário a sua localização ou a que região onde esse recurso está contido. Se um cliente precisar acessar um arquivo ou um documento qualquer, não deve ser relevante o fato de esse documento estar armazenado em seu próprio computador ou em um servidor localizado na Austrália, pois o importante é que o cliente tenha acesso ao documento, sem se importar com a localização física do recurso.

Coulouris (2005) ilustra a transparência de localização afirmando que esta pode ser exemplificada através dos nomes de recursos da Web, isto é, os URLs, pois a parte do URL que identifica o nome de um servidor Web se refere a um nome de computador em um domínio, em vez de seu endereço IP. Entretanto, os URLs não são transparentes à mobilidade, pois se a página Web muda para um domínio diferente, todas as referencias existentes nas outras páginas ainda apontarão para a página original.

Segundo Tanenbaum (2002) a transparência de replicação é exemplificada através dos recursos e dados da internet tratados como objetos, isso significa que várias instâncias dos recursos possam ser usadas simultaneamente, aumentando assim a disponibilidade e o desempenho de processos. A replicação influência fortemente na tolerância a falhas de um sistema distribuído, possibilitando que clientes continuem a ler e gravar arquivos mesmo que algum servidor por ventura fique indisponível durante este processo.

4.4 FERRAMENTAS DE SISTEMAS DE ARQUIVOS DISTRIBUÍDOS

Ao longo deste capítulo são apresentadas algumas das principais ferramentas de SAD do mercado.

4.4.1 *CODA FILE SYSTEM*

O *Coda File System* começou a ser desenvolvido em 1987 pela Universidade de Carnegie Mellon, nos EUA, sendo descendente da versão 2 do *Andrew File System* (AFS) .

Braam (1998) afirma que o principal objetivo do *Coda File System* é fornecer suporte a realização de operações desconectadas ao sistema de arquivos para computadores portáteis que costumam ficar grande parte do tempo fora da rede, provendo assim uma alta disponibilidade dos arquivos aos seus usuários. Para tanto, uma cópia do arquivo que o usuário esta manipulando é armazenada na máquina local, evitando assim que falhas de comunicação entre o cliente e o servidor prejudiquem o usuário.

O suporte a operações desconectadas surgiu com a proposta de criar um sistema de arquivos tolerante a falhas de rede entre clientes e servidores. O esforço empregado no *Coda File System* para lidar com este problema mostrou-se conveniente com o advento de clientes móveis. Quando um usuário atualiza um arquivo, as alterações precisam ser propagadas aos servidores que o armazenam e caso o cliente esteja conectado ao servidor, esta atualização é feita de forma síncrona, ou seja, a alteração é propagada no momento de sua gravação no cliente. Caso haja problemas nesta comunicação, ao invés de reportar o erro ao usuário, as informações são gravadas localmente em um registro de alterações pendentes e assim que a comunicação com os servidores for restabelecida, estas alterações são propagadas automaticamente (KISTLER, 1991, p.75).

4.4.2 ARQUITETURA – *CODA FILE SYSTEM*

Segundo Tanenbaum (2002) o *Coda File System* herdou muitas características arquitetônicas de seu descendente, o *Andrew File System*, que foi projetado para disponibilizar a cada aluno e professor da Universidade de *Carnegie Mellon* uma estação de trabalho com um sistema compatível com o Unix, onde os

usuários entrariam em qualquer computador da rede e a sua visão do sistema de arquivos deveria ser a mesma. Sendo assim, era essencial tomar o máximo de cuidado quanto a escalabilidade da ferramenta, uma vez que ela deveria atender a aproximadamente 10 mil estações de trabalho.

Conforme Kon (1994) para satisfazer essa exigência, o AFS é implementado em dois grupos de computadores, conforme ilustrado na Figura 1, um grupo consiste em um número relativamente pequeno de servidores de arquivos dedicados que são administrados de forma centralizada, denominados de *Vice*. O segundo grupo consiste em uma coleção maior de estações de trabalho, denominadas de Virtue, que concedem aos usuários acesso ao sistema de arquivos distribuído.

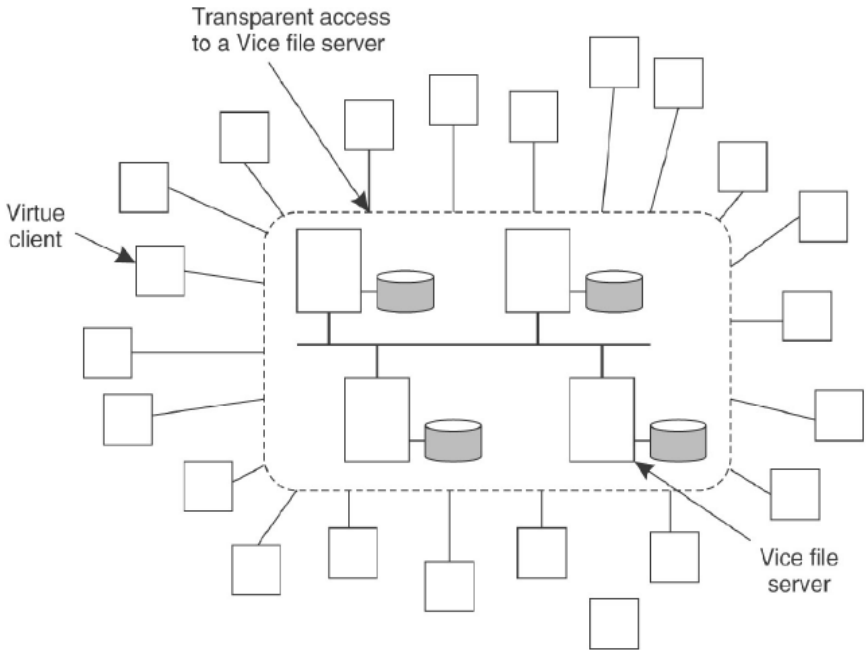

Figura 1 - Arquitetura do Andrew File System
Fonte: Tanenbaum, 2002, p. 605

O *Coda File System* segue esta mesma organização. Tanenbaum (2002) afirma que todas as estações de trabalho possuem um processo denominado de *Venus* que é executado em nível de usuário no sistema operacional e tem como

objetivo fornecer ao cliente o acesso ao sistema de arquivos distribuído. O processo *Venus* também é responsável por permitir que o cliente continue manipulando seus arquivos mesmo que a estação de trabalho não esteja conseguindo estabelecer conexão com o servidor, sendo esta sua principal característica em relação ao NFS.

A arquitetura interna de uma estação de trabalho do AFS é ilustrada na Figura 2. Assim como no NFS, ele faz uso da *interface* VFS que intercepta todas as chamadas do cliente e as encaminha ao sistema de arquivos local ou ao processo *Venus*, que por sua vez, estabelece conexão com o servidor *Vice* através das chamadas de procedimentos remotos.

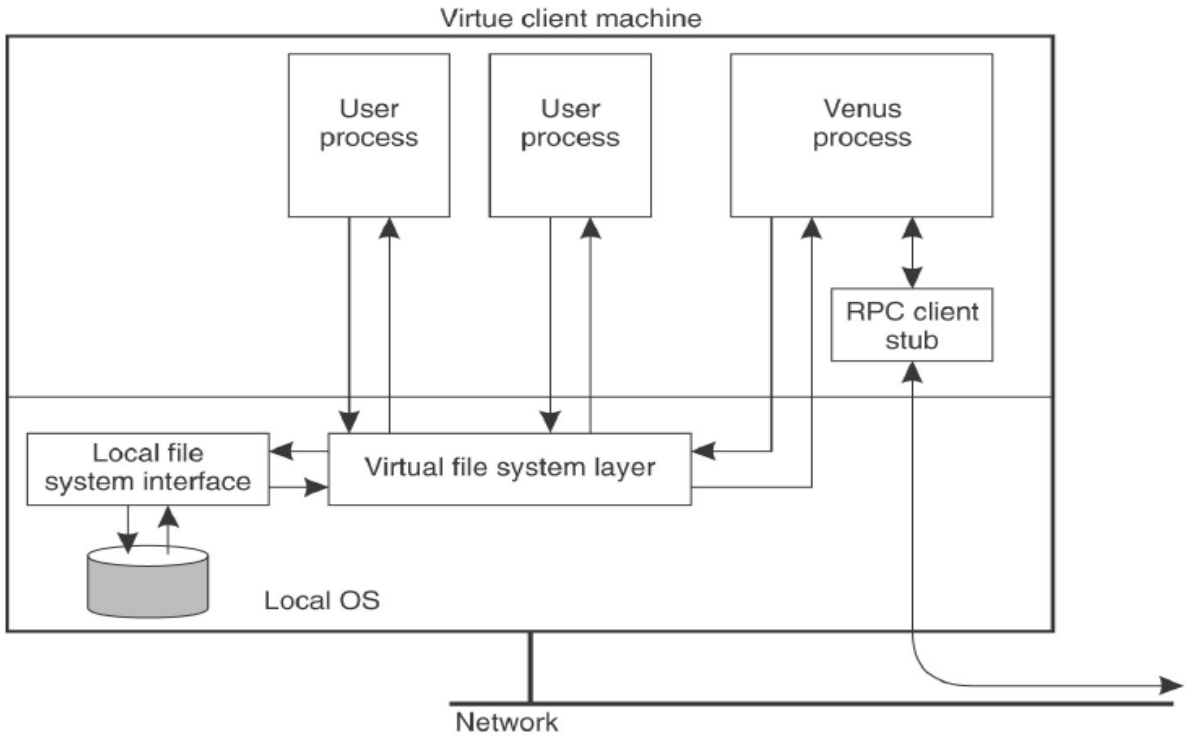

Figura 2 - Arquitetura interna de uma estação de trabalho do AFS
Fonte: Tanenbaum, 2002, p. 606

O *Coda File System* disponibiliza aos usuários uma visão de um sistema de arquivos local e suporta todas as operações do sistema operacional desde que sigam as especificações da *interface* VFS, que é semelhante à ilustrada na Figura 2.

4.4.3 COMUNICAÇÃO – *CODA FILE SYSTEM*

A comunicação entre os processos do *Coda File System* é realizada através de chamadas de procedimentos remotos. Entretanto, o *Coda File System* possui uma implementação mais sofisticada do RPC, denominada de RPC2, que oferece chamadas de procedimentos remotos seguras sobre um protocolo UDP.

A cada chamada de procedimento remoto, o cliente do RPC2 inicia um processo que envia um pedido de requisição do servidor e subseqüentemente os blocos da solicitação até receber uma resposta. Como o processamento do pedido pode levar um tempo arbitrário para ser concluído, o servidor envia regularmente mensagens ao cliente para que este saiba que seu pedido esta sendo processado. Caso ocorra uma falha de comunicação entre o cliente e o servidor, o processo iniciado no computador cliente perceberá que as mensagens cessaram e notificará a aplicação que o invocou do fracasso da operação (TANENBAUM, 2002, P. 134).

4.4.4 PROCESSOS – *CODA FILE SYSTEM*

O *Coda File System* mantém uma distinção entre os processos do cliente, representados através do *Venus* e processos do servidor, representados através do *Vice*, sendo que ambos utilizam uma biblioteca de threads não-preemptiva para permitir que as requisições sejam processadas concorrentemente no cliente (onde vários processos de usuário podem ter requisições de acesso a arquivo em andamento concorrentemente) e no servidor (TANENBAUM, 2002).

4.4.5 SERVIÇO DE NOMES – *CODA FILE SYSTEM*

Os arquivos no *Coda File System* agrupam-se em unidades denominadas de volumes. Os volumes se assemelham a uma partição do sistema operacional, mas geralmente possuem uma granularidade muito menor e correspondem a uma coleção de arquivos associados a um usuário. Os volumes são importantes por duas razões, primeiro porque eles formam a unidade básica pela qual o espaço de nomes

é constituído e esta construção ocorre quando eles são montados na estação cliente e a segunda razão é que eles formam a unidade que será replicada entre os servidores da rede.

É importante notar que, diferente do NFS, o *Coda File System* possui um espaço de nomes compartilhado. A Figura 3 representa esse compartilhamento exemplificando um ambiente com o AFS.

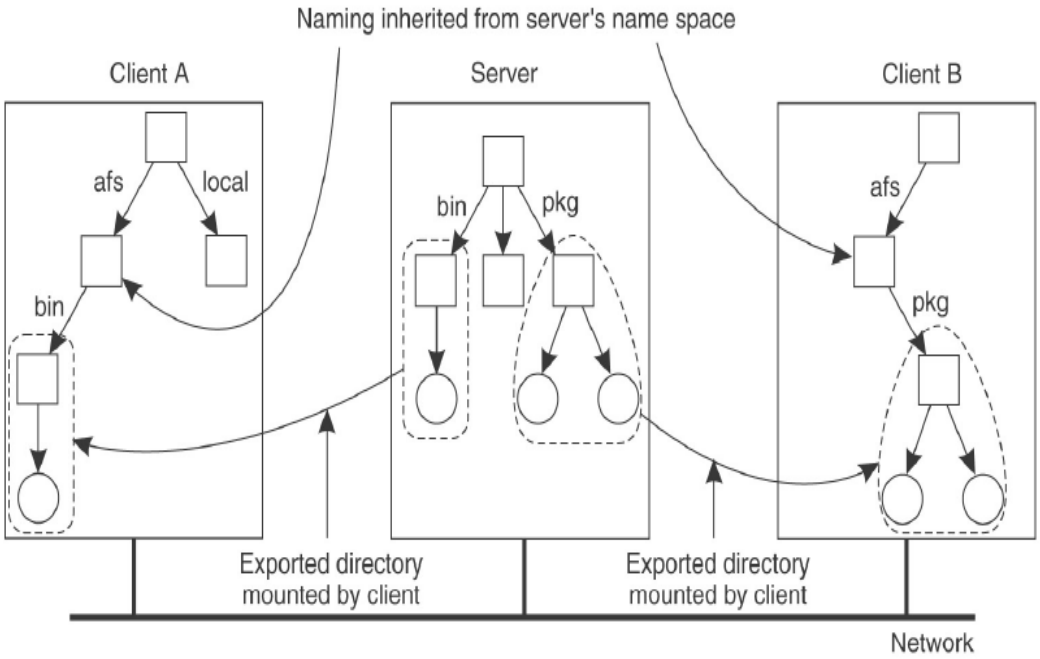

Figura 3 - Espaço de nomes compartilhado oferecido pelo Coda File System
Fonte: Tanenbaum, 2002, p. 610

Segundo BRAAM (1994) no *Coda File System*, os clientes têm seus volumes do sistema de arquivos distribuído montado abaixo do diretório /coda, similar ao seu antecessor. Qualquer arquivo compartilhado por algum servidor do *Coda File System* estará disponível neste ponto de montagem para todos os clientes. Diferente do NFS, os usuários se conectam ao serviço fornecido pelo *Coda File System* e não individualmente nos servidores, que são adicionados como membros do serviço de forma transparente.

4.4.5 CACHE E REPLICAÇÃO – *CODA FILE SYSTEM*

O Cache e a replicação são importantes implementações do *Coda File System*, sendo elas fundamentais para o suporte a alta disponibilidade da qual a ferramenta se propõe.

Segundo TANENBAUM (2002) quando um arquivo é aberto pelo cliente do *Coda File System* para leitura ou escrita, este é transferido e armazenado integralmente no computador local com o intuito de aumentar o grau de tolerância a falhas fazendo com que o cliente não fique dependente da disponibilidade do servidor. Quando o processo *Vice* fornece uma cópia de um arquivo para um processo *Venus*, ele também fornece uma promessa de *callback*, ou seja, um token emitido pelo servidor garantindo que ele notificará o processo cliente quando qualquer outro usuário modificar o arquivo. Quando um servidor realiza uma requisição de atualização de um arquivo, ele notifica todos os processos *Venus* para os quais tem promessas de callback emitidas, enviando um *callback* (uma chamada de procedimento remoto de um servidor para um processo Venus) notificando-os que o arquivo armazenado em *cache* está desatualizado.

Caso o processo Venus receba uma operação para abrir determinado arquivo, ele verificará primeiramente sua existência em *cache* e caso seja encontrado, ele verificará se não recebeu nenhum callback de notificação quanto a sua desatualização.

Caso o arquivo que esta armazenado em *cache* esteja desatualizado, então uma nova cópia é resgatada do servidor, mas caso o arquivo não tenha sofrido alterações, a cópia armazenada em *cache* é aberta e utilizada sem referencia ao servidor.

Segundo BRAAM (1994) no *Coda File System*, os volumes podem ser replicados entre vários servidores, diferente de seu antecessor. Cada volume é associado a um grupo denominado de Volume Storage Group (VSG), que consiste em um conjunto de servidores que armazenam as réplicas do volume. O conjunto de servidores acessíveis de um certo grupo em um certo momento é chamado de Accessible Volume Storage Group (AVSG).

Quando um cliente necessita acessar um arquivo, ele contata um membro do AVSG que contenha em seu volume o arquivo solicitado. Caso este tenha sofrido alterações, após seu fechamento pelo cliente, o arquivo é transferido em paralelo para todos os servidores do AVSG e posteriormente para os demais servidores do VSG.

KON (1994) afirma que quando um servidor reintegra a rede após uma falha, nenhuma providência é tomada de imediato para a atualização dos arquivos. Os arquivos somente são atualizados neste servidor caso seja realizada uma requisição de acesso a uma versão mais recente. Se o cliente descobre que o seu servidor está com uma versão antiga do arquivo, o cliente solicita a versão mais recente de quem a possuir e emite uma mensagem ao AVSG informando da existência de uma versão desatualizada.

4.4.6 TOLERÂNCIA A FALHAS – *CODA FILE SYSTEM*

Segundo TANENBAUM (2002) a arquitetura do *Coda File System* foi projetada para proporcionar que o sistema não seja sensível a partições na rede, garantindo ao usuário uma alta disponibilidade do serviço. Este recurso é alcançado com a implementação de *cache* no cliente, possibilitando que o usuário manipule arquivos mesmo com a presença de falhas ou instabilidades na rede, e com a implementação da replicação de volumes em diversos servidores, garantindo uma maior disponibilidade dos computadores fornecedores do serviço.

4.4.7 *NETWORK FILE SYSTEM*

O *Network File System* (NFS) foi originalmente desenvolvido pela Sun para ser utilizado em ambientes Unix, mas atualmente é possível encontrar implementações para quase todas as arquiteturas de computadores e sistemas operacionais. Segundo TANENBAUM (2002) a idéia principal do NFS é possibilitar que um conjunto de clientes e servidores possam compartilhar um sistema de

arquivos em comum, permitindo que um nodo seja ao mesmo tempo um cliente acessando arquivos remotos ou um servidor exportando arquivos.

Segundo COULOURIS (2005) o NFS é uma coleção de protocolos que provê ao cliente a visão de um sistema de arquivos distribuído, similar ao funcionamento do CORBA. Nesta seção, são apresentados diversos aspectos do NFS, concentrando-se nas versões 3 (especificado na RFC 1813) e 4 (especificado na RFC 3530).

4.4.8 ARQUITETURA - *NETWORK FILE SYSTEM*

Nos sistemas operacionais, as chamadas de sistema são utilizadas para fazer a *interface* de comunicação entre o cliente e o sistema de arquivos local. Segundo TANENBAUM (2002) nos sistemas operacionais baseados em Unix, esta *interface* é substituída por outra *interface* denominada de Virtual File System (VFS), que realiza a comunicação e oculta as diferenças entre os diversos sistemas de arquivos. As operações encaminhadas à *interface* VFS são direcionadas ao sistema de arquivos local ou algum componente previamente instalado, como o cliente do NFS, que é responsável pelo acesso aos arquivos armazenados nos servidores através das chamadas de procedimentos remotos.

A Figura 4 ilustra a arquitetura do NFS.

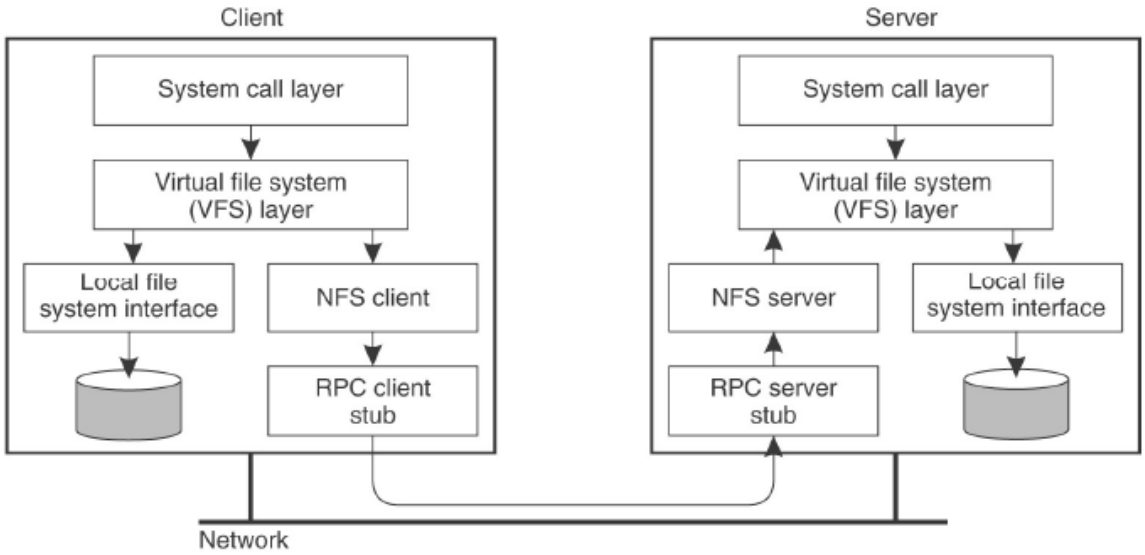

Figura 4 - Arquitetura do Network File System em ambientes Unix
Fonte: Tanenbaum, 2002, p. 578

Quando o cliente efetua uma requisição ao serviço de arquivos, esta solicitação é encaminhada para a *interface* do VFS, que de acordo com a localização física do arquivo solicitado, remete a requisição para o sistema de arquivos local ou para um componente.

Conforme TANENBAUM (2002), o VFS mantém uma tabela para cada sistema de arquivos montado, com uma entrada para cada um dos arquivos abertos, denominada de *v-node*, que é utilizado para informar se o arquivo é local ou remoto. Caso o arquivo esteja armazenado localmente, o *v-node* possuirá uma referência para o *i-node* do arquivo local e caso o arquivo esteja armazenado remotamente, ele conterá o manipulador do arquivo que fornece todas as informações necessárias para acessá-lo.

Segundo KON (1996) uma importante vantagem na arquitetura do NFS é quanto a independência do sistema de arquivos local, possibilitando que sejam compartilhados arquivos em ambientes heterogêneos.

4.4.9 COMUNICAÇÃO - *NETWORK FILE SYSTEM*

Uma das características importantes na arquitetura do NFS é sua independência quanto a sistema operacional, rede e protocolo de comunicação, possibilitando que, por exemplo, clientes Windows acessem servidores Unix. Essa independência é atingida devido ao fato do NFS possuir em sua arquitetura uma camada de RPC que oculta as diferenças citadas, sendo esta responsável pela comunicação entre clientes e servidores.

Uma RPC é similar a uma invocação a método remoto, pois um processo cliente chama um procedimento que está sendo executado em um processo servidor, sendo que os servidores podem ser clientes de outros servidores para permitir encadeamentos de RPCs. Um processo servidor define em sua *interface* de serviço os procedimentos que estão disponíveis para serem chamados de forma remota. Todas as operações da versão 4 do NFS são implementadas com uma única chamada composta de procedimento remoto a um servidor de arquivos, ou seja, uma mesma chamada RPC para a leitura de um arquivo, por exemplo, pode conter uma operação complexa envolvendo bloqueio, abertura, leitura etc., a exemplo da Figura 5 (b), diferente da versão 3, onde este processo era resolvido com duas chamadas de procedimento remoto, ilustrado na Figura 5 (a) (TANENBAUM, 2002, p.582).

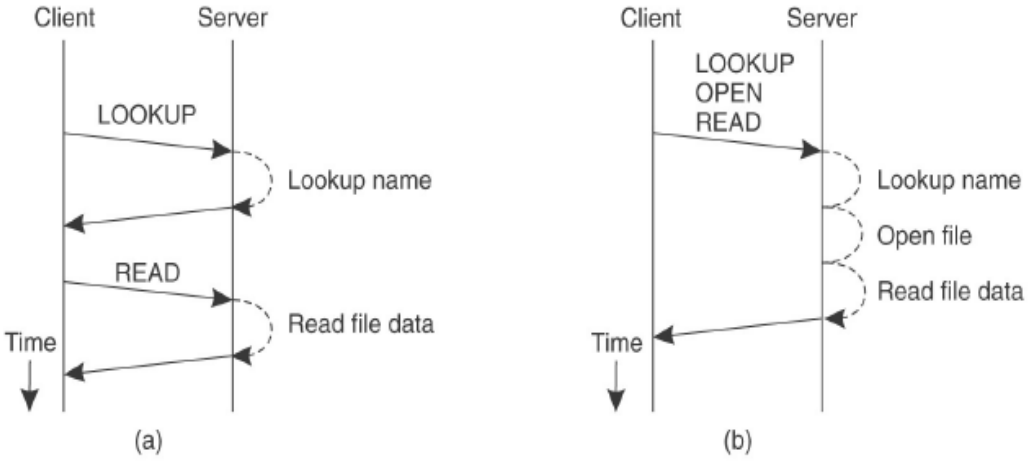

Figura 5 - Processo de leitura de um arquivo no NFSv3 (a) e no NFSv4 (b)
Fonte: Tanenbaum, 2002, p. 582

4.4.10 PROCESSOS - *NETWORK FILE SYSTEM*

Segundo KON (1996) o NFS é um sistema cliente-servidor no qual usuários solicitam ao servidor que realize operações em determinado arquivo. TANENBAUM (2002) complementa que uma de suas grandes características, que foi abandonado na versão 4, é o fato do servidor não guardar o estado das transações realizados.

Esta característica traz algumas vantagens imediatas, sendo que a mais significativa é que, quando ocorre uma falha no servidor, ele não perde nenhuma informação sobre o estado da comunicação com os clientes, não precisando, desta forma, gastar tempo na tentativa de reconstituir este estado quando ele se recuperar da falha. Para o cliente, tudo que ele necessita fazer quando a comunicação com o servidor falhar é repetir as chamadas ao serviço até que obtenha resposta. Entretanto, esta característica impede o servidor de gerenciar o acesso concorrente de seus arquivos e desta forma, o NFS não assume a consistência de seu sistema de arquivos, podendo haver casos em que, diferentes clientes possuam diferentes cópias do mesmo arquivo ou diretório no *cache* de seus computadores.

Em sua versão 4, o NFS guarda o estado das transações realizadas, mantendo uma tabela de todos os arquivos abertos pelos usuários, assemelhando-se a um sistema de arquivos centralizado (TANENBAUM, 2002, p.584).

4.4.11 SERVIÇO DE NOMES - *NETWORK FILE SYSTEM*

Assim como em todos os sistemas de arquivos distribuídos, o serviço de nomes é uma característica importante do NFS. A idéia principal do modelo de serviço de nomes do NFS é prover aos usuários um acesso transparente ao sistema de arquivos dos servidores, que é alcançada quando o cliente monta em seu sistema de arquivos local o sistema de arquivos distribuído, conforme apresentado na Figura 6. Esta ilustração demonstra que os usuários não compartilham o mesmo espaço de nomes, pois em ambos os clientes, o mesmo diretório do servidor esta compartilhado em pontos de montagem diferentes. A desvantagem deste modelo reflete no processo de comunicação entre os clientes, que terão dificuldades em referenciar um mesmo arquivo, uma vez que o caminho de acesso a ele será diferente entre os usuários (TANENBAUM, 2002, p.583).

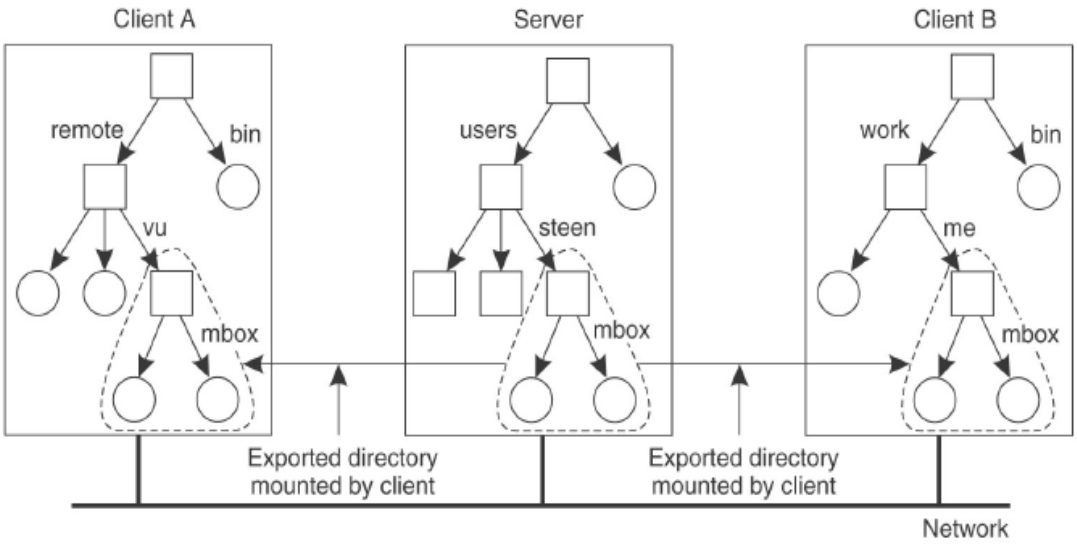

Figura 6 - Montagem de um sistema de arquivos distribuído no NFS
Fonte: Tanenbaum, 2002, p. 583

4.4.12 CACHE E REPLICAÇÃO - *NETWORK FILE SYSTEM*

O módulo cliente do NFS armazena em *cache* os resultados obtidos em determinadas operações para reduzir o número de requisições feitas aos servidores.

O uso do *cache* no cliente implica na possibilidade de existirem diversas versões de arquivos, ou porções deles, em diferentes nós clientes, pois as gravações feitas por um cliente não resultam na atualização imediata do arquivo em outros computadores. Em vez disso, os clientes são responsáveis por fazerem uma consulta seqüencial no servidor para verificar se os dados armazenados em *cache* que eles contêm são atuais. A Figura 7 ilustra a integração do *cache* do computador local com o módulo cliente do NFS. Cada cliente pode possuir *cache* em memória ou no disco rígido que contém informações recentemente acessadas do servidor, tais como o conteúdo de um arquivo, seus atributos e diretórios (TANENBAUM, 2002, p.583)

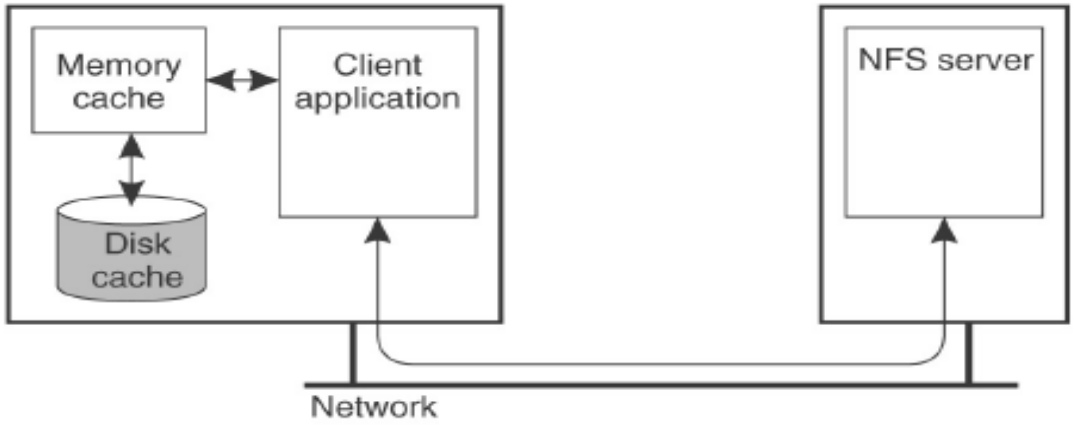

Figura 7 - Representação do cache ao lado do cliente no Network File System
Fonte: Tanenbaum, 2002, p. 595

A versão 4 do NFS disponibiliza um suporte ainda limitado a replicação de arquivos, permitindo que os dados sejam copiados entre diversos servidores, sendo que estas cópias estarão disponíveis somente para consulta dos usuários. Quando o cliente estabelecer acesso ao sistema de arquivos remoto do servidor, ele obterá a relação dos locais que disponibilizam uma réplica do ponto de montagem e caso ocorram falhas de comunicação, o cliente tentará estabelecer uma nova conexão com os servidores informados na lista (TANENBAUM, 2002, p.595).

4.4.13 TOLERÂNCIA A FALHAS - *NETWORK FILE SYSTEM*

Até a versão 4 do NFS, a tolerância a falhas não era um assunto muito mencionado, uma vez que seu protocolo não exigia que o servidor guardasse informações sobre o estado da comunicação com os seus clientes para oferecer o seu serviço satisfatoriamente, entretanto, com a mudança de arquitetura entre a versão 3 e 4 do NFS, a tolerância e a recuperação de falhas passam a ser preocupações desta ferramenta.

Um dos problemas existentes com o RPC diz respeito ao fato dele não proporcionar nenhuma garantia em relação a sua confiabilidade, sendo que o principal problema do NFS é a falta de uma implementação para detecção de solicitações duplicadas. Entretanto, essas situações são amenizados com o uso de

um *cache* implementado no servidor que gerencia as requisições (que possuem em seu cabeçalho um código identificador único) vindas do cliente.

A Figura 8 ilustra esta implementação com a exemplificação da negociação de mensagens entre o cliente e o servidor. Na primeira situação, apresentada na Figura 8 (a), o cliente envia uma requisição e inicia um *timer* que, se o cronômetro encerrar antes da resposta chegar, o cliente retransmite a mensagem original com seu código identificador original, que será ignorada pelo servidor, que ainda não havia concluído o processamento da primeira requisição.

No segundo caso, apresentado na Figura 8 (b), o servidor recebe uma requisição retransmitida logo após ter enviado o processamento da solicitação original ao cliente, sendo esta nova requisição ignorada, uma vez que seus códigos identificadores são idênticos e a resposta à mensagem original foi encaminhada (TANENBAUM, 2002, p.598).

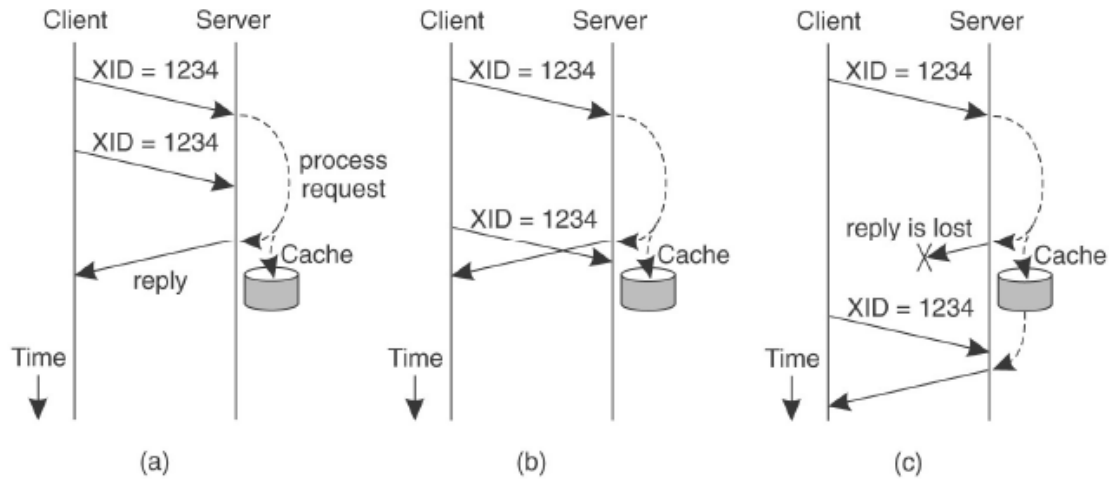

Figura 8 - Exemplo de situações para controle de mensagens retransmitidas
Fonte: Tanenbaum, 2002, p. 598

Na terceira situação, apresentada na Figura 8 (c), a resposta enviada ao cliente se perdeu, sendo então retransmitido o conteúdo existente em *cache*, evitando que o servidor tivesse que processar o pedido novamente (TANENBAUM, 2002, p.598).

4.5 NOVAS TECNOLOGIAS EM SISTEMAS DE ARMAZENAMENTO DISTRIBUÍDOS

As ferramentas apresentadas a seguir nesta seção consistem em soluções de armazenamento distribuído criadas e mantidas por empresas renomadas, como Google, IBM e Microsoft, apoiadas por pesquisas de várias universidades.

4.5.1 *ANDREW FILE SYSTEM*

O *Andrew File System* é um ambiente de computação distribuída desenvolvido na Universidade de Carnegie Mellon com o apoio da IBM para uso como sistema de computação e informação do campus. O projeto, cuja arquitetura é similar ao *Coda File System*, citado na Seção 4.4.1, reflete a intenção de suportar o compartilhamento de informações em larga escala, minimizando a comunicação entre cliente e servidor. Esta funcionalidade foi alcançada através da adoção da semântica de sessão e pela transferência de arquivos inteiros entre computadores clientes e servidores, armazenando-os em *cache* nos computadores locais até que o servidor receba uma versão mais atualizada.

A fim de limitar a possível falta de segurança no sistema, o AFS adota uma série de mecanismos como, por exemplo, o *Kerberos Authentication Server*, que permite a autenticação mútua de servidores e clientes.

Inicialmente, o projeto foi implementado na Universidade de Carnegie Mellon em uma rede com estações de trabalho e servidores executando Unix BSD e o sistema operacional Macintosh e, subseqüentemente, se tornou disponível em versões comerciais e de domínio público.

4.5.2 DFS – *DISTRIBUTED FILE SYSTEM*

Segundo informações contidas no *site* da Microsoft, o DFS permite que os administradores agrupem pastas compartilhadas localizadas em diferentes servidores, conectando-os de forma transparente a um ou mais espaços de nomes DFS. Um espaço de nomes DFS é uma exibição virtual de pastas compartilhadas em uma organização.

Usando as ferramentas do DFS, um administrador pode selecionar que pastas compartilhadas deverão ser apresentadas no espaço de nomes aos usuários do sistema, projeta a hierarquia na qual as pastas aparecerão e determina os nomes que as pastas compartilhadas exibirão no espaço de nomes.

Quando um usuário visualiza o espaço de nomes, as pastas parecem residir em um único disco rígido, transmitindo assim uma visão unificada do sistema, desta forma os usuários podem navegar pelas pastas no espaço de nomes sem que precisem saber os nomes dos servidores ou das pastas compartilhadas que hospedam os dados.

Segundo a Microsoft (2012) o DFS também oferece muitos outros benefícios, incluindo tolerância a falhas e a capacidade de compartilhamento de carga, recursos que o tornam ideal para organizações que necessitam de um alto nível de disponibilidade a seus arquivos de dados.

4.5.3 *FARSITE*

Segundo Adya (2002) o *Farsite* é um sistema de arquivos distribuído que vem sendo desenvolvido nos laboratórios de pesquisa da Microsoft. Ele disponibiliza aos usuários a visão de um sistema de arquivos centralizado enquanto que sua estrutura física encontra-se dispersa na rede em servidores e estações de trabalho. O *Farsite* fornece os mesmos recursos de um sistema de arquivos centralizado (armazenamento confiável, espaço de nomes compartilhado, transparência de acesso e transparência de localização) e os benefícios de um sistema de arquivos local (baixo custo e segurança no acesso aos arquivos).

Em termos de arquitetura, o *Farsite* agrupa os computadores no que ele denomina de grupos de diretórios. Todos os integrantes dos grupos de diretórios têm como função armazenar réplicas dos arquivos que o grupo disponibiliza a rede, atender requisições de forma determinística, atualizar as réplicas e enviar respostas aos usuários.

Quando o cliente solicita acesso a um arquivo, ele envia uma mensagem a um determinado grupo de diretórios que responde com o conteúdo do arquivo pedido. Caso o usuário atualize esse arquivo, ele envia a atualização ao grupo, que internamente fará o trabalho de replicação.

O *Farsite* possibilita que os grupos de diretórios deleguem porções de arquivos aos demais grupos da rede, possibilitando um balanceamento de carga e aumentando a performance do serviço. Além desses recursos, o *Farsite* implementa suporte a *cache* no cliente para o aumento de performance e a criptografia de arquivos armazenados na rede como meio de segurança.

4.5.4 *GOOGLE FILE SYSTEM*

Segundo Ghemawat (2003) o *Google File System* é um sistema de arquivos distribuído desenvolvido e estendido extensamente dentro da empresa *Google* como plataforma de armazenamento, onde o maior agrupamento de servidores existente para dados provê centenas de Terabytes através de milhares de discos rígidos em mais de mil máquinas que são acessadas simultaneamente por centenas de clientes. O sistema compartilha de muitos dos mesmos objetivos dos demais sistemas de arquivos distribuídos, tais como escalabilidade, desempenho, confiança e disponibilidade, que foram implementados levando em consideração as necessidades da empresa e o ambiente tecnológico utilizado por ela.

Sua arquitetura é composta por um único servidor principal e diversos computadores que armazenam os arquivos acessados pelos usuários. Assim como outras soluções de armazenamento distribuído, o *Google File System* possui um relacionamento cliente-servidor simétrico, sendo possível que uma estação de trabalho atue na rede tanto como cliente, quanto como servidor.

Quando um cliente necessita de um arquivo, ele entra em contato com o servidor principal que o orienta informando em qual computador está localizado o arquivo solicitado. A partir dai, todas as alterações realizadas pelo cliente no arquivo solicitado são encaminhadas diretamente ao computador indicado pelo servidor principal. Além deste apontamento, o servidor principal é responsável por gerenciar a replicação dos arquivos e coordenar os processos de inclusão e remoção de computadores na rede.

5 METODOLOGIA

Para alcançar os resultados esperados neste trabalho de conclusão de curso, foi utilizada a Pesquisa Experimental. Na primeira parte deste trabalho foram pesquisadas as bibliografias referentes ao tema, com o intuito de prover o referencial teórico necessário para o embasamento da pesquisa. Na segunda parte foram realizados os experimentos propostos a seguir com as ferramentas de sistemas distribuídos do Microsoft Windows Server e do Linux.

Segundo Gil (2002), de modo geral, o experimento representa o melhor exemplo de pesquisa científica. Essencialmente, a pesquisa experimental consiste em determinar um objeto de estudo, selecionar as variáveis que seriam capazes de influenciá-lo, definir as formas de controle e de observação dos efeitos que a variável produz no objeto.

Os experimentos identificaram o grau de transparência das ferramentas de sistemas distribuídos classificando-os em alto e baixo, para isto dois cenários distintos foram elaborados. O primeiro cenário experimentou a transparência de acesso e o segundo a transparência de localização e replicação do DFS em cada plataforma. Os cenários propostos neste experimento foram repetidos em ambas as plataformas testadas, onde o critério para indicar o grau de transparência alto foi a obtenção do sucesso no respectivo cenário e o critério para indicar o grau de transparência baixo foi consequentemente a não obtenção do sucesso no respectivo cenário.

5.1 PLANEJAMENTO DO CENÁRIO DE TESTES

Segundo os conceitos de sistemas de arquivos distribuídos estudados no capítulo 4, um sistema de arquivos distribuídos transparente deve ser similar às redes de energia elétrica, onde toda a complexidade da geração, transmissão e distribuição da energia é oculta para o usuário, que pode apenas usufruir deste recurso para ligar seus equipamentos eletrônicos sem precisar se preocupar de onde veio ou como a energia está chegando até ele, desta mesma forma, o usuário

do sistema não deve se preocupar de onde está vindo e para onde são enviadas as informações para armazenamento.

Tendo em mente esta parábola, foi criado um cenário para simular um ambiente onde os usuários dispõem de serviços de armazenamento distribuído de arquivos, de modo que através da estação cliente estes usuários podem gerenciar seus arquivos distribuídos de forma transparente, sem se preocupar com a localização física destes arquivos e tampouco com a utilização de *softwares* específicos para a manipulação destes, cumprindo assim o propósito de transmitir a impressão de que todos os arquivos acessados pelos usuários estão armazenados localmente em suas estações de trabalho.

Foram utilizadas cinco máquinas virtuais para a aplicação dos experimentos, sendo que duas maquinas virtuais atuaram como servidores para a realização dos experimentos com a plataforma Windows, outras duas também atuaram como servidores para a plataforma Linux. A quinta e última máquina virtual atuou como Cliente, rodando o Microsoft Windows. Isto é possível graças a utilização do Samba do Linux, que permite a comunicação entre estações e servidores que executam sistemas operacionais distintos. O fato de utilizar o mesmo sistema operacional para o cliente em ambas as plataformas possibilita colher os dados deste experimento de maneira homogênea, isolando assim características intrínsecas às diferentes plataformas testadas.

O uso da tecnologia de virtualização adotada neste experimento se mostrou bastante eficiente por oferecer benefícios como, por exemplo, a economia com equipamentos de *Hardware*, espaço físico e consumo de energia elétrica, além disto, a virtualização foi capaz de emular com sucesso todas as características desejadas para a realização dos experimentos planejados.

Como o foco dos experimentos propostos é apenas medir a transparência das ferramentas, a configuração de *Hardware* dos computadores utilizados ficará em segundo plano. Apenas foram observadas as configurações mínimas necessárias para suportar a correta execução das máquinas virtuais. Para ambas as plataformas foram usadas as mesmas configurações, sendo que os experimentos foram todos

realizados em um único computador capaz de executar satisfatoriamente as máquinas virtuais de ambas as plataformas simultaneamente.

5.2 CENÁRIO DE TESTES DA TRANSPARÊNCIA DE ACESSO

A transparência de acesso, como já citado na seção 4.3.8, possibilita que os recursos remotos de um sistema de arquivos distribuídos sejam acessados de forma semelhante aos recursos do sistema local. Desta forma, para identificar o grau da transparência de acesso existente nas ferramentas neste cenário, foi observado como os usuários do sistema obtêm acesso aos recursos presentes no sistema de arquivos distribuídos.

A condição esperada para a obtenção do sucesso neste cenário é a mesma para ambas as plataformas, pelo fato de que o cliente executa o Microsoft Windows XP. O cliente deve obter acesso ao DFS através de um ponto de montagem similar a um diretório local que exibe seus recursos sob a forma de diretórios, pastas e arquivos. Desta forma o sistema recebe a nota “Alta” para a transparência de acesso.

A não obtenção de sucesso neste cenário ocorre caso os usuários necessitem utilizar qualquer tipo de aplicativo ou ferramenta de terceiros para obter acesso aos seus arquivos armazenados dentro do sistema de arquivos distribuído, como por exemplo, um browser ou outro aplicativo que não faça parte do sistema operacional. Em outras palavras, se existir qualquer característica que defira da forma tradicional do sistema explorar pastas e diretórios locais, no caso, pelo fato de que o cliente executará o Microsoft Windows XP, qualquer forma de acesso que não ocorra através do Windows Explorer definirá o cenário como mal sucedido. Desta forma o sistema recebe a nota “Baixa” para a transparência de acesso.

5.3 CENÁRIO DE TESTES DA TRANSPARÊNCIA DE REPLICAÇÃO E LOCALIZAÇÃO

Para realizar os testes com a transparência de replicação e de localização, foi montado um cenário onde ocorreu a cópia de um arquivo qualquer para o local

criado no computador do cliente que concede acesso ao ponto de montagem do sistema de arquivos distribuídos e, em seguida, foi simulada a falha do servidor primário. No entanto para garantir o sucesso do experimento descrito neste cenário, a falha do servidor primário foi simulada somente após o servidor secundário receber a cópia replicada do arquivo do usuário.

Esta condição tornou-se necessária pelo fato de que existe uma latência imprevisível para que o processo de replicação ocorra entre os servidores. Esta latência mostrou-se relativa a inúmeros fatores relacionados ao processo de replicação, como por exemplo, a velocidade de transferência da rede de dados onde a replicação ocorre, o tamanho do arquivo a ser replicado, a capacidade do *Hardware* dos computadores envolvidos e até mesmo a carga aferida, tanto nos servidores, clientes e até mesmo na rede de dados.

O intuito deste cenário é identificar a transparência com que o processo de replicação ocorre para o usuário final do sistema, por isto, mesmo que desejável, o fator tempo foi desconsiderado, visto que para medi-lo seria preciso controlar todas as variáveis descritas acima capazes de influenciá-lo.

A obtenção de sucesso neste cenário ocorre caso o cliente continue acessando seus arquivos através do ponto de montagem para o qual ele foi copiado anteriormente, mesmo após a falha do servidor primário. Desta forma o sistema recebe a nota “Alta” para a transparência de replicação.

A não obtenção de sucesso neste cenário ocorre caso os usuários percebam de alguma forma que o diretório ou o arquivo presente no sistema de arquivos distribuídos tornou-se indisponível após a falha do servidor primário. Os usuários também não devem ter conhecimento de que o seu acesso foi direcionado a um novo servidor. Desta forma o sistema recebe a nota “Baixa” para a transparência de replicação.

Se bem sucedido, o ambiente proposto neste cenário, além de oferecer a transparência de replicação, também disponibilizará aos clientes um serviço de armazenamento distribuído tolerante a falhas, isto significa que o cliente não precisa se preocupar com a localização física entre os volumes dos servidores e dos arquivos que ele manipula o que indica que caso o servidor acessado falhe, suas

requisições serão encaminhadas ao segundo servidor ativo, sem o conhecimento do cliente que continuará a usufruir dos serviços, desta vez localizados em outro servidor.

Por este fato, pode-se afirmar que, dependendo do resultado obtido com os experimentos da transparência de replicação, a transparência de localização também estará presente na plataforma, caso o experimento com a transparência de replicação seja bem sucedido. Desta forma o sistema recebe também a nota “Alta” para a transparência de localização.

Caso o resultado do experimento proposto neste cenário para a identificação da transparência de replicação seja mal sucedido, pode-se afirmar que a transparência de localização também não estará presente, desta forma o sistema recebe também a nota “Baixa” para a transparência de localização.

6 RESULTADOS

Neste capitulo são relatados todos os detalhes relevantes a criação dos cenários e a realização dos experimentos propostos no capítulo anterior. Com o intuito de promover um entendimento claro para as comparações realizadas em cada plataforma, foram montadas duas tabelas expositivas para mostrar os resultados obtidos dos testes para ambas as plataformas testadas.

6.1 APLICAÇÃO DOS EXPERIMENTOS NA PLATAFORMA WINDOWS

Para que a aplicação dos experimentos pudesse ocorrer, primeiramente foi preciso criar as maquinas virtuais que receberiam a instalação dos sistemas. As duas primeiras máquinas receberam a instalação do Microsoft Windows Server 2008 R2 e a terceira foi configurada como cliente, recebendo a instalação do Microsoft Windows XP.

Para a correta execução dos sistemas instalados, foi preciso também verificar a configuração mínima necessária para executar os sistemas satisfatoriamente. Os servidores foram instalados em maquinas virtuais com 1 gigabyte de memória RAM, 30 gigabytes de disco rígido e o cliente recebeu 256 megabytes de memória RAM e 15 gigabytes de disco. A figura 9 mostra as três maquinas virtuais em execução.

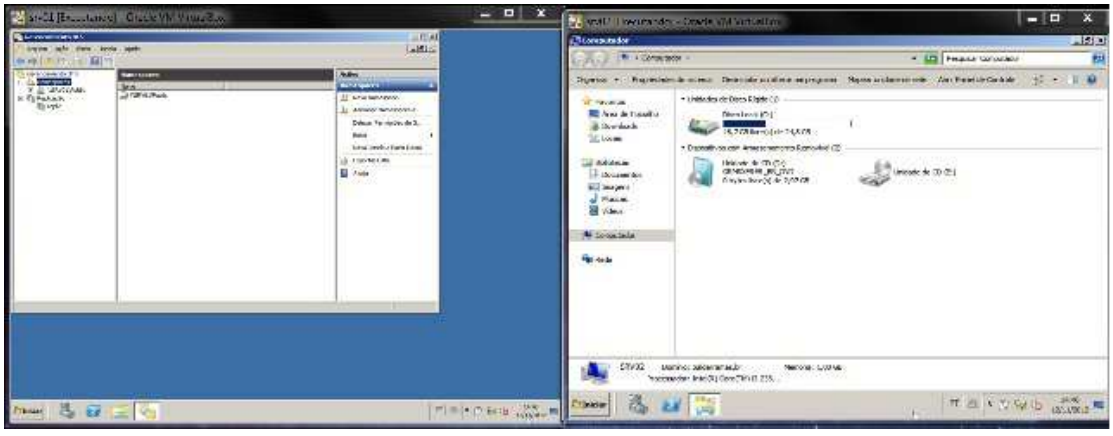

Figura 9 - Máquinas Virtuais dos servidores em execução.

Fonte: Própria

Não é necessário baixar e instalar qualquer programa de terceiros para que o DFS funcione no Windows Server, entretanto este recurso vem desabilitado por padrão, portanto após a instalação do Windows Server foi preciso habilitar o recurso “Serviços de Arquivos” em “Gerenciamento do Servidor” e marcar os módulos do DFS com os módulos para suporte a “Namespace” e “Replicação”. A Figura 10 mostra a tela de configuração “Serviços de Arquivo”.

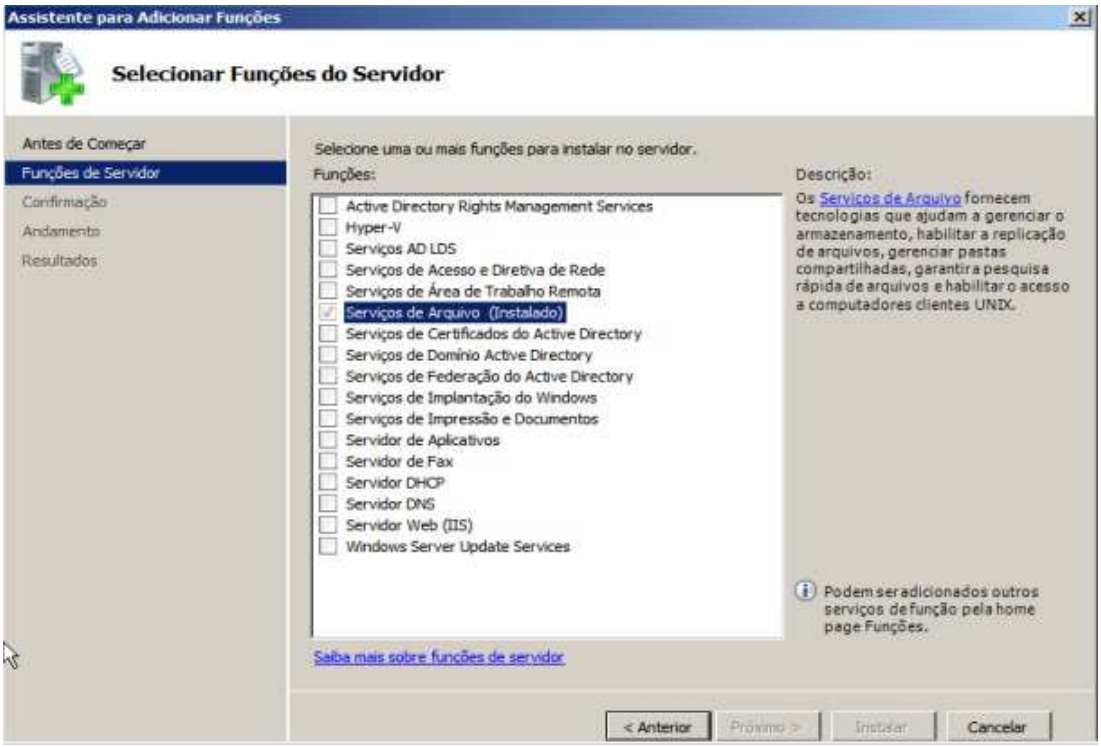

Figura 10 - Adicionando Serviço de Arquivos.

Fonte: Própria

Antes de configurar o DFS, outro ponto importante a ser observado é a configuração do Active Directory (AD), pois apesar de não ser obrigatória, sem ele não é possível obter a transparência de Localização no DFS. Isto porque quando configurado, o AD cria por padrão um nome de domínio, como “balderramas.br” por exemplo. Este domínio é usado para visualizar unificadamente as pastas replicadas disponíveis ao usuário do DFS, de modo que o endereço físico dos servidores e sua quantidade sejam ocultos aos usuários do sistema, ou seja, basta que o usuário acesse este domínio para que possa visualizar as pastas disponíveis a ele, incluindo as pastas do DFS. A Figura 11 mostra o acesso ao diretório do domínio criado.

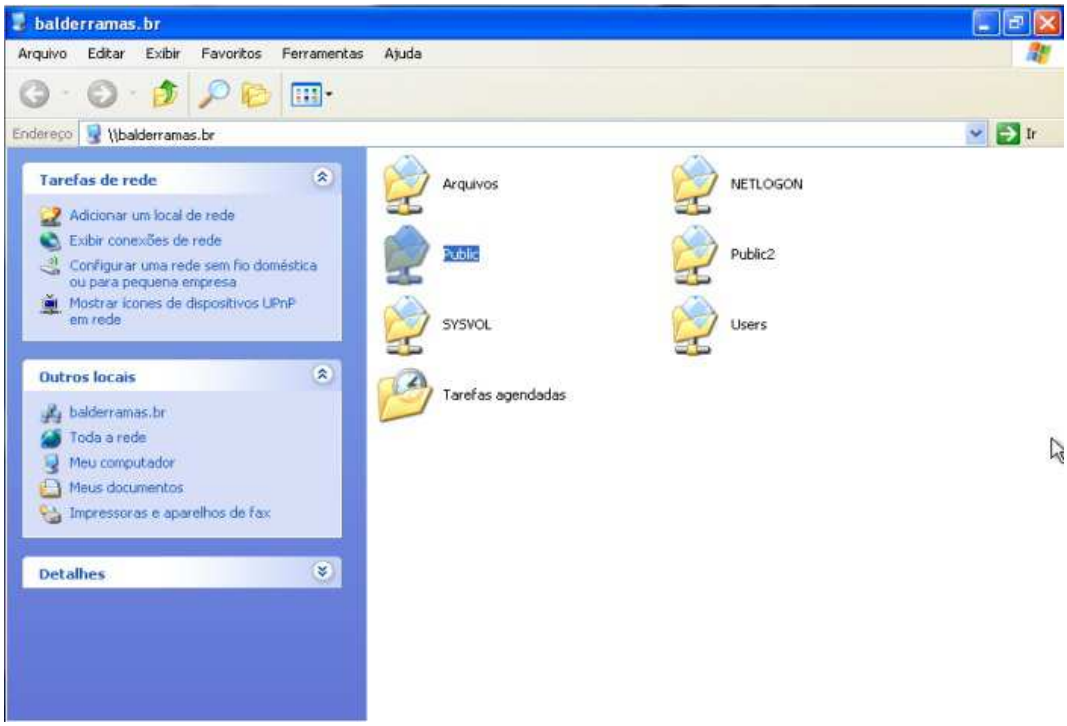

Figura 11 - Diretório raiz do Domínio.

Fonte: Própria

Sem o uso do AD o usuário precisaria acessar manualmente outro endereço para ser direcionado ao servidor secundário que possui as cópias provenientes da replicação caso o servidor primário por ventura torne-se indisponível.

O próximo passo então foi criar um novo Namespace do DFS dentro da guia “Gerenciamento DFS”. Este Namespace nada mais é do que um diretório virtual do DFS que concentrará todas as pastas e diretórios distribuídos por toda a rede. A Figura 12 mostra a tela de adição de pastas ao Namespace.

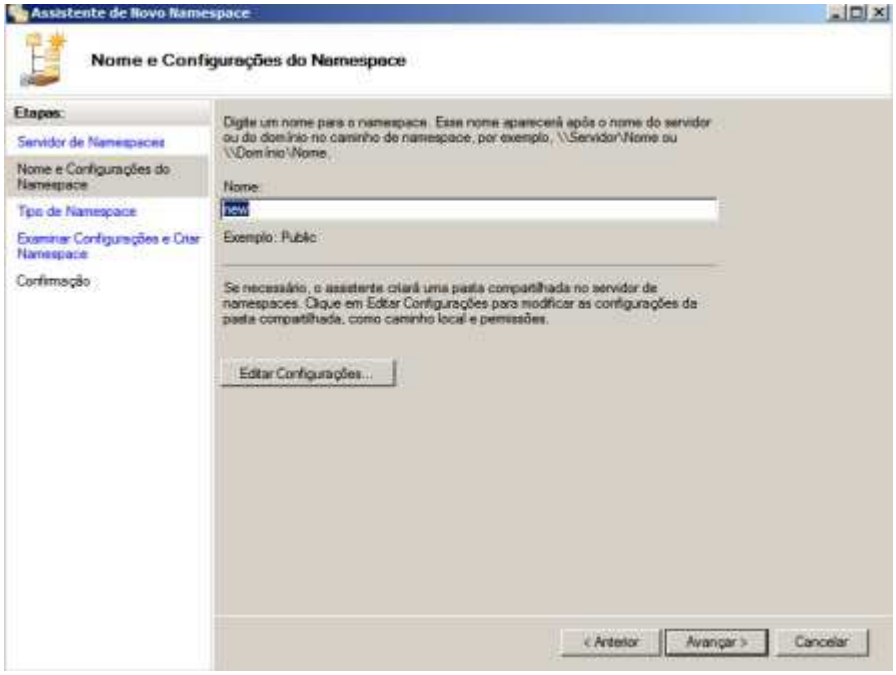

Figura 12 - Adicionando novo Namespace.

Fonte: Própria

Após sua configuração, quando um usuário acessar o domínio, ele visualizará um diretório com o nome atribuído ao Namespace, ao acessar este diretório são exibidas todas as pastas previamente adicionadas ao Namespace. A Figura 13 mostra o diretório raiz do domínio “balderramas.br” com o Namespace “new”.

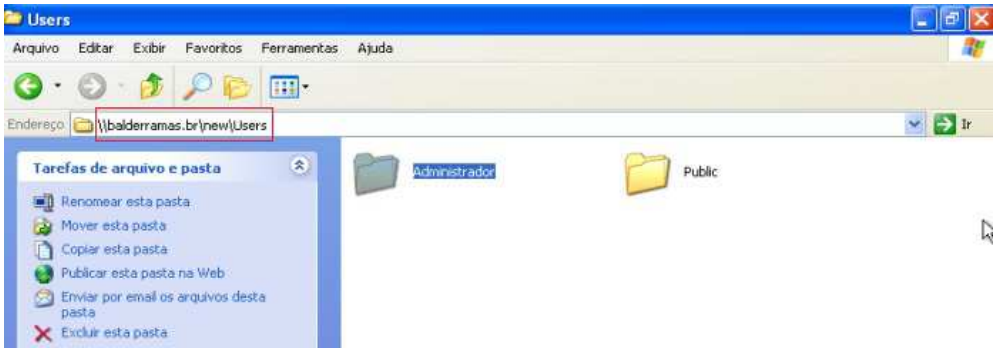

Figura 13 - Diretório do DFS.

Fonte: Própria

Após a configuração do Namespace foi configurado então um novo Grupo de Replicação do DFS para os diretórios do Namespace criados. Esta configuração também é realizada no “Gerenciamento DFS” através de um assistente.

Durante a configuração do novo grupo de replicação o Windows permite escolher diversas opções de configuração tais como os servidores membros do grupo (no mínimo dois servidores), a topologia com que será configurada a replicação, largura de banda utilizada, dias e horários específicos para a replicação, membro primário (que terá seus dados replicados para outros servidores) e por fim o diretório local que receberá a replicação física dos arquivos no membro secundário de replicação.

Após a configuração da replicação e das permissões de usuários todas as características necessárias para a realização dos testes com a plataforma Windows estão contempladas.

6.1.1 TRANSPARÊNCIA DE ACESSO NO MICROSOFT WINDOWS SERVER

Para que a Transparência de acesso possa ser testada o usuário precisa possuir acesso ao ponto de montagem do DFS, para isto utilizei o recurso de mapeamento de pasta, desta forma pode-se mapear um diretório remoto qualquer da rede para que seja criada uma unidade virtual no computador, assim o usuário pode acessar a unidade mapeada como se fosse uma nova unidade local. O diretório mapeado foi o diretório virtual criado anteriormente pelo Namespace, que concede acesso as pastas do DFS, desta forma o usuário terá a impressão de que os arquivos do DFS estão presentes em seu computador. As Figuras 14 e 15 mostram como ficou o mapeamento do diretório “new” no cliente, que concede acesso ao DFS.

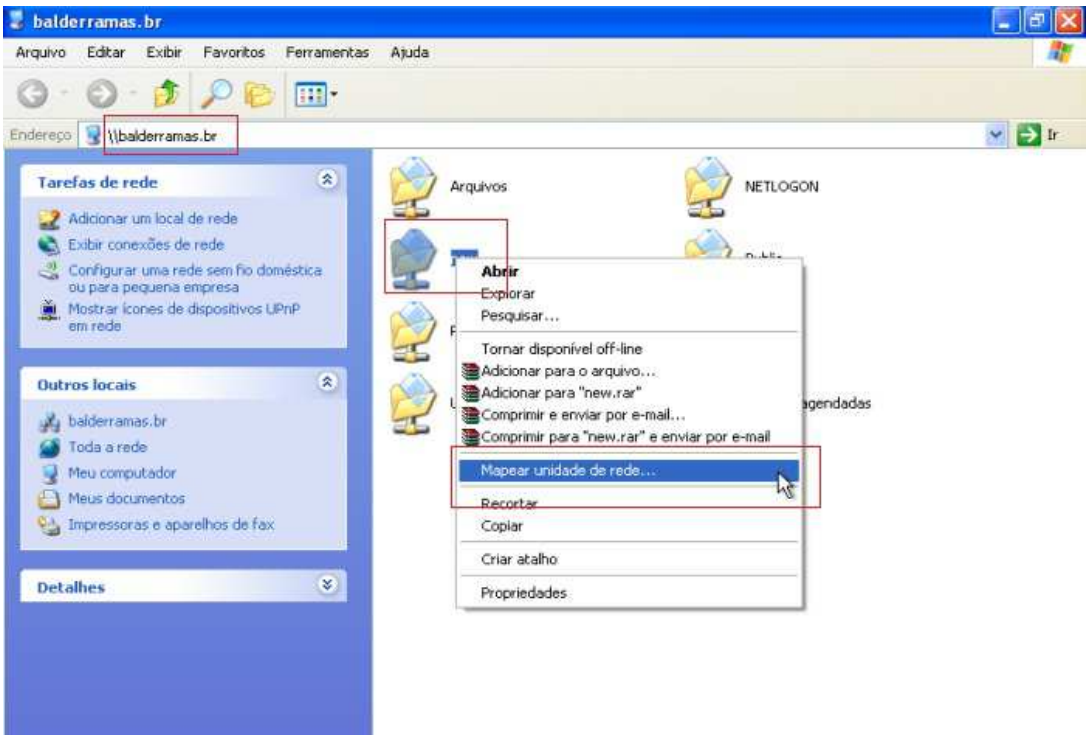

Figura 14 - Mapeando Unidade do DFS

Fonte: Própria

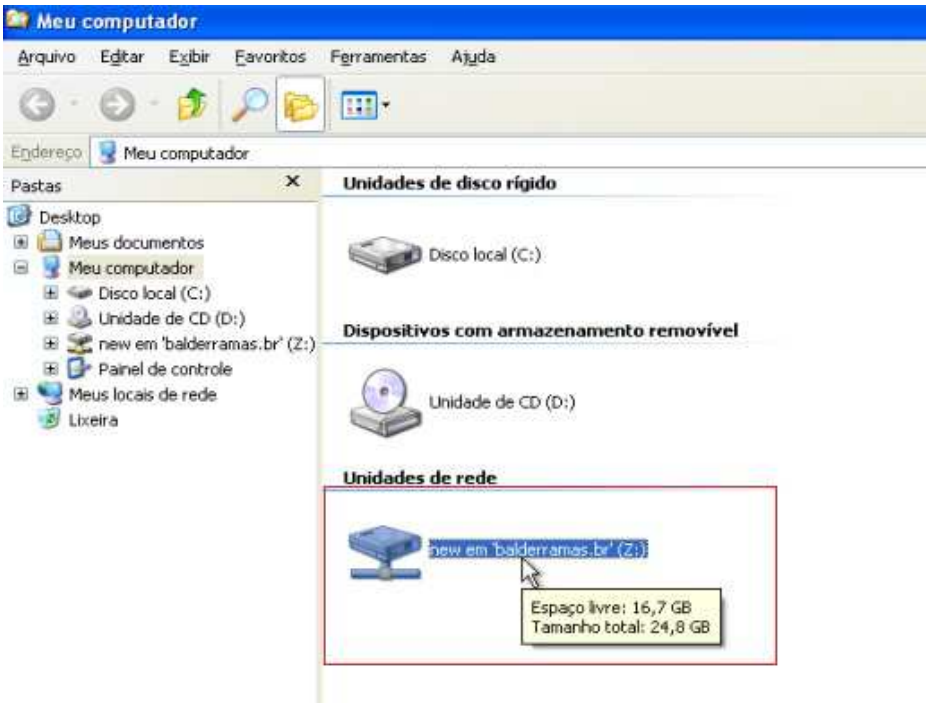

Figura 15 - Mapeamento Criado

Fonte: Própria

Existem também outras formas de criar este acesso aos usuários, se a rede abrigar muitos clientes é possível criar uma regra através do AD para montar automaticamente unidades aos usuários, escolhi mapear a unidade manualmente pois o experimento utilizará apenas um único cliente. Uma vez criado o mapeamento, pode-se utilizá-lo de várias formas, como por exemplo criar um atalho para ele na área de trabalho do cliente ou até apontar para o mapeamento toda a pasta “Meus Documentos” do cliente, esta escolha depende da necessidade específica de cada ambiente de trabalho.

Para a realização do experimento apenas criei um atalho para esta unidade mapeada na “área de trabalho” do usuário e o chamei de “Documentos”, conforme ilustrado na Figura 16, desta forma basta que o usuário abra o atalho criado para que obtenha acesso aos arquivos distribuídos do DFS.

Figura 16 - Atalho para unidade mapeada do DFS
Fonte: Própria

Ao abrir o atalho criado, o diretório do DFS é aberto perfeitamente igual a qualquer outra pasta ou diretório armazenado localmente no computador do usuário.

Lembrando que o objetivo deste cenário é verificar se o usuário é capaz de obter acesso aos arquivos distribuídos do DFS da mesma forma com que acessa seus arquivos locais, o experimento com este cenário foi considerado bem sucedido, portanto a plataforma Windows recebeu nota Alta para a Transparência de Acesso.

6.1.2 TRANSPARÊNCIA DE REPLICAÇÃO E LOCALIZAÇÃO NO MICROSOFT WINDOWS SERVER

Neste cenário foram testadas as transparências de Replicação e de Localização da plataforma Microsoft Windows Server 2008 R2, para isto foi preciso configurar e ativar a replicação do DFS nos servidores. A Figura 17 mostra como ficou a tabela de replicação contendo as pastas replicadas e seus respectivos destinos.

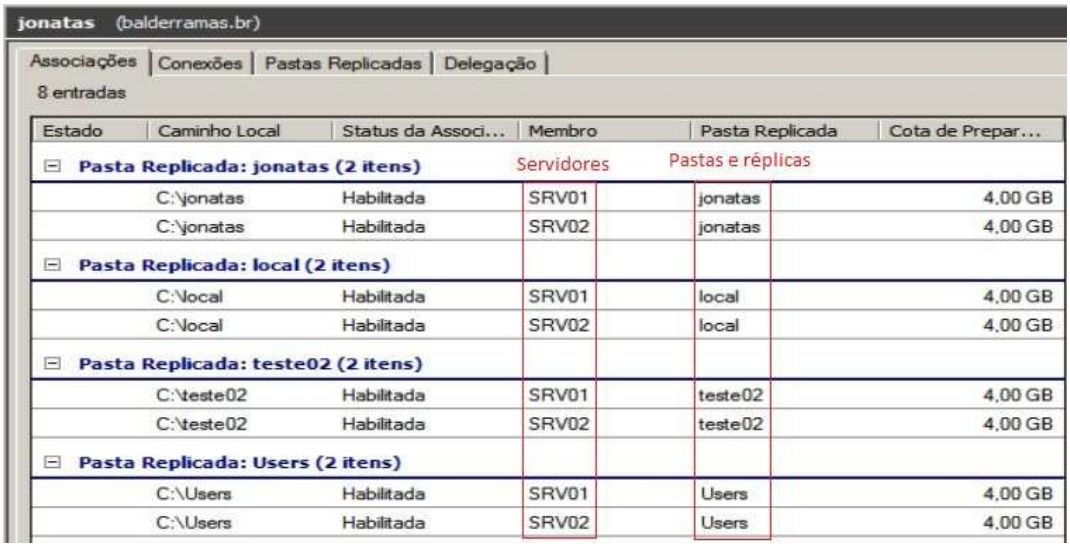

Figura 17 - Lista de replicação do DFS

Fonte: Própria

As pastas indicadas na lista tem de ser criadas em ambos os servidores e depois são criadas as regras de replicação. Após criar a replicação, as pastas replicadas foram adicionadas ao Namespace, para que pudessem aparecer aos usuários que acessam o diretório virtual do DFS.

Feito isso foi efetuada a copia de um arquivo qualquer para o atalho criado anteriormente no computador do usuário que concede acesso ao DFS, em seguida foi observado no servidor primário a presença do arquivo salvo pelo usuário.

A replicação não ocorre de imediato, portanto foi preciso aguardar vários segundos até que o servidor secundário recebesse a cópia do arquivo proveniente da replicação.

Após confirmar o sucesso da replicação do arquivo foi emulada então a falha do servidor primário do DFS. Para criar a falha emulada foi preciso simplesmente desconectar a rede do servidor primário, impedindo assim sua comunicação com os demais hosts.

Após emular a falha do servidor voltei ao computador do usuário que já estava acessando o diretório do DFS e tentei abrir novamente o arquivo salvo, notei que houve uma demora maior que o comum para que o arquivo fosse aberto, mas mesmo assim o arquivo abriu normalmente após poucos segundos, mesmo sem sequer fechar a janela do diretório acessado.

Pelo fato do usuário desconhecer a existência de ambas as réplicas de seus arquivos e também por poder continuar acessando normalmente seus recursos presentes no DFS mesmo após a falha do servidor principal, este teste foi considerado bem sucedido, recebendo assim nota “Alta” para as transparências de Replicação e Localização.

6.2 APLICAÇÃO DOS EXPERIMENTOS NA PLATAFORMA LINUX

Para realizar os testes com a plataforma Linux, criei duas máquinas virtuais rodando o Linux Ubuntu 12.4 com o intuito de utilizá-las como servidor. O Linux não possui uma versão pré configurada para operar como servidor, portanto foi preciso instalar uma versão comum do sistema destinada aos usuários comuns e acrescentar os recursos necessários para criar o ambiente desejado. A Figura 18 mostra as máquinas criadas virtuais em execução.

Figura 18 - Maquinas Virtuais executando estações Linux Ubuntu
Fonte: Própria

Por padrão o Linux não se comunica com estações Windows, para que isso ocorra é preciso instalar um recurso bem conhecido do Linux, o “Samba”. Este recurso permite que o Linux torne-se um servidor de arquivos, além de permitir sua comunicação com estações Windows. O Samba também agrega diversos recursos de arquivos, um destes recursos é o DFS, portanto para instalar o DFS no Linux é preciso instalar o Samba.

No Linux os programas são distribuídos sob a forma de código fonte, para instalá-los é preciso primeiramente compilá-los e então instalar os arquivos compilados. Para facilitar o processo de instalação o Linux Ubuntu dispõe de uma ferramenta automatizada que efetua a busca do *software* desejado, baixa a versão mais recente disponível, compila e instala automaticamente tudo o que for necessário para o correto funcionamento dos *softwares* desejados, este recurso é denominado “apt-get”, através dele realizei a instalação do Samba.

Para instalar o Samba no Ubuntu bastou digitar no Terminal o comando exibido na Figura 19.

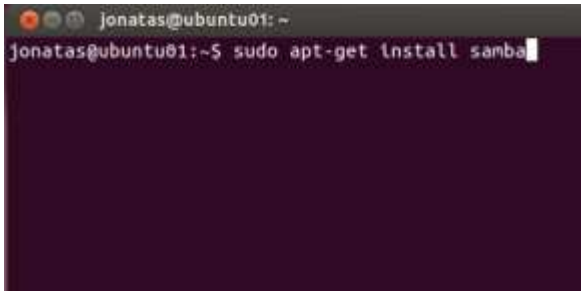

Figura 19 - Instalando o Samba.

Fonte: Própria

Logo após o término de sua instalação foi preciso editar o arquivo de configuração do Samba, o smb.conf através do comando exibido na Figura 20.

Figura 20 - Configurando o DFS no Samba.

Fonte: Própria

.

No arquivo smb.conf são configurados todos os parâmetros necessários para o funcionamento do Samba e seus recursos, incluindo o DFS, a replicação e o acesso aos hosts. A Figura 21 mostra alguns dos parâmetros necessários para serem adicionados no arquivo smb.conf.

Figura 21 - Configurando parâmetros do DFS no Samba
Fonte: Própria

Para habilitar e configurar o DFS no Samba, é necessário adicionar as seguintes linhas ao smb.conf, a primeira linha mostrada na imagem habilita o serviço do DFS no Samba, já a ultima linha habilita a replicação dos dados entre os servidores. Os parâmetros mostrados são apenas os principais, mas existem muitos outros comandos necessários para o perfeito funcionamento do DFS no Samba, como o foco do experimento não está na configuração dos serviços, para não fugir ao tema as demais configurações foram deixadas em segundo plano.

6.2.1 TRANSPARÊNCIA DE ACESSO NO LINUX

Este cenário é o mesmo que foi usado nos testes com o a transparência de acesso do Windows. Para o Linux existe uma ferramenta similar ao Active Directory da Microsoft, o OpenLDAP que passou a ser integrado ao Samba a partir de sua 4ª versão. Sua configuração é complicada até mesmo para usuários experientes e não existem assistentes de configuração com telas intuitivas, assim como no Windows, toda a configuração é realizada através de linhas de comando, em diversos arquivos de configuração que precisam ser editados.

Após configurar o Domínio “balderramas2.br” no Samba 4, o acesso do cliente ao diretório do DFS é realizado através deste domínio, que quando acessado exibe as pastas adicionadas ao DFS, semelhantemente ao Windows. Para que os testes com a plataforma Linux fossem iguais aos anteriormente realizados com o Windows, foi criado um mapeamento no cliente para o diretório do DFS através do

domínio, de modo semelhante a uma unidade local. A Figura 22 mostra como ficou o mapeamento na plataforma Linux.

Figura 22 - Mapeamento do DFS no cliente
Fonte: Própria

Também criei um atalho para esta unidade mapeada na Área de Trabalho do usuário, denominado de “Arquivos” conforme ilustrado pela Figura 23.

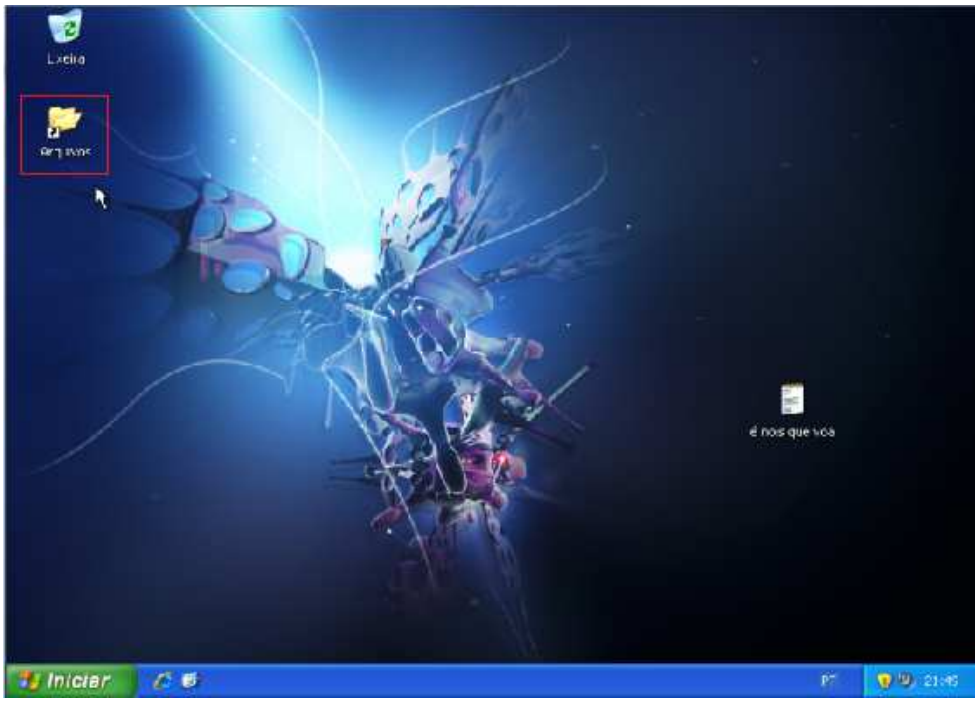

Figura 23 - Atalho para o Mapeamento do DFS no cliente
Fonte: Própria

No Linux o acesso aos arquivos do DFS, assim como no Windows, ocorre da mesma forma com que os arquivos locais são acessados, por este motivo este cenário foi considerado bem sucedido. Por este motivo a plataforma Linux Recebeu nota “Alta” para a Transparência de Acesso.

6.2.2 TRANSPARÊNCIA DE REPLICAÇÃO E LOCALIZAÇÃO NO LINUX

Para criar este cenário foi preciso configurar uma rotina para a replicação dos arquivos do DFS para o servidor secundário, no Linux isto é feito através de um comando, exibido na ultima linha da Figura 21, também é preciso criar um diretório que receberá a replicação, neste caso o diretório foi denominado de share1, ou seja, tudo o que for copiado para o diretório share1 presente no servidor primário será replicado a um diretório de mesmo nome no servidor secundário.

Foi copiado um arquivo qualquer para o ponto de montagem no cliente que concede acesso ao diretório “share1” dentro do DFS e, assim como no Windows, esperei até que o arquivo fosse replicado ao diretório do servidor secundário. No Linux notei que o processo de replicação possui uma latência consideravelmente menor que no Windows, talvez pelo consumo de recursos distintos dos dois sistemas.

Logo após confirmar a criação da réplica do arquivo no servidor secundário foi então preciso emular a falha no servidor primário, assim como realizado com o Windows, então desativei o adaptador de rede do servidor, deixando-o indisponível. Enquanto isso, no computador cliente o diretório do DFS permanecia aberto, no entanto quando tentei acessar novamente o arquivo criado recebi uma mensagem de que este estava indisponível, além disto, ocorreu o “travamento” do diretório, sendo preciso abri-lo novamente para obter acesso. O experimento foi repetido algumas vezes, na esperança de que o diretório não se tornasse inacessível após a falha do servidor primário, no entanto obtive o mesmo resultado em todas as tentativas.

Observando os resultados obtidos é possível constatar que apesar de não obter grande tolerância a falhas, os experimentos com as transparências de replicação e localização na plataforma Linux foram bem sucedidos, pois o arquivo do usuário foi replicado ao servidor secundário sem que o usuário possuísse qualquer conhecimento da existência desta réplica e também, apresar do “travamento”, o usuário obteve acesso ao arquivo replicado presente no servidor secundário utilizando o mesmo caminho acessado anteriormente, portanto tanto a transparência de replicação como a transparência de localização receberam nota “Alta”.

7 CONCLUSÃO

Após a aplicação dos experimentos propostos ficou muito claro que, apesar das diferenças existentes entre as plataformas Windows e Linux, o funcionamento do DFS em ambas é muito similar. Os experimentos foram aplicados em ambas as plataformas seguindo os mesmos critérios sem que houvesse a necessidade alterar suas características. Esta equivalência pôde ser alcançada graças ao “Samba”, que realmente cumpriu seu papel e proveu serviços similares aos disponíveis na plataforma da Microsoft.

O DFS mostrou-se muito estável no Windows, durante toda sua configuração e utilização não ocorreram problemas. Foi capaz de transmitir a segurança de uma ferramenta voltada ao uso profissional, com recursos que possibilitam sua aplicação em grandes redes, inclusive a WAN. O DFS no Windows obteve sucesso total, recebendo a nota “Alta” em todas as transparências testadas, além de mostrar-se tolerante a falhas.

No Linux a aplicação do DFS foi muito mais complicada pelo fato de que toda a configuração é feita através da inclusão de linhas de comando em diversos arquivos de configuração. Foram necessários vários dias de pesquisa em fóruns especializados, livros e outras fontes de conhecimento para reunir todo o conteúdo necessário para configurá-lo adequadamente e ainda assim, mediante a falha de um servidor, mostrou-se instável. Apesar de obter nota “Alta” em todas as transparências testadas, sua utilização em uma grande rede de dados pode se tornar problemática pela baixa tolerância a falhas apresentada.

O uso do Linux torna-se atraente por se tratar de uma plataforma gratuita e por permitir que os usuários possam customizá-lo segundo suas necessidades, além de possuir diversas distribuições estáveis e totalmente gratuitas, a exemplo do Ubuntu utilizado para a aplicação dos testes. No entanto, em um ambiente de produção que requer alta disponibilidade para os recursos distribuídos, o uso do Microsoft Windows Server mostrou-se mais indicado pelo fato de ser um sistema mantido por uma das maiores empresas de *software* do mundo e por possuir suporte técnico especializado, já para o Linux, não existem garantias de que os serviços

oferecidos funcionarão conforme o esperado pelo fato de que ele não possui um suporte técnico especializado, desta forma, mesmo que existam diversos fóruns especializados em Linux, o usuário terá que resolver qualquer problema que por ventura venha a encontrar por conta própria. Este fato foi claramente observado durante o processo de instalação e configuração do DFS no ambiente Linux visto que não pude encontrar com facilidade nos fóruns especializados conteúdos precisos que pudessem ser útil de alguma forma, todo o processo de configuração dos serviços necessários para criar um ambiente equivalente ao criado para o ambiente Windows no Linux consumiu mais que o dobro do tempo gasto para configurar todo o cenário sob a plataforma da Microsoft.

A Figura 24 ilustrará, sob a forma de uma tabela, os resultados obtidos com os experimentos realizados para comparar as transparências DFS em ambas as plataformas testadas.

Transparência / Plataforma	Linux	Windoows
Transparência de Acesso	"Alta"	"Alta"
Transparência de Replicação	"Alta"	"Alta"
Transparência de Localização	"Alta"	"Alta"

Figura 24 - Tabela Expositora dos Resultados Obtidos

Fonte: Própria

REFERÊNCIAS

ADYA, Atul et. al. Farsite: federated, available, and reliable storage for an incompletely trusted environment. In: ACM SIGOPS OPERATING SYSTEMS REVIEW, 36., 2002, New York. Anais... New York: ACM Press, 2002. p. 1-14.

BECK, Micah; MOORE, Terry; PLANK, James S. An end-to-end approach to globally scalable programmable networking. In: ACM SIGCOMM WORKSHOP ON FUTURE DIRECTIONS IN NETWORK ARCHITECTURE, 2003, New York. Anais... New York: ACM Press, 2003, p. 328-339.

BEINSYNC. BeInSync. Disponível em: <http://www.beinsync.com/>. Acesso em: 11 Mai. 2012.

BRAAM, Peter J. The Coda Distributed File System. Linux Journal, Seattle, v. 1998, n. 50es, 6 p., Jun. 1998.

CARVALHO, Roberto Pires de. Sistemas de Arquivos Paralelos e Distribuídos. São Paulo: 2003. 75 p. Tese (Mestrado em Ciência da Computação) - Instituto de Matemática e Estatística, Universidade de São Paulo, 2003.

CHOW, Randy; JOHNSON, Theodore. Distributed operating systems and algorithms. Reading, Massachusetts: Addison Wesley Longman, 1998. 569 p.

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. Distributed systems: concepts and design. 4. ed. Harlow: Addison Wesley Longman, 2005. 927 p.

DABEK, Frank et al. Wide-area cooperative storage with CFS. In: ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES, 18., 2001, New York. Anais... New York: ACM Press, 2001. p. 202-215.

GALLI, Doreen L. Distributed Operating Systems: Concepts and Practice. New Jersey: Prentice Hall, 2000. 463 p.

GHEMAWAT, Sanjay; GOBIOFF, Howard; LEUNG, Shun-Tak. The Google File System. In: ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES, 19., 2003, New York. Anais... New York: ACM Press, 2003. p. 29-43.

GIL, Antonio Carlos. Como Elaborar Projetos de Pesquisa. 4. ed. - São Paulo : Atlas, 2002. 176 p.

HARKES, Jan. Re: Multiple coda servers. Coda Mailing Lists, 11 Set. 1999. Disponível em: <http://www.coda.cs.cmu.edu/maillists/codalist/codalist-1999/1749.html>. Acesso em: 11 Fev. 2012.

HARKES, Jan. Re: what a realm is exactly? Coda Mailing Lists, 28 Out. 2004. Disponível em: <http://www.coda.cs.cmu.edu/maillists/codalist/codalist-2004/6966.html>. Acesso em: 11 Fev. 2012.

KIRNER, Claudio; MENDES, Sueli B. T. Sistemas operacionais distribuídos: aspectos gerais e análise de sua estrutura. Rio de Janeiro: Campus, 1988. 184 p.

KISTLER, James J.; SATYANARAYANAN M. Disconnected operation in the Coda file system. In: ACM SYMPOSIUM ON OPERATING SYSTEMS PRINCIPLES, 13., 1991, New York. Anais... New York: ACM Press, 1991, p. 213-225.

KON, Fabio. Distributed File Systems Past, Present and Future. 1996.

KON, Fabio. Sistemas de Arquivos Distribuídos. São Paulo: 1994. 154 p. Tese (Mestrado em Matemática Aplicada) - Instituto de Matemática e Estatística, Universidade de São Paulo, 1994.

MEDIAMAX. MediaMax. Disponível em: <http://www.mediamax.com/>. Acesso em: 20 Fev. 2012.

Microsoft. TechNet. Disponível em: http://technet.microsoft.com/pt-br/library/cc736868%28v=ws.10%29.aspx. Acesso em: 29 outubro 2012.

SIEWIOREK, Daniel P. Architecture of Fault-Tolerant Computers. Computer, IEEE Computer Society, v. 17, n. 8, p. 9-18, Ago. 1984.

SILBERSCHATZ, Abraham; GALVIN, Peter B.; GAGNE, Greg. Applied Operating System Concepts. New York: John Wiley & Sons, 2000. 840 p.

SINGHAL, Mukesh; SHIVARATRI, Niranjan G. Advanced concepts in operating systems: distributed, database, and multiprocessor operating systems. Nova York: McGraw-Hill, 1994. 522 p.

TANENBAUM, Andrew S. Computer Networks. New Jersey: Prentice Hall, 2003. 384 p.

TANENBAUM, Andrew S.; WOODHULL, Albert S. Operating Systems Design and Implementation. 3. ed. New Jersey: Prentice Hall, 2006. 1080 p.

TANENBAUM, Andrew S.; STEEN, Maarten van. Distributed Systems: Principles and Paradigms. New Jersey: Prentice Hall, 2002. 803 p.

VERISSIMO, Paulo; RODRIGUES, Luis. Distributed Systems for System Architects.Norwell: Kluwer Academic Publishers, 2001. 623 p.

WOHLIN, Claes et. al. Experimentation in Software Engineering: An Introduction. Norwell: Kluwer Academic Publishers, 2000. 204 p.

APÊNDICE A – Artigo SBC

Transparência em Sistemas de Arquivos Distribuídos

Jônatas Minucci Balderramas, Prof. Esp. Henrique Pachioni Martins, Prof .Dr. Elvio Gilberto da Silva, Prof. Esp. Andre Ferraz Castro

Ciências da Computação – Universidade Sagrado Coração (USC)
CEP 17011-160 - Bauru – SP - Brasil

Departamento de Ciências Exatas e Sociais Aplicadas
Universidade Sagrado Coração (USC) – Bauru, SP – Brasil
jonatasbalderramas@gmail.com, henrique.martins@usc.br, egsilva@usc.br, andre.castro@usc.br

Abstract. *The file system is responsible for organizing, storing, retrieving, naming, sharing and protecting files. For a long time File System coverage was local, but the constant presence of Computer Networks fostered the emergence and spread of Distributed File Systems (DFS), which has the same task File System, but aggregates breadth local and foreign. For a DFS allows access to files stored on a server with performance and reliability similar to the files stored on local computers must face one of the biggest challenges in the deployment of these systems, Transparency. This research aims to identify, through the application of experiments, the degree of replication transparency, access transparency and location transparency in existing tools DFS on different operating systems.*

Resumo. *O Sistema de Arquivos é responsável pela organização, armazenamento, recuperação, denominação, compartilhamento e proteção de arquivos. Por muito tempo o Sistema de Arquivos tinha abrangência local, mas a presença constante das Redes de Computadores propiciou o surgimento e a disseminação de Sistemas de Arquivos Distribuídos (SAD), que tem a mesma incumbência do Sistema de Arquivos, mas agrega abrangência local e externa. Para que um SAD possibilite acesso a arquivos armazenados em um servidor com desempenho e confiabilidade semelhante aos arquivos armazenados em computadores locais é preciso enfrentar um dos maiores desafios presentes na implantação destes sistemas, a Transparência. Esta pesquisa tem o intuito de identificar, através da aplicação de experimentos, o grau da transparência de replicação, transparência de acesso e transparência de localização existente nas ferramentas de SAD em diferentes sistemas operacionais.*

1. INTRODUÇÃO

A vulnerabilidade dos dados gravados em mídias de armazenamento portátil aliado a crescente oferta de largura de banda e a redução acentuada do custo para utilização de meios de transmissão, os protocolos HTTP, FTP e serviços Peer-to-Peer (P2P) tornaram-se alternativas acessíveis para o armazenamento e transferência de arquivos entre computadores pessoais. Alguns analistas prevêem que, no futuro, os dados que ainda são armazenados localmente, residirão em servidores existentes na Internet. Alguns sistemas operacionais possuem ferramentas nativas que possibilitam o armazenamento distribuído de arquivos, a exemplo do Microsoft Windows Server, que em algumas versões é integrado pelo DFS. Esta ferramenta possibilita acessar recursos compartilhados na rede interna e na WAN de uma forma centralizada, sem que o usuário precise saber em qual servidor ou computador o recurso compartilhado se encontra. Grandes redes de dados com abrangência global abrigam diversos equipamentos distintos, a exemplo dos servidores que executam sistemas operacionais distintos e protocolos de comunicação distintos. A questão da interoperabilidade também pode afetar as ferramentas de sistemas de arquivos distribuídos, a exemplo do DFS, que também possui aplicações para a plataforma Linux.

2. MOTIVAÇÃO

Os sistemas distribuídos representam um grande avanço para a informática, oferecendo um conjunto de novas possibilidades e abrindo caminho para novas tecnologias, mas existem também alguns conceitos implícitos a estes sistemas que devem ser consideradas ao debatermos sobre Sistemas de Arquivos Distribuídos (SAD), a exemplo da transparência.

> "*Sistemas de Arquivos Distribuídos são uma coleção de computadores independentes que aparecem para os usuários do sistema como um único computador" (TANENBAUM, 2002, p. 2).*

É notável que os sistemas distribuídos tenham uma relação muito estreita com a transparência, uma coleção de computadores deve aparecer ao usuário como um único computador para que possa ser denotado como um sistema distribuído. Este artigo propõe a utilização da transparência de acesso, transparência de replicação e transparência de localização como meio para comparar a interoperabilidade do Distributed File System (DFS) para as plataformas Microsoft Windows e Linux

3. DISTRIBUTED FILE SYSTEM - DFS

Segundo O DFS permite que os administradores agrupem pastas compartilhadas localizadas em diferentes servidores, conectando-os de forma transparente a um ou mais espaços de nomes DFS. Um espaço de nomes DFS é uma exibição virtual de pastas compartilhadas em uma organização.

Usando as ferramentas do DFS, um administrador pode selecionar que pastas compartilhadas deverão ser apresentadas no espaço de nomes aos usuários do sistema, projeta a hierarquia na qual as pastas aparecerão e determina os nomes que as pastas compartilhadas exibirão no espaço de nomes.

Quando um usuário visualiza o espaço de nomes, as pastas parecem residir em um único disco rígido, transmitindo assim uma visão unificada do sistema, desta forma os usuários podem navegar pelas pastas no espaço de nomes sem que precisem saber os nomes dos servidores ou das pastas compartilhadas que hospedam os dados.

O DFS também oferece muitos outros benefícios, incluindo tolerância a falhas e a capacidade de compartilhamento de carga, recursos que o tornam ideal para organizações que necessitam de um alto nível de disponibilidade a seus arquivos de dados (MICROSOFT, 2012).

4. METODOLOGIA

Para alcançar os resultados esperados foram realizados experimentos que identificaram o grau de transparência das ferramentas de sistemas distribuídos classificando-os em alto e baixo. Dois cenários distintos foram elaborados, o primeiro cenário experimentou a transparência de acesso e o segundo a transparência de localização e replicação do DFS em cada plataforma. Os cenários propostos neste experimento foram repetidos em ambas as plataformas testadas, onde o critério para indicar o grau de transparência alto foi a obtenção do sucesso no respectivo cenário e o critério para indicar o grau de transparência baixo foi consequentemente a não obtenção do sucesso no respectivo cenário.

A transparência de acesso possibilita que os recursos remotos de um sistema de arquivos distribuídos sejam acessados de forma semelhante aos recursos do sistema local. Desta forma, para identificar o grau da transparência de acesso existente nas ferramentas neste cenário, foi observado como os usuários do sistema obtêm acesso aos recursos presentes no sistema de arquivos distribuídos. A condição esperada para a obtenção do sucesso neste cenário é a mesma para ambas as plataformas, pelo fato de que o cliente executa o Microsoft Windows XP. O cliente deve obter acesso ao DFS através de um ponto de montagem similar a um diretório local que exibe seus recursos sob a forma de diretórios, pastas e arquivos. Desta forma o sistema recebe a nota “Alta” para a transparência de acesso. A não obtenção de sucesso neste cenário ocorre caso os usuários necessitem utilizar qualquer tipo de aplicativo ou ferramenta de terceiros para obter acesso aos seus arquivos armazenados dentro do sistema de arquivos distribuído, como por exemplo, um browser ou outro aplicativo que não faça parte do sistema operacional. Em outras palavras, se existir qualquer característica que defira

da forma tradicional do sistema explorar pastas e diretórios locais, no caso, pelo fato de que o cliente executará o Microsoft Windows XP, qualquer forma de acesso que não ocorra através do Windows Explorer definirá o cenário como mal sucedido. Desta forma o sistema recebe a nota “Baixa” para a transparência de acesso.

Para realizar os testes com a transparência de replicação e de localização, foi montado um cenário onde ocorreu a cópia de um arquivo qualquer para o local criado no computador do cliente que concede acesso ao ponto de montagem do sistema de arquivos distribuídos e, em seguida, foi simulada a falha do servidor primário. No entanto para garantir o sucesso do experimento descrito neste cenário, a falha do servidor primário foi simulada somente após o servidor secundário receber a cópia replicada do arquivo do usuário. O intuito deste cenário é identificar a transparência com que o processo de replicação ocorre para o usuário final do sistema, por isto, mesmo que desejável, o fator tempo foi desconsiderado, visto que para medi-lo seria preciso controlar todas as variáveis descritas acima capazes de influenciá-lo. A obtenção de sucesso neste cenário ocorre caso o cliente continue acessando seus arquivos através do ponto de montagem para o qual ele foi copiado anteriormente, mesmo após a falha do servidor primário. Desta forma o sistema recebe a nota “Alta” para a transparência de replicação. A não obtenção de sucesso neste cenário ocorre caso os usuários percebam de alguma forma que o diretório ou o arquivo presente no sistema de arquivos distribuídos tornou-se indisponível após a falha do servidor primário. Os usuários também não devem ter conhecimento de que o seu acesso foi direcionado a um novo servidor. Desta forma o sistema recebe a nota “Baixa” para a transparência de replicação.

Se bem sucedido, o ambiente proposto neste cenário, além de oferecer a transparência de replicação, também disponibilizará aos clientes um serviço de armazenamento distribuído tolerante a falhas, isto significa que o cliente não precisa se preocupar com a localização física entre os volumes dos servidores e dos arquivos que ele manipula o que indica que caso o servidor acessado falhe, suas requisições serão encaminhadas ao segundo servidor ativo, sem o conhecimento do cliente que continuará a usufruir dos serviços, desta vez localizados em outro servidor. Por este fato, pode-se afirmar que, dependendo do resultado obtido com os experimentos da transparência de replicação, a transparência de localização também estará presente na plataforma, caso o experimento com a transparência de replicação seja bem sucedido. Desta forma o sistema recebe também a nota “Alta” para a transparência de localização. Caso o resultado do experimento proposto neste cenário para a identificação da transparência de replicação seja mal sucedido, pode-se afirmar que a transparência de localização também não estará presente, desta forma o sistema recebe também a nota “Baixa” para a transparência de localização.

5. RESULTADOS OBTIDOS

Após a aplicação dos experimentos propostos ficou muito claro que, apesar das diferenças existentes entre as plataformas Windows e Linux, o funcionamento do DFS em ambas é muito similar. Esta equivalência pôde ser alcançada graças ao recurso “Samba” do Linux, que realmente cumpriu seu papel e proveu serviços similares aos disponíveis na plataforma da Microsoft. Os experimentos foram aplicados em ambas as plataformas seguindo os mesmos critérios sem que houvesse a necessidade alterar suas características. A Figura 1 ilustra, sob a forma de uma tabela, os resultados obtidos com os experimentos realizados para comparar as transparências DFS em ambas as plataformas testadas.

Transparência / Plataforma	Linux	Windoows
Transparência de Acesso	"Alta"	"Alta"
Transparência de Replicação	"Alta"	"Alta"
Transparência de Localização	"Alta"	"Alta"

Figura 1 – Resultados obtidos

6. CONSIDERAÇÕES FINAIS

O DFS mostrou-se muito estável no Windows, durante toda sua configuração e utilização não ocorreram problemas. No Windows o DFS foi capaz de transmitir a segurança de uma ferramenta voltada ao uso profissional, com recursos que possibilitam sua aplicação em grandes redes, inclusive a WAN, já no Linux a aplicação do DFS foi muito mais complicada pelo fato de que toda a configuração deve ser realizada através da inclusão de linhas de comando em diversos arquivos de configuração. No Linux também ocorreu um problema quando o servidor primário tornou-se indisponível, ocasionando o travamento do diretório acessado pelo usuário, porém bastou abrir o mesmo novamente para que este voltasse a ficar disponível, mesmo sob a falha do servidor primário.

Este estudo demonstra que é possível utilizar o DFS sob a plataforma Linux, porém para grandes redes de dados onde exista necessidade de alta disponibilidade dos recursos distribuídos é altamente recomendável o uso do Microsoft Windows Server, pelo fato de ser um sistema mantido por uma das maiores empresas de software do mundo e por possuir suporte técnico especializado, já para o Linux, não existem garantias de que os serviços oferecidos funcionarão conforme o esperado pelo fato de que ele não possui um suporte técnico especializado, desta forma, mesmo que existam diversos fóruns especializados em Linux, o usuário terá que resolver qualquer problema que por ventura venha a encontrar por conta própria.

REFERÊNCIAS

CARVALHO, Roberto Pires de. Sistemas de Arquivos Paralelos e Distribuídos. São Paulo: 2003. 75 p. Tese (Mestrado em Ciência da Computação) - Instituto de Matemática e Estatística, Universidade de São Paulo, 2003.

CHOW, Randy; JOHNSON, Theodore. Distributed operating systems and algorithms. Reading, Massachusetts: Addison Wesley Longman, 1998. 569 p.

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. Distributed systems: concepts and design. 4. ed. Harlow: Addison Wesley Longman, 2005. 927 p.

Microsoft. TechNet. Disponível em: http://technet.microsoft.com/pt-br/library/cc736868%28v=ws.10%29.aspx. Acesso em: 29 outubro 2012.

KON, Fabio. Distributed File Systems Past, Present and Future. 1996.

KON, Fabio. Sistemas de Arquivos Distribuídos. São Paulo: 1994. 154 p. Tese (Mestrado em Matemática Aplicada) - Instituto de Matemática e Estatística, Universidade de São Paulo, 1994.

TANENBAUM, Andrew S.; STEEN, Maarten van. Distributed Systems: Principles and Paradigms. New Jersey: Prentice Hall, 2002. 803 p.