U-STORE: Implementação de Data Cloud utilizando Ferramentas P2P

Rodrigo Elia Assad,
Silvio R. de Lemos Meira
Centro de Informática - UFPE
C.E.S.A.R
Recife, Pernambuco, Brazil
{rea,srlm}@cin.ufpe.br

Vinicius Cardoso Garcia
Centro de Informática - UFPE
Recife, Pernambuco, Brazil
vcg@cin.ufpe.br

Fernando Antonio Mota Trinta
Instituto UFC Virtual
Universidade Federal do Ceará
Fortaleza, Ceará, Brazil
fernando.trinta@virtual.ufc.br

ABSTRACT
This article presents the specification and implementation of a tool that aims to be an alternative to the problem of excessive data replication in order to be considered a reliable platform from the perspective of data availability. The tool works with the concept of data federation. The validation scenarios are presented, along with the results achieved.

Palavras-chave
Cloud Computing, Data Cloud, Federações, Armazenamento como Serviço, P2P, Confiabilidade.

1. INTRODUÇÃO
Com o aumento da quantidade de dados produzida pelos usuários domésticos através do uso de maquinas fotográficas, pen drivers, musicas, filmes e outros dispositivos produtores de recursos multimídia, bem como pelos sistemas em produção nas empresas que necessitam armazenar dados históricos, surge a necessidade de plataformas de retenção de dados [1, 2].

O modelo tradicionalmente aplicado pelos usuários em empresas é da aquisição de *storages* para armazenamento destes dados. No entanto, para os usuários finais, os *storages* domésticos trazem a complexidade de gerenciá-los e operá-los, já para as empresas o custo é um dos fatores mais considerados.

Um efetivo sistema de backup deve ser norteado por dois princípios básicos: (i) possuir uma infra-estrutura dedicada para garantir a disponibilidade dos dados quando o usuário solicitar o seu acesso ou recuperação e (ii) ser totalmente confiável do ponto de vista da garantia da disponibilidade dos dados para acesso ou recuperação.

Neste contexto, este artigo apresenta a ferramenta U-STORE a partir dos resultados obtidos por [3] onde foi definido um algoritmo estatístico que visa garantir a disponibilidade de dados armazenados em ambientes P2P. A utilização desta plataforma (P2P) permite a construção de um ambiente totalmente distribuído para armazenamento dos dados por meio da disponibilização de Peers que ofertam serviços na rede, dentre eles: a) a re-distribuição dos dados de forma dinâmica, com o intuito de prover a disponibilidade do serviço de armazenamento sem haver necessidade de excesso de replicações; b) criação de federações de dados, definindo assim uma *Data Cloud*.

WebMedia'11: Proceedings of the 17th Brazilian Symposium on Multimedia and the Web. X Workshop on Tools and Applications.
October 3 -6, 2011, Florianópolis, SC, Brazil.
ISSN 2175-9650.
SBC - Brazilian Computer Society

2. OBJETIVOS E VANTAGENS
O objetivo da ferramenta U-STORE é permitir a criação de uma *Data Cloud* onde a criação de federações e das replicações, que permitam garantir a disponibilidade dos dados, é feita dinamicamente.

Como principal vantagem da utilização desta abordagem pode-se destacar a possibilidade da diminuição significativa no CAPEX (do inglês *capital expenditure* que abrange as despesas de capital ou investimento em bens de capital) das organizações na aquisição de sistemas de armazenamento (para grandes volumes de dados), visto que os recursos utilizados são os já existentes e adquiridos pelas organizações.

O acesso a esta plataforma é disponibilizado de duas formas: (i) através de uma partição no sistema operacional de cada dispositivo pertencente à rede e (ii) através de uma interface SOA [4], utilizando protocolo REST [5] que permitirá a criação de aplicações de acesso aos dados e uma futura integração com sistemas existentes e que necessitem armazenar grande volume de dados.

3. SISTEMA P2P CONFIÁVEL PARA DATA CLOUD
A ferramenta U-STORE pode ser vista como um conjunto de *peers* em uma rede P2P que se agrupam dinamicamente formando federações de dados. São necessários três tipos de *peers* para compor a rede: a) um super *peer* que pode ser visto como um Grid computacional [6] que escolhe dinamicamente os peers e servidores das federações baseando-se em um algoritmo de proximidade [3]; b) *peer* local (ou *SimplePeer)* , que se anuncia para o *super peer* e possui os dados a serem salvos; c) servidor que recebe o anuncio de cada *peer* local e compõe a federação de dados. Cada *peer* possui uma interface de serviço REST [5] que permite a autenticação do usuário, salvamento, recuperação e remoção dos dados salvos. O super *peer* mantém uma lista com os endereços IP e federações e informa cada *peer* cliente quando ele solicita que um dado seja salvo. A arquitetura das federações gerenciadas pela ferramenta é apresentada na Figura 1.

Ainda na Figura 1 é descrita a proposição de como o sistema final irá funcionar para uma federação. Neste caso a interface de acesso aos dados será uma interface REST compatível com o S3 da Amazon. Nesta arquitetura, o serviço de armazenamento dos

dados pode ser modificado para outra alternativa (i.e. Rackspace[†] ou Nimbus[‡]).

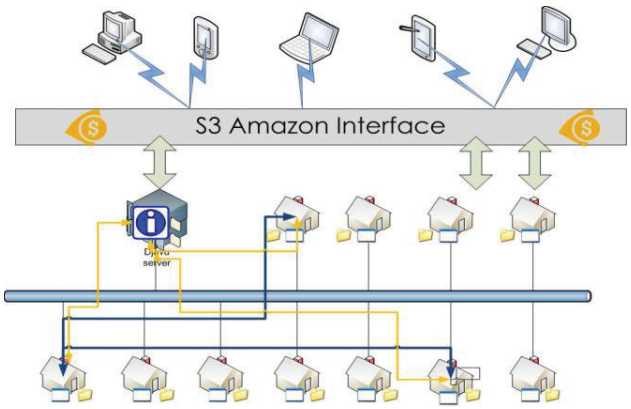

Figura 1 - Modelo geral do sistema

Cada *peer* deve informar seu perfil ao se conectar na rede, ou seja, a sua disponibilidade de permanência on-line conectado a rede por hora. Em resumo, cada *peer* informa qual os horários em que ele estará disponível (ligado e online) e o seu SLA (ou *service level agreement* que é parte de um contrato de prestação de serviços onde o nível do serviço – bem como sua disponibilidade, efetividade ou qualidade – é formalmente definido) na rede. Conforme descrito na sessão 1, a U-STORE utilizou como base o trabalho desenvolvido em [5] que especificou um algoritmo que, baseado nos perfis das máquinas (sua agenda de disponibilidade online) calcula em quantas maquinas deve-se replicar os dados, de forma que um usuário tenha a garantia estatística de acesso aos dados na *data cloud* quando desejar. No intuito de manter a igualdade no provimento e uso dos recursos da *data cloud,* cada *peer* somente poderá solicitar a garantia máxima de acesso aos seus dados, caso oferte em seu perfil este período em termos de disponibilidade.

A Figura 2 apresenta a relação entre perfil, algoritmo, arquivos recebidos e a função de confiabilidade definida. Conforme apresentado, é calculado dinamicamente em quantos *peers* cada *hash* e *chunk* do arquivo deve ser salvo para que quando solicitado, se tenha a garantia do acesso ao mesmo.

Um possível problema para esta abordagem é aferir que o *peer* possui a disponibilidade descrita no seu perfil. Para aferir isso o servidor que gerencia a sua federação recebe a cada cinco minutos uma mensagem de *keep alive,* oriunda de todos os *peers* sobre seu controle, permitindo assim a aferição da sua disponibilidade real na rede.

Este mecanismo de *keep alive* permite que o perfil seja aferido e caso haja a violação deste SLA o mesmo é redefinido de forma dinâmica, através de um mecanismo automatizado de re-distribuição dos dados. Isto ocorre porque como o servidor sabe onde os dados estão salvos, o perfil definido pelo *peer* (e seu SLA) e a real disponibilidade do *peer* na rede. As atividades realizadas pelo servidor no caso do não cumprimento ou indisponibilidade de um *peer*, são:

a) Remover da lista de *peers* disponíveis para aquele horário;

b) Identificar os pedaços que este *peer* possui;

c) Selecionar um novo *peer* com disponibilidade semelhante;

d) Copiar os dados de um dos *peer* para este novo *peer*; e,

e) Quando o peer que teve a falha voltar, ele será informado da falha para remover os pedaços que foram replicados.

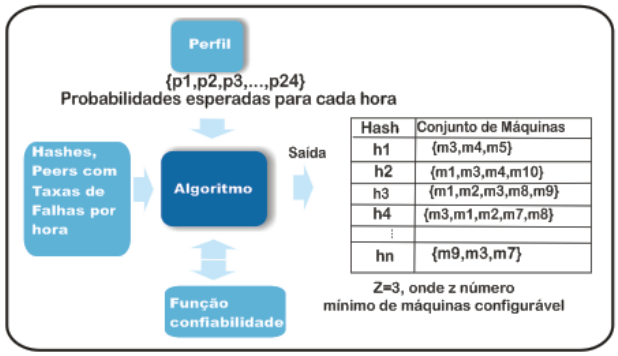

Figura 2 - Definição do perfil

Desta forma se pode ter a garantia de que um dado esteja disponível quando um usuário o solicitar, por meio da disponibilidade dos *peers* que se baseia no confronto entre a sua taxa de disponibilidade informada versus a sua taxa de falhas

4. ARQUITETURA DA FERRAMENTA

Esta seção relata as orientações para o projeto e implementação do sistema de acordo com a arquitetura estabelecida. A Figura 3 ilustra a visão de implementação da arquitetura do sistema.

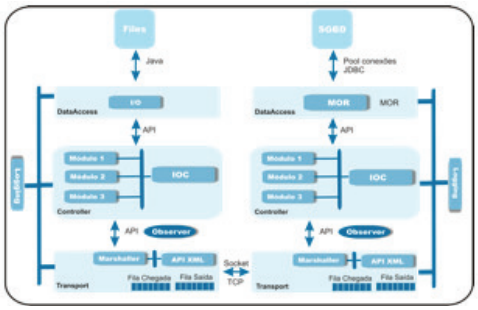

Figura 3 - ARQUITETURA DA U-STORE

Fisicamente a figura acima representa máquinas diferentes que se comunicam por meio da camada de transporte usando o protocolo JXTA [7]. O bloco denominado API XML é uma biblioteca usada para transcrever as mensagens do mundo orientado a objeto para a linguagem neutra (XML) usada na comunicação. Usando o padrão *Observer,* eventos são disparados para a lógica de negócio, sinalizando a chegada de uma mensagem na fila. A fila de saída é checada através de um tempo parametrizável por arquivos de configuração. O módulo IOC agrupa, por meio de inversão de controle, um conjunto de serviços úteis a todos os módulos da camada no qual está presente. O bloco MOR é um *framework* de mapeamento objeto relacional usado para transcrever mensagens da aplicação para o sistema gerenciador de banco de dados. E por fim, o bloco de *Logging* tem o objetivo de proporcionar aos administradores a opção de auditoria das transações efetuadas por meio do sistema

5. RESULTADOS DOS TESTES

Os testes foram realizados em 3 cenários distintos. Os dois primeiros foram mais focados para testar a viabilidade do

[†] http://www.rackspacecloud.com/, Acessado em 08/07/2011

[‡] http://www.nimbusproject.org/, Acessado em 08/07/2011

algoritmo de seleção de *peers*, já o terceiro projeto piloto de validação foi realizado com a ferramenta implementada, e a validação ocorreu na infra-estrutura do CESAR[§]. Neste caso verificou-se a escalabilidade da solução adicionando-se *peers* incrementalmente até alcançar 40 clientes P2P com 5Gb de armazenamento máximo e dois servidores provendo as interfaces REST para acesso aos nós. Por questões de espaço, apenas o terceiro cenário será detalhado.

No 3º cenário a ferramenta já estava em fase de piloto e o que se procurava validar é a escalabilidade da mesma. No cenário de testes que pode ser acessado através da URL (http://storage.cesar.org.br:8080/ServiceConsumer/) se testou a capacidade do sistema crescer tanto verticalmente quanto horizontalmente. Para isso utilizou-se dois servidores provendo as interfaces de serviço REST que se conectavam a rede P2P, executavam o download do arquivo e o salvavam na rede P2P. durante o teste a medida que a) as interfaces de serviços começavam a se tornar ponto de gargalo outra era adicionado; b) a medida que era necessário mais nós a rede P2P estes eram adicionados, permitindo o sistema crescer.

Neste caso todos os *peers* possuíam a disponibilidade total no perfil, fazendo com que cada pedaço fosse replicado de 2 a 3 maquinas.

O objetivo destes testes foi demonstrar que o sistema possui escalabilidade e aferir os tempos para a obtenção dos arquivos através de um cenário mais real utilizando interfaces de serviço. Os testes feitos para a leitura dos arquivos mostraram que para um arquivos com uma média te tamanho de 11Mb sendo salvos em um diretório compartilhado da rede Windows o tempo de espera para obter o arquivo foi de em média 28 segundos. Já em relação ao mesmo teste para realizar o download via a interface de serviço provida pelo sistema o tempo médio foi de 31 segundos.

Estes testes mostraram que, comparando-se com as soluções de armazenamento em rede o sistema possui um desempenho aceitável, se mostrando viável. A Figura 4 apresenta a tela do sistema com os arquivos salvos.

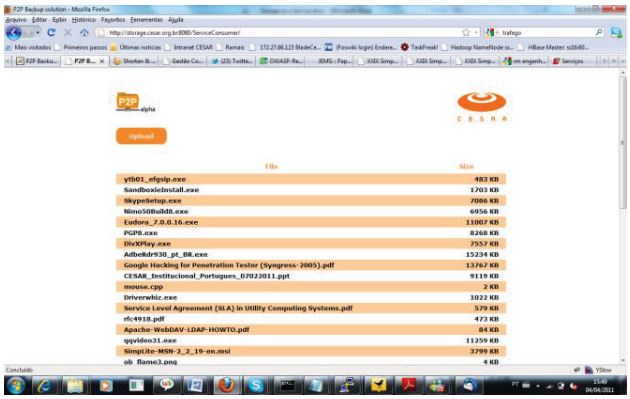

Figura 4 - Tela da ferramenta U-STORE (Cenário de Teste)

Atualmente esta se negociando como possíveis clientes interessados em prover aos seus usuários sistemas de armazenamento de dados como serviço e que queiram prover o mesmo sobre demanda e a baixo custo. Estamos em negociação com provedores de serviços e com *call centers*.

6. CONCLUSÕES

Conforme descrito, a ferramenta U-STORE provê a capacidade de se definir sistemas de armazenamento utilizando os conceitos de *cloud computing* a baixo custo e confiável. Esta ferramenta é composta por *peers* em uma rede P2P e por um algoritmo que permite calcular dinamicamente em quantos nós um *chunk* deve ser replicado de forma que quando se for solicitado a recuperação de um arquivo completo o sistema garanta que o mesmo estará disponível. Como atividades de melhorias futuras e novos trabalhos de pesquisa, temos:

a) Definição do tamanho ideal de cada *chunk* de forma a aumentar a eficiência do armazenamento bem como da recuperação dos dados (trabalho em andamento).

b) Definição de um sistema de bilhetagem que permita identificar quanto um determinado usuário da rede P2P esta utilizando de recursos e quanto o mesmo contribui. Este sistema já existe e esta sendo adequado a outro projeto de pesquisa para bilhetar dados (http://200.199.23.21/cloudmonitor/home.mvc/list)

Utilização e/ou adaptação de um algoritmo que permita agrupar de forma mais eficiente os *peers* em federações melhorando a eficiência do armazenamento e sua replicação. Este trabalho de pesquisa esta em andamento hoje.

7. AGRADECIMENTOS

Este trabalho é parcialmente apoiado pelo Instituto Nacional de Ciência e Tecnologia para Engenharia de Software (INES[**]), CNPq e FACEPE, processos 573964/2008-4, APQ-1037-1.03/08 e APQ-1044-1.03/10 e CNPq processos 475743/2007-5 e 140060/2008-1.

8. REFERÊNCIAS

[1] Sérgio R. Loest, Marcelo C. Madruga, Carlos A. Maziero, and Lau C. Lung. 2009. BackupIT: An Intrusion-Tolerant Cooperative Backup System. In *Proc. of the 8th IEEE/ACIS International Conference on Computer and Information Science* (2009). IEEE Computer Society, 724-729.

[2] Landon P. Cox and Brian D. Noble. 2003. Samsara: honor among thieves in peer-to-peer storage. *SIGOPS Oper. Syst. Rev.* 37, 5 (October 2003), 120-132.

[3] Duarte, Marcos Pinheiro, Um Algoritmo de Disponibilidade em Sistemas de Backup Distribuído Seguro Usando a Plataforma Peer-to-peer, Dissertação de Mestrado 2010 Programa de Pós-graduação, CIn-UFPE.

[4] Erl, Thomas (2005). Service-Oriented Architecture: Concepts, Technology & Design. New York: Prentice Hall/PearsonPTR. ISBN 0-131-85858-0.

[5] Jim Webber, Savas Parastatidis, Ian Robinson. REST in Practice: Hypermedia and Systems Architecture. O'Reilly, 2010, p. 448.

[6] M. Oliveira. OurBackup: A P2P backup solution based on social networks, MSc Thesis, Universidade Federal de Campina Grande, Brazil, 2007.

[7] JXTA; https://jxta.dev.java.net/; último acesso em 08/07/2011.

[§] http://www.cesar.org.br, Acessado em 28/08/2011

[**] http://www.ines.org.br, Acessado em 28/08/2011