LEONARDO CHAVES DUTRA DA ROCHA

UMA METODOLOGIA DE CARACTERIZAÇÃO DE SERVIÇOS DE MINERAÇÃO DE DADOS

Belo Horizonte, Minas Gerais

12 de agosto de 2005

LEONARDO CHAVES DUTRA DA ROCHA

UMA METODOLOGIA DE CARACTERIZAÇÃO DE SERVIÇOS DE MINERAÇÃO DE DADOS

Dissertação apresentada ao Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Belo Horizonte, Minas Gerais

12 de agosto de 2005

UNIVERSIDADE FEDERAL DE MINAS GERAIS

FOLHA DE APROVAÇÃO

Uma Metodologia de Caracterização de Serviços de Mineração de Dados

LEONARDO CHAVES DUTRA DA ROCHA

Dissertação defendida e aprovada pela banca examinadora constituída por:

Prof. Wagner Meira Junior – Orientador
Universidade Federal de Minas Gerais

Prof. Dorgival Olavo Guedes Neto
Universidade Federal de Minas Gerais

Prof. Renato Antonio C. Ferreira
Universidade Federal de Minas Gerais

Profa. Lucila Ishitani
Pontifícia Universidade Católica de Minas Gerais

Belo Horizonte, Minas Gerais, 12 de agosto de 2005

Resumo

Serviços Web estão se tornando um padrão para o desenvolvimento de um grande conjunto de aplicações que utilizam a Internet. Um exemplo desse tipo de aplicação é a mineração de dados, cujo objetivo é extrair informações úteis de um grande conjunto de dados. Um serviço Web de mineração de dados bem sucedido deve cumprir os requisitos de interação de uma tarefa de mineração de dados e os requisitos de processamento intensivo e armazenamento de grandes conjuntos de dados geralmente associados às técnicas empregadas. Nesta tese apresentamos uma metodologia para caracterização de serviços Web computacionalmente intensivos, em particular serviços de mineração de dados. Nossa metodologia de caracterização foca em ambos os lados do serviço, interativo e não interativo, bem como o relacionamento entre eles. Nós aplicamos nossa metodologia a um serviço de mineração de dados real, o Tamanduá. Os resultados mostram que há uma alta variabilidade entre os usuários em termos de comportamento, mas eles agem de forma similar com respeito à natureza das tarefas de mineração que eles requisitam e em como eles analisam os resultados. Nossos resultados também mostram que nós podemos dividir os usuários em dois grupos distintos, um grupo que utiliza o sistema de forma seqüencial e outro que apresenta um comportamento assíncrono, impondo uma demanda maior ao sistema. Esses resultados não só mostram a aplicabilidade de nossa proposta, mas também abre novas direções em termos de mecanismos de sustentação para esse tipo de sistema Web. Além disso, nós também determinamos modelos de distribuições estatísticas que podem ser utilizadas para a geração de cargas sintéticas com a finalidade de realizar uma detalhada análise de desempenho do sistema.

Abstract

Web services are becoming a standard for deploying a large spectrum of applications using the Internet. One example of such application is data mining, which aims to extract useful information from large sets of data. A successful data mining service must fulfill both interaction requirements of a data mining task, and the intensive processing and large storage requirements usually associated with the techniques employed. In this paper we present a methodology for characterizing computationally intensive Web services, in particular data mining services. Our characterization methodology focuses on both the interactive and non-interactive sides of the service, as well as their relationships. We applied our methodology on an actual data mining service, Tamanduá. The results show that there is a high variability among users in terms of their behavior, but they act similarly with respect to the nature of the data mining tasks they request and how they analyze the results. Our results also show that we are able to divide the users into two distinct groups, one that uses the system in a sequential fashion and other that presents an asynchronous behavior, placing a bigger demand on the system. These results not only show the applicability of our approach, but also open new directions in terms of system support mechanisms for such Web services. Further, we are also able to determine statistical distributions that may be used for generating synthetic workloads that would allow a detailed performance analysis of the system.

Agradecimentos

Primeiramente agradeço a Deus por mas essa oportunidade em minha vida. Agradeço a minha mãe querida, no verdadeiro sentido da palavra MÃE, por me ensinar os verdadeiros valores da vida como honestidade, caráter, humildade e perseverança. Agradeço pelo seu apoio incondicional, por ser o pai do Diogo na minha ausência (e que ausência nesses últimos anos...), por ser meu pai quando precisei, minha amiga nos momentos de dificuldade. Mãe, você me ensinou o quanto é importante o conhecimento e hoje cá estou em busca dele, cada vez mais. Muito obrigado! Te amo muito!!!

Agradeço aos meus dois pais: papai e vovô Chaves, que, mesmo não presente mais entre nós, de uma forma inexplicável, me sussurravam força e garra para lutar e não desistir do meu sonho. Agradeço aos meus irmãos Leandro e Luciano. Vocês dois são meus melhores amigos, companheiros. Mesmo sem entender muito bem o que eu estava fazendo, estavam sempre do meu lado! Amo demais vocês dois, e sempre será assim.

Agradeço a minha amada Elisa pelo companheirismo incondicional, procurando sempre entender minha ausência em casa, me apoiando em minhas decisões, por mais difíceis que a mesmas fossem, revezando comigo nas longas madrugadas dos primeiros meses de vida do nosso filho. Amor, sabemos o quanto foi e é difícil, passamos por tudo juntos, continuamos lutando juntos e sempre estaremos juntos...TE AMO!! Agradeço ao meu sogro José Braga e a a minha sogra Inez Helena, pelo amor e carinho com que vocês cuidaram e cuidam do nosso Diogo. Por me acolherem na família de vocês, não como o marido da Elisa mas como um filho.

Agradeço ao meu filho e amigo Diogo. Diogo, hoje, com 1 ano e 3 meses você não entende o quanto você foi, é e sempre será importante em minha vida. Graças a você meu filho, aprendi a ter um pouco mais de paciência, aprendi o quanto é bom amar de forma incondicional. Graças a você, me sinto muito importante porque te vendo assim, tão pequenino e tão inocente, vejo o quanto você precisa de mim. Se hoje você não entende, espero que um dia isso aconteça, mas independente de qualquer coisa, sempre te amarei.

Agradeço ao meu mestre, orientador e amigo Prof. Wagner Meira. Meira, trabalhamos juntos desde de 2000, quando após uma aula de AEDS3 você me convidou para uma iniciação científica. Desse dia para cá foram muitos trabalhos, muitos artigos publicados, muitos artigos recusados, mas ao mesmo tempo muita amizade e conversa. Seu entusiasmo, paciência e dedicação em ensinar é algo contagiante. Hoje, após 5 anos trabalhando juntos, tenho por você uma profunda admiração e respeito. Muito obrigado mesmo Meira!! Agradeço também aos professores e amigos Renato e Dorgival por serem pessoas extraordinárias! Na ausência do Meira, eram vocês que me ajudavam, era a vocês que eu recorria com minhas dúvidas, e vocês, sempre muito educados e com toda a paciência do mundo, me ajudavam. Agradeço demais a vocês pelas longas conversas sobre a vida, me aconselhando diante de algumas das dificuldades nesses quase dois anos.

Devo grande parte desse trabalho ao meu amigo e companheiro de trabalho Pedro Calais. São dois anos de convivência de muito trabalho e aprendizado. Agradeço muito a você meu amigo, por abraçar este trabalho como se fosse seu próprio trabalho, obrigado mesmo, de coração. Tenho certeza de que daqui a dois anos será você quem estará apresentando sua dissertação!! Muito obrigado Pedro, e continue assim!

Ao pessoal da secretaria (Túlia, Maristela, Renata, Gilmara, Cida, Sheila, Lu, Fernanda, Cláudia, Stella, Valéria, Sônia , Emília , Lizete, Orlando, Alexandre, Gilberto e Geraldo) meu muito obrigado pelo apoio e amizade durante todos esses anos. Vocês todos são pessoas maravilhosas!

Agradeço ao pessoal do speed, aos desmaiados, ao pessoal de Lafaiete e ao Bar do Beto pelos momentos de descontração. Não só de trabalho vive o homem, e vocês contribuíram e muito nessa caminhada, me proporcionando momentos de alegria e descontração para esquecer um pouco o trabalho árduo que está por trás de uma dissertação de mestrado... E ai, vamo pro Beto? ;-)

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A Web tem ido muito além de sua proposta inicial de ser uma plataforma simples de troca documentos e está se tornando de fato um ambiente para desdobramento de uma grande variedade de aplicações. Inicialmente, essas aplicações eram desenvolvidas (e uma grande parte delas ainda é) usando o protocolo HTTP. No entanto, tecnologias *Web Services* vêm se tornando a escolha de um número cada vez maior de aplicações que são providas através da Web, motivada por suas características de interoperabilidade, as quais permitem que aplicações diferentes interajam e executem sob uma variedade de plataformas e arcabouços.

Enquanto o número e a variedade das aplicações aumentam, nós podemos também observar que novos requisitos funcionais e não-funcionais precisam ser satisfeitos como a execução de tarefas de computação intensiva, manipulação de grandes volumes de dados e visualização de padrões de informação cada vez mais complexos. Ao mesmo tempo, métricas tradicionais de desempenho Web tal como tempo de resposta ao usuário devem ser mantidas em níveis aceitáveis. As exigências e as características destas novas aplicações baseadas na Web, assim como o comportamento de seus usuários, precisam ser caracterizados e compreendidos de modo que se possa lidar com as demandas e as expectativas desses usuários.

Um exemplo de tal aplicação é a mineração dos dados. Da perspectiva da aplicação, a mineração de dados tem como objetivo extrair informações úteis de um grande conjunto de dados. Um exemplo de tal informação são as regras de associação, isto é, relacionamentos causais entre elementos de dados. Aplicações canônicas de regras de associação são usualmente conhecidas como "análise de cesta de compras", que busca encontrar correlações entre os artigos comprados por clientes em um supermercado ou em qualquer outra loja. Um exemplo de regra de associação para esse contexto pode ser "70% dos clientes que compram leite compram também cereal". Estas informações podem ser muito relevantes para fins de suporte à decisão

nos mais diversos campos. Por exemplo, as companhias de cartão de crédito podem usar regras da associação para detectar fraudes, um supermercado pode usá-las para maximizar seu lucro ou otimizar o posicionamento de seus produtos nas prateleiras.

Os algoritmos de mineração de dados, que são o núcleo dessas aplicações, são, em geral, computacionalmente intensivos e exigem um espaço de armazenamento bastante significativo, em conseqüência dos dados de entrada, que são geralmente volumosos, e dos dados que são gerados durante sua execução, incluindo os resultados finais. Além disso, a informação extraída por esses algoritmos é geralmente complexa, sendo um campo ativo de pesquisa o projeto e a implantação de metáforas visuais que ajudam usuários a compreender as informações mineradas. O uso de um serviço Web de mineração de dados é caracterizado pela interação típica de aplicações Web, que segue um padrão de requisição-resposta, e também por requisições de execução de tarefas de mineração de dados, que são essencialmente assíncronas, se assemelhando a cargas em lote (*batch*), que podem ter longa duração. Em conseqüência destas características, os usuários desses sistemas comportam-se diferentemente, misturando interatividade e requisições em lote. Além disso, os usuários aprendem também como extrair melhor e mais rapidamente a informação do sistema, à medida que o usam. Em conseqüência, a carga do sistema como um todo é significativamente diferente das aplicações Web tradicionais e deve-se, assim, compreendê-la com o objetivo de melhorar o projeto dessas aplicações assim como a escalabilidade[1] das mesmas.

Neste trabalho apresentamos uma metodologia hierárquica de caracterização de serviços de mineração de dados providos pela Internet. Além disso apresentamos uma caracterização detalhada de um serviço desse tipo. Pelo conhecimento que temos, é a primeira caracterização de serviços desta natureza, isto é, um serviço Web que combine requisições interativas e em lote. Embora nosso estudo focalize uma aplicação específica, a metodologia que nós propomos é aplicável a qualquer tipo de serviço Web que apresente características similares. Nós aplicamos nossa metodologia para caracterizar a carga de um serviço de mineração de dados real, o Tamanduá [Tam05], que possuía 3 conjuntos de dados disponíveis para mineração de dados e um total de 145 usuários registrados. A coleta dos dados foi feita no período de Abril a Maio de 2005, quando houve usuários submetendo tarefas de mineração, visualizando resultados e avaliando o sistema.

[1]Entende-se por escalabilidade a capacidade que o serviço possui de manter a qualidade de atendimento, mesmo que a carga de trabalho a que ele está submetido aumente.

1.1 Conceitos

A Mineração de dados pode ser definida como a "extração de informações implícitas, previamente desconhecidas e potencialmente úteis de grandes bases de dados" [WF00]. Trata-se de uma etapa de um processo muito mais amplo conhecido como Descoberta de Conhecimentos em Banco de Dados, ou simplesmente KDD (*Knowledge Discovery in Databases*) [HK00]. Este processo é constituído de cinco passos principais (Figura 1.1) que podem ser agrupados em três etapas: preparação dos dados composta dos passos de seleção, preprocessamento e transformação; descoberta de padrões composta do passo de mineração de dados; visualização composta do passo de avaliação/interpretação de resultados.

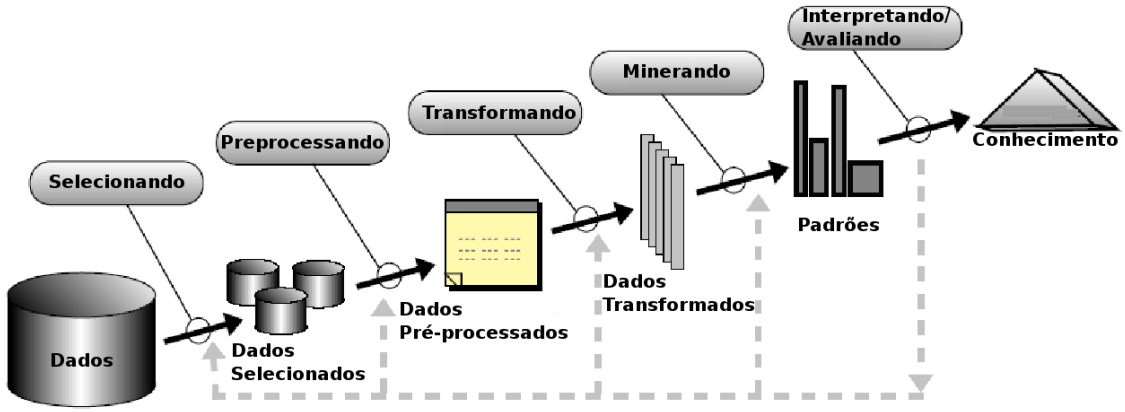


Figura 1.1: Descoberta de conhecimento em bancos de dados.

A primeira etapa do processo KDD é a etapa de preparação dos dados. Esta etapa inclui diversas tarefas, como as tarefas de limpeza dos dados (para remoção de inconsistências, eliminação de ruídos e tratamento de valores ausentes), integração de dados (para combinação de dados de diferentes fontes), seleção de atributos (para escolha dos dados mais relevantes para mineração) e transformação de dados (onde técnicas como discretização, normalização ou agregação são aplicadas sobre os dados para tornar a sua forma mais conveniente). A segunda etapa do processo KDD é a etapa de mineração de dados propriamente dita. Esta etapa consiste de tarefas que podem levar à descoberta de diferentes tipos de padrões. Dentre essas tarefas, destacam-se três tipos: análise de associações, que busca encontrar correlações entre itens de um conjunto de dados; análise de agrupamentos, que busca organizar os itens de uma base de dados em conjuntos, de acordo com algum critério de similaridade; e classificação, que consiste na descoberta de modelos que descrevam as características de determinadas classes de dados, de tal forma que um novo dado, cuja classe não

seja conhecida, possa ser classificado com base nas suas características conhecidas, através da utilização dos modelos.

A terceira etapa do processo KDD é a apresentação para os usuários dos resultados obtidos na mineração. Nesta etapa, são utilizadas metáforas visuais para representação dos padrões minerados de forma a possibilitar uma interpretação clara e inequívoca do conhecimento descoberto. São oferecidos também mecanismos que permitem ao usuário interagir com os resultados, filtrando os mais relevantes, explorando os relacionamentos existentes entre eles e detalhando aqueles que forem do seu interesse [FGW02, SD02].

Cada uma dessas etapas pode ser vista como uma unidade de trabalho independente, podendo ser executada em momentos e locais distintos. Um sistema que implementasse todas essas etapas pode exigir investimentos muito altos, em geral maiores do que aqueles que as organizações podem ou estão dispostas a fazer. No entanto, uma divisão de esforços, onde instituições distintas se especializem em diferentes etapas do processo e disponibilizem as soluções desenvolvidas na forma de serviços, de maneira que outras organizações, com necessidades semelhantes, possam usufruir delas, minimiza esse problema. Para que essa divisão possa acontecer, é necessário que se use uma arquitetura definida usando padrões abertos, possibilitando assim a interoperabilidade [2] entre os vários componentes. Dessa forma, associado ao uso amplo que a Web vem apresentando nos últimos anos, tecnologias *Web Services* se apresentam como uma boa solução para prover cada uma das etapas do KDD como serviços.

Web services são a mais recente evolução nos padrões de desenvolvimento de aplicações distribuídas permitindo que aplicações cooperem facilmente e compartilhem informações e dados umas com as outras. Tecnicamente, *Web services* são serviços distribuídos que processam mensagens *SOAP* (*Simple Object Access Protocol*) codificadas em XML, enviadas através de HTTP e que são descritas através de *WSDL* (*Web Services Description Language*) e que usam tecnologias programáveis e reutilizáveis que aproveitam a flexibilidade da Internet [FF03]. Com eles é possível ter uma infinidade de aplicativos conectados em rede, mesmo rodando em plataformas diferentes, fornecendo informações a todos os seus clientes, parceiros e funcionários. As tecnologias *Web Services* tentam fazer a comunicação entre aplicações tão fácil quanto a Web tornou a publicação de documentos [Shi02]. Através dessas

[2] Interoperabilidade pode ser definida como capacidade de compartilhar e trocar informações e processos entre ambientes heterogêneos, autônomos e distribuídos, de tal forma que um sistema (informatizado ou não) possa se comunicar de forma transparente (ou o mais próximo disso) com outro sistema (semelhante ou não).

tecnologias é possível a criação de aplicações distribuídas, com vários componentes espalhados pela Internet, trabalhando em cooperação para a realização de tarefas.

Uma das principais razões para o crescente interesse em tecnologias *Web services* é que eles permitem uma Arquitetura Orientada a Serviços *SOA* (*Service-Oriented Architecture*). Quando se utiliza *Web services* para a construção de tal arquitetura, as soluções consistem de coleções de serviços autônomos, identificados por URLs, com interfaces documentadas através de WSDL e processando mensagens XML. SOA é um complemento a abordagens orientadas a objetos e procedimentais.

1.2 Motivação

A mineração de dados oferecida como *Web Service* trata-se de uma estratégia nova que possui algumas características em comum com aplicações Web tradicionais como comércio eletrônico, bibliotecas digitais, vídeo sob demanda, aprendizado à distância etc. Em todos esses serviços podemos observar uma variabilidade nos perfis de usuários, que em diferentes sessões[3], realizam requisições de custo variado. Além disso, essas aplicações podem ser caracterizadas por páginas Web complexas e geradas dinamicamente, com conteúdo personalizado e integradas a um sistema de banco de dados, ao mesmo tempo que satisfazem alguns requisitos de qualidade de serviço (por exemplo, bom desempenho e alta disponibilidade) [MA02].

No entanto, além dessas características já citadas, serviços de mineração de dados possuem outras características próprias que os diferenciam dos demais. Esse tipo de serviço, apesar de possuir alto grau de interatividade com os usuários, também possui processamento em lote que são representados pelas tarefas de mineração de dados propriamente dita. Essas tarefas, caracterizadas por um algoritmo, por uma base de dados e por parâmetros utilizados nos mesmos, são normalmente intensivas computacionalmente.

Sistemas dessa natureza têm crescido em uso e aplicabilidade, como conseqüência natural da popularização de serviços Web e do seu crescente uso como canal de acesso a serviços computacionalmente intensivos. O entendimento de como usuários interagem com tais sistemas em termos de sua atuação e do uso conjunto dos componentes interativos (síncronos) e em lote (assíncronos) ainda está em aberto.

Entender a aleatoriedade associada ao modo como os usuários solicitam cada serviço dessas aplicações, assim como a carga gerada por eles, se torna uma tarefa indispensável na análise da viabilidade da estratégia e na busca de soluções para

[3]sessões são caracterizadas por um conjunto de requisições feitas por um determinado usuário, durante uma interação com um serviço Web, em um período delimitado

melhora de desempenho e disponibilidade para tornar o sistema escalável. Uma ferramenta importante que provê esse entendimento é a caracterização do comportamento dos usuários e da carga gerada por eles. Entretanto, por se tratar de uma estratégia nova para o provimento de serviços Web, com características próprias, caracterizações tradicionais como as apresentadas em [AKR01], [Bor83] e [DKB02] não são apropriadas. Este cenário motivou-nos a desenvolver uma metodologia de caracterização mais abrangente, que contemple não somente as características comuns a outras aplicações Web, mas também as características específicas dessas aplicações, trabalho este inédito até então.

Através da aplicação dessa metodologia de caracterização, será possível fazer uma análise de como os usuários reagem ao comportamento síncrono ou assíncrono do sistema, entendendo como suas requisições se relacionam em termos de similaridades e diferenças. Além disso, através dela será possível prover subsídios para geração de cargas sintéticas fiéis ao comportamento do usuário, além de desenvolver um modelo de previsão de desempenho e disponibilidade voltado para o gerenciamento e planejamento de capacidade desses serviços.

Apesar da metodologia ter sido desenvolvida focada em serviços de mineração de dados, ela pode ser muito bem aplicada na caracterização de outras aplicações que possuam características similares às aplicações de mineração de dados como aplicações de compartilhamento remoto e distribuído de informações [TKC+04, AEM+04, BMPZ02, DCJ+05], de análise de risco em aplicações financeiras [Col05], ambientes de pesquisas de arqueologia virtual e de diagnóstico climático distribuído [Ser05], de microscopia virtual [ABB+98], de análise de placentas [PH05] dentre outros.

1.3 Objetivo

O objetivo deste trabalho é apresentar uma metodologia de caracterização de serviços de mineração de dados, elaborada com base em metodologias tradicionais de caracterização Web e nas características específicas desses serviços como requisições computacionalmente intensivas (em lote), requisições interativas e a sobreposição entre elas.

1.4 Descrição

Este trabalho apresenta, portanto, uma metodologia para caracterizar serviços de mineração de dados de forma hierárquica, cujos níveis são apresentados a seguir:

- **Sessões de Usuários**: Esse nível de nossa caracterização tenta modelar o padrão de acesso dos usuários do sistema com o objetivo de encontrar modelos que representem informações temporais relacionadas à carga criada pelos usuários, como eles iniciam as sessões e quão longa é a duração das mesmas.

- **Comportamento de Usuário**: Nesse nível de nossa metodologia caracterizamos o padrão das requisições dos usuários. Isso significa entender o comportamento dos usuários em termos das requisições dos mesmos a cada uma das funcionalidades do sistema e como essas requisições estão relacionadas durante uma sessão, como por exemplo a sobreposição entre as requisições interativas e não interativas.

- **Tarefas dos Usuários**: Caracterizado o padrão de acesso dos usuários e o padrão de suas requisições, nesse nível de nossa metodologia caracterizamos e modelamos a natureza das requisições. Em mineração de dados, requisições importantes são as execuções de algoritmos de mineração de dados, ou tarefas.

Para a avaliação da metodologia, utilizaremos a plataforma do projeto Tamanduá [Tam05], que é uma arquitetura para a distribuição do processo KDD, onde repositórios de dados, servidores de mineração e servidores de visualização são disponibilizados por diferentes instituições e integrados através das tecnologias *Web Services* para a realização de tarefas específicas. No capítulo 4 é apresentada a arquitetura do sistema de forma mais detalhada.

1.5 Organização

O trabalho está organizado em mais quatro capítulos além desta introdução. O capítulo 2 apresenta um estudo feito com trabalhos de caracterização e avaliação de comportamento de usuários, além de outros trabalhos relacionados a serviços de mineração e arquiteturas baseadas em tecnologias *Web Services*. O capítulo 3 apresenta a metodologia de caracterização proposta neste trabalho. No capítulo 4 apresentamos um estudo de caso aplicando a metodologia proposta em um sistema real de serviços de mineração de dados, Tamanduá. Nesse capítulo descrevemos a arquitetura Tamanduá de forma detalhada, e apresentamos o resultados obtidos na aplicação da metodologia nessa arquitetura; por fim no capítulo 5 apresentamos as conclusões do trabalho, além de alguns potenciais trabalhos futuros.

Capítulo 2

Trabalhos Relacionados

Este capítulo apresenta uma síntese dos principais trabalhos relacionados a serviços de mineração de dados e arquiteturas baseadas em tecnologias *Web Services*. Apresentamos, dessa forma, uma coletânea de artigos científicos publicados em simpósios e conferências, além de artigos técnicos e capítulos de livros relacionados a caracterização e planejamento de capacidade de aplicações Web tradicionais que serviram de base para a realização deste trabalho. No entanto, por se tratar de uma abordagem recente, não foram encontrados trabalhos relacionados à caracterização de serviços de mineração de dados ou de outros serviços com características similares.

Dessa forma dividimos os trabalhos em três grupos. A Seção 2.1 descreve algumas aplicações Web que vêm fazendo uso de tecnologias *Web Services* além de algumas avaliações de aspectos no uso dessas tecnologias que devem ser considerados; a Seção 2.2 apresenta os trabalhos relacionados a serviços de mineração de dados; e, finalmente na Seção 2.3 apresentamos os esforços de pesquisa na caracterização da interação entre usuários e serviços Web e nas formas de extração de informações do uso do sistema para fins de caracterização. Além disso, discutimos a caracterização de cargas de trabalho associadas com a execução de tarefas de mineração.

2.1 Web Services

Christopher Ferris *et al.* [FF03] define *Web Service* como sendo uma aplicação de software identificada por uma URI, cujas as interfaces e ligações sejam capazes de serem definidas, descritas, e descobertas como artefatos XML, suportando assim interações diretas com outros agentes de software usando troca de mensagens baseadas em XML através dos protocolos de Internet. Em complemento a essa definição, para Almeida e Menascé [AM02], *Web Service* descreve funcionalidades de negócio específicas expostas por uma companhia, usualmente através de uma conexão In-

ternet, com o propósito de prover um mecanismo para que outras companhias ou programas usem ou complementem o serviço. Já em [MA02, MA00] esses mesmos autores discutem detalhadamente questões relacionadas à qualidade de serviço em todos os tipos de serviços Web, usando uma abordagem quantitativa, além de prover um arcabouço para entender as complexas relações da Web e como isso impacta no desempenho e disponibilidade dos serviços.

Já existem hoje diversos trabalhos que utilizam tecnologias *Web Services* [TKC+04, AEM+04, BMPZ02, DCJ+05]. Em [TKC+04], Kerry L. Taylor *et al.*, apresenta uma arquitetura baseada em *Web Services* para prover uma nova plataforma para facilitar pesquisas na área de saúde denominada de HRDN (*Health Research Data Network*). Essa plataforma foi criada com o intuito de compartilhar e analisar dados distribuídos de forma segura. Um trabalho semelhante é apresentado em [AEM+04]. Neste artigo os autores apresentam uma arquitetura de gerência e colaboração distribuída de dados relacionados a diagnósticos de câncer de mama, uma vez que estes possuem um alto grau de variabilidade dentro da população. Essa arquitetura também é baseada em tecnologias *Web Services*. Ainda na direção de compartilhamento de informação, um trabalho bastante interessante é apresentado por Baglietto *et al.* [BMPZ02]. Nesse trabalho os autores descrevem uma arquitetura também baseada em *Web Services* que possibilita o compartilhamento de informações de negócios de uma mesma área. Além disso os autores apresentam um estudo de caso real de uso da arquitetura no Porto de Gênova. E, por fim, em [DCJ+05] os autores apresentam um sistema colaborativo entre participantes remotos em uma tele-imersão construído utilizando *Web Services* e *Microsoft Visual Studio .Net*.

Outra proposta que utiliza tecnologias *Web Service* é descrita em [Col05]. Nesse trabalho, o autor apresenta uma aplicação na qual processos computacionalmente intensivos e paralelos, relacionados a análise de risco de mercado, são gerenciados através de um *front-end* Web. Outros trabalhos relacionados ao uso de tecnologias *Web Service* como solução podem ser encontrados em [Ser05], onde temos desde propostas de compartilhamento de dados até gerência de processos computacionalmente intensivos, como por exemplo um ambiente de pesquisa arqueológica virtual e um sistema de diagnóstico climático distribuído, dentre outros.

Além de propostas de arquiteturas baseadas em tecnologias *Web Services*, existe na literatura um conjunto muito grande de trabalhos que avaliam alguns dos aspectos relacionados a esse modelo [GSC+00, CGB02, GSC+04, AGLG04, Mic02, DP02, KS03, SCP04]. Em [GSC+00], os autores discutem o desempenho dos protocolos de comunicação entre serviços remotos para computação científica, mais especificamente o *SOAP* (*Simple Object Access Protocol*), muito comumente utilizado em outros

contextos. Através de experimentos os autores mostram que o *SOAP*, por si só, não é eficiente o bastante para aplicações científicas de larga escala, mas, quando usado em conjunto com um arcabouço de multi-protocolos de chamada remota, sua eficiência melhora significativamente, podendo ser utilizado como um protocolo de controle universal.

Um trabalho semelhante é apresentado em [CGB02], no qual os autores investigam as limitações de algumas tecnologias *Web Services* em computação científica. Nesse trabalho os autores detectam quais são as deficiências que limitam o *SOAP* em termos de desempenho em aplicações científicas de computação intensiva, além de implementarem um nova versão de melhor desempenho para o mesmo. Em [AGLG04], os autores também apresentam um proposta de como melhorar o desempenho do SOAP.

Ainda com relação à avaliação de tecnologias *Web Services*, em [GSC+04] os autores comparam e contrastam algumas das ferramentas mais populares para implementação de arquiteturas baseadas em *Web Services*, provendo como resultado diretrizes para o desenvolvimento de projetos nessas arquiteturas com melhor desempenho. Neste trabalho são avaliadas o conjunto de ferramentas de implementação *SOAP* mais utilizadas recentemente, como *gSOAP* $_{2.4}$, *AxisC++ CVS* $_{May28}$, *AxisJava* $_{1.2}$, *.NET* $_{1.1.4322}$ e *XSOAP4/XSUL* $_{1.1}$. Um trabalho idêntico é apresentado em [Mic02, DP02], onde, no entanto, as implementações de *SOAP* avaliadas são JavaRMI, CORBA, HTTP. Outros trabalhos como [KS03, SCP04] também apresentam comparações entre outras tecnologias de implementação de *Web Services* muito utilizadas atualmente.

Além desses trabalhos, temos um artigo em [Dim05] que explica como tecnologias *Web Services* estão se tornando populares, ou seja, padrão para o desenvolvimento de aplicações que gerenciam processos computacionalmente intensivos e um Workshop interno da Microsoft [Wor05] que discute tecnologias para aplicações de computação intensiva de dados usando *Web Services*

Em suma, a diversidade e aplicabilidade de tecnologias *Web Services* é muito ampla e crescente, sendo para nós uma motivação para a realização de estudos de caracterização de carga de trabalho nesses ambientes.

Nesta Seção apresentamos um estudo de aplicações implementadas com base nas tecnologias *Web Service* existentes, assim como um estudo de avaliações dessas aplicações. Esses estudos foram feitos com o intuito de entender como essas tecnologias vêm sendo utilizadas nos mais diversos contextos, destacando a importância dessas no cenário atual de desenvolvimento de aplicativos. Além disso, entender a arquitetura dessas aplicações é uma etapa essencial neste trabalho, uma vez que nosso

objetivo é propor uma metodologia de caracterização de serviços de mineração de dados baseados nessas arquiteturas.

2.2 Serviços de Mineração de Dados

Existem alguns trabalhos relacionados à disponibilização de serviços de mineração de dados como serviços na Internet. Uma das primeiras propostas foi apresentada por Sarawagi *et. al.* em [SN00], no qual alguns desafios no provimento desse tipo de serviços são discutidos, tais como padrões para descrição de modelos de mineração, integração de modelos de origens distintas, segurança e confiabilidade, e personalização usando dados do usuário.

Já Krishnaswamy *et al.* [KZL01b, KZL01a] compararam diversos provedores de serviços de mineração de dados baseados na Internet, como *digiMine* [dig02] e *Information Discovery* [Dis02]. Usuários submetem as requisições de mineração enviando seus dados ao provedor de serviços (com quem normalmente a empresa possui um contrato) e acessam os resultados através de uma interface Web. Após discutirem situações de uso desses serviços e questões relacionadas a cada um, propuseram um modelo de múltiplos provedores de serviço, definindo os protocolos de interação e um mecanismo de troca de informações e descrição de requisição por tarefa baseado em XML. Além desse trabalho, Krishnaswamy apresenta em [KZL03] um modelo para mineração de dados baseado em agentes. Resumidamente, nessa proposta são os agentes que vão até o local onde os dados estão armazenados, que é o mesmo onde a mineração é feita.

Além dos trabalhos discutidos acima, existe ainda um conjunto grande de propostas de serviços de mineração de dados utilizando grades computacionais (*grids*) [MHHK99, CTT01, CTT02, CT03, BHTW03b, BHTW03a, KBTH04]. Em cada um desses trabalhos são apresentadas propostas de utilização de grades computacionais, discutindo questões relacionadas a infraestrutura necessária, arquiteturas e desempenho das mesmas. Em [MHHK99] temos um dos primeiros relatos de uma implementação distribuída de um algoritmo de mineração de dados. Nesse trabalho os autores implementaram uma versão paralela de um algoritmo de agrupamento sobre vários ambientes geograficamente distribuídos.

Nos trabalhos [CTT01, CTT02, CT03] encontramos as primeiras publicações sobre infraestrutura para serviços de mineração de dados sobre *grids*. Nesses trabalhos, os autores apresentam uma arquitetura de software para aplicações de descoberta de conhecimento distribuídas e paralelas denominada *Knowledge Grid*, construída sobre um conjunto básico de serviços normalmente oferecidos pelas plataformas de

grid. Apesar de apresentar o ambiente, os autores não apresentam nenhum caso real de uso do mesmo.

Em [BHTW03b, BHTW03a, KBTH04, VJF+04] encontramos trabalhos mais recentes sobre mineração de dados em ambientes distribuídos. Peter Brezany *et al.* em [BHTW03b] apresentam os princípios para projetos de serviços de mineração de dados de alto desempenho em ambientes *grid*, além de apresentar um desses serviços. Os mesmos autores apresentam em [BHTW03a] o *GridMiner*, uma infraestrutura para mineração de dados em *grid*, baseada nos princípios apresentados no artigo anterior([BHTW03b]). Uma infraestrutura bastante semelhante a dos artigos anteriores é proposta em [KBTH04].

Já em [VJF+04], os autores apresentam uma implementação altamente escalável de um algoritmo de mineração de regras de associação. Esse algoritmo foi implementado para ser executado de forma paralela em *clusters*, baseado no paradigma *Filter Labeled-Stream*, descrito no próprio artigo.

Em resumo, podemos observar que mineração de dados é uma aplicação que vem sendo bastante estudada, apesar de não termos encontrado nenhum registro de uso em operação real delas. A validação dessas aplicações ainda é um problema em aberto, investiu-se muito no desenvolvimento dessas aplicações mas pouco se sabe sobre os usuários realmente usariam as arquiteturas propostas, como seria seu comportamento, sendo mais uma motivação para a realização desse trabalho.

Apesar de não fazer parte desse trabalho propor qualquer outra arquitetura para o provimento de serviços de mineração de dados, conhecer as diversas arquiteturas e implementações desses serviços existentes é fundamental para entender as características e os requisitos dos mesmos, e assim propor uma metodologia de caracterização desses serviços. De qualquer forma, foram trabalhos fundamentais para a proposta de metodologia apresentada neste trabalho.

2.3 Caracterizações Web

Existem diversas frentes de pesquisa relacionadas com a caracterização de serviços Web [AKR01, Bor83, AJ99, AW97]. Em [Bor83], os autores apresentam a caracterização de comportamento de usuários de um sistema de consultas on-line a uma biblioteca universitária nos Estados Unidos. Métricas como tipo de consultas realizadas, padrões de uso do sistema, tempo gasto para execução de tarefas foram quantificadas no trabalho. Um estudo da escalabilidade de um sistema de compras Web é apresentado em [AKR01]. Esse estudo foi feito com base na caracterização da carga gerada pelos usuários no sistema, utilizando métricas como popularidade

de serviços e produtos, taxa de requisições e requisições por cliente. Um estudo semelhante é apresentado em [AJ99], porém relacionado ao provimento de vídeos ao vivo pela Internet, no qual os autores fazem um estudo da carga gerada pelos usuários em um Web Site que oferecia a transmissão dos jogos da copa do mundo de futebol de 1998 ao vivo. Também em [AW97] é apresentada uma caracterização de carga de aplicações em três ambientes distintos: ambiente acadêmico, ambiente organizacional de pesquisa científica e ambiente comercial. Em todos esses trabalhos acima mencionados, assim como em [Pit98, DKB02], as métricas utilizadas por eles foram utilizadas em nossa caracterização nos níveis de sessão e tarefa de nossa metodologia. Entretanto, essas métricas não são suficientes para esse trabalho, uma vez que o critério de interação usuário-sistema e seus diferentes tipos de requisição (interativa e em lote) não são tratadas.

No artigo [JvO04] é apresentado um método de extração de informações relevantes sobre o comportamento de navegação dos usuários em serviços Web. Além desse artigo, em [JvO04] é apresentado um modelo de previsão de comportamento de usuário. Em outros dois artigos [CJ90, RMP90] são apresentadas propostas de aquisição de conhecimento. Apesar de não fazer parte do nosso trabalho propor qualquer método de previsão de comportamento ou de aquisição de conhecimento, as técnicas utilizadas nesses dois artigos, de certa forma, foram úteis para extrair informações do uso do sistema em nossa caracterização, além de servir como base para identificação de refinamento entre as tarefas requisitadas pelos usuários.

Wasserman et. al. [WMSR04] apresentaram uma caracterização de carga de inteligência de negócios baseados no *benchmark* TPC-H . O *benchmark* TPC-H consiste de um conjunto de consultas que imitam aplicações complexas de análise de negócios, em particular a atividade de um fornecedor por atacado. Ela não representa a atividade de nenhum segmento em particular, mas sim o de uma empresa que deve controlar, vender ou distribuir um produto mundialmente. Eles consideraram 22 consultas do *benchmark* e as agruparam em quatro grupos, baseados na demandas das mesmas por processamento e entrada e saída, em particular caracterizando a demanda de suas bases de dados. O processo de agrupamento é baseado em um trabalho anterior de Calzarossa e Serazzi [CS94]. Sua suposição é que a execução de uma aplicação pode ser expressa em termos da quantidade e dos SLAs de cada um dos tipos de consultas. Sua proposta é diferente da nossa no sentido que eles não focalizam a interação dos usuários com o sistema, mas assumem que o *benchmark* representa a carga que os usuários reais gerariam e avaliam a demandas correspondente.

Outra fonte importante que também serviu como base para nosso trabalho são

[MA02, MA00], que tratam de questões relacionadas a planejamento de capacidade e a importância disso para o provimento de um serviço escalável e de boa qualidade, alto desempenho e boa disponibilidade, além de apresentarem um modelo de representação de padrão de comportamento de usuário denominado CBMG(*Customer Behavior Model Graph*) utilizado em vários contextos [HTMRG+04, NAR+04], que serviu como ponto de partida para a criação do modelo de representação utilizado em nosso trabalho no nível de comportamento de usuário denominado USIMG(*User-System Interaction Model Graph*) que será apresentado na Seção 3.4.3.

Apesar dos esforços acima mencionados, nós não conseguimos encontrar nenhum trabalho que alvejasse a caracterização de serviços de mineração de dados ou de um sistema Web que apresentasse características similares tais como tarefas computacionais intensivas e assíncronas. De qualquer forma, os trabalhos apresentados nesse capítulo foram fundamentais na proposta de metodologia que é apresentada a seguir.

Capítulo 3

Metodologia

Neste capítulo descrevemos a metodologia de caracterização proposta neste trabalho. Nossa metodologia tem por objetivo guiar a execução de caracterizações de sistemas que possuem características interativas e não-interativas, a fim de analisar o comportamento dos usuários desses sistemas, quantificando, qualificando e modelando a carga criada por eles.

Antes de apresentarmos a metodologia de caracterização propriamente dita, é importante definirmos a arquitetura do sistema que pretendemos caracterizar, ou seja, definir os componentes da arquitetura e os mecanismos de interação entre esses componentes. Além disso, descrevemos quais são os requisitos de caracterização desses sistemas e como coletar e preparar os dados necessários para essa caracterização. Essas informações são apresentadas nas seções seguintes, seguidas da metodologia propriamente dita.

3.1 Arquitetura dos Sistemas Caracterizados

Os sistemas em foco neste trabalho são uma especialização de sistemas provedores de serviços Web tradicionais no sentido de que o usuário acessa todos os recursos do sistema utilizando recursos da Web. Entretanto, esses sistemas se distinguem de serviços Web tradicionais por possuir um ou mais componentes assíncronos que normalmente executam requisições computacionalmente intensivas, à semelhança de serviços em lote em computadores de grande porte. Um bom exemplo desse tipo de sistema são os serviços de mineração de dados.

A arquitetura dos sistemas objeto desta metodologia de caracterização é composta basicamente de 3 componentes (Figura 3.1):

- **Usuários:** é o conjunto de usuários que requisitam os serviços providos pelos servidores;

- **Servidores Interativos:** um ou mais servidores de serviços Web tradicionais que provêem serviços síncronos baseados no protocolo HTTP ou outro protocolo similar, e que servem de entreposto para as requisições assíncronas aos servidores em lote, assim como o acesso aos seus resultados, ou seja, são os responsáveis pela intermediação entre os usuários e os servidores em lote; e
- **Servidores em Lote:** um ou mais servidores de processamento de tarefas computacionalmente intensivas como algoritmos de mineração de dados.

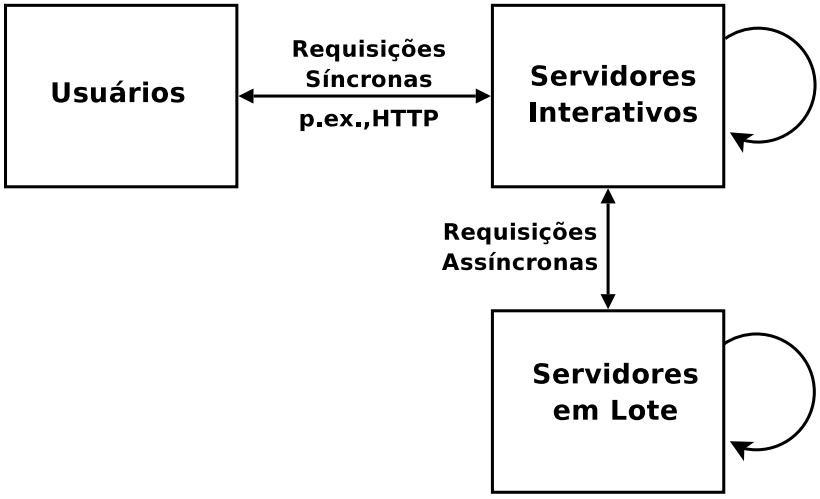


Figura 3.1: Arquitetura do Sistema Objeto de Caracterização

Neste contexto, os serviços providos pelos servidores interativos são síncronos, ou seja, os usuários submetem requisições aos servidores e aguardam uma resposta dos mesmos para então submeter uma nova requisição. A carga de trabalho observada no sistemas interativos é conseqüência da interatividade dos usuários, ou seja, ela é função de variáveis como a freqüência de submissão de requisições, da natureza dessas requisições e do custo de cada uma delas.

Ao incorporarmos componentes assíncronos, representados pelos servidores em lote, há duas mudanças fundamentais que devem ser levadas em consideração ao se analisar a carga. Primeiro, o comportamento dos usuários deixa de ser diretamente afetado pela latência das requisições que geralmente não é alta no caso de serviços Web tradicionais, ou seja, o usuário não precisa mais esperar a resposta de uma requisição para efetuar outra, várias requisições podem ser disparadas em paralelo pelos usuários podendo haver assim sobreposição temporal entre vários processos em lote. Segundo, a carga assíncrona gerada pelos usuários é independente da interação e sim influenciada pela demanda do usuário por ações assíncronas, ou seja, o

usuário pode continuar interagindo com o sistema enquanto uma determinada tarefa é executada nos servidores em lote, de tal forma que um dado usuário pode estar submetendo requisições interativas e em lote que serão satisfeitas por servidores diferentes simultaneamente. Nesse segundo caso a carga também poderá ser fortemente influenciada pela natureza dos resultados obtidos, ou seja, caso os resultados encontrados não sejam satisfatórios, novas requisições serão submetidas aos servidores em lote.

Dessa forma, temos que o funcionamento desses sistemas pode ser dividido, de forma simplificada, em três mecanismos de interação descritos abaixo:

- **Requisição-Resposta Síncrona:** o usuário submete uma requisição a um servidor interativo e normalmente espera a resposta para submeter uma nova requisição. Ou seja, corresponde a interações típicas de sistemas WEB tradicionais;
- **Requisição Assíncrona:** o usuário requisita a execução de uma tarefa assíncrona a um servidor interativo que a repassa ao servidor em lote correspondente. Em geral trata-se de uma solicitação computacionalmente intensiva que possivelmente irá demorar algum tempo (minutos, horas ou até mesmo dias) para ser finalizada; e
- **Resposta Assíncrona:** o usuário acessa a resposta a uma requisição assíncrona feita em algum momento no passado, que é armazenada pelo sistema quando da sua conclusão. A qualquer momento durante a utilização interativa do sistema, o usuário pode acessar a resposta de uma requisição concluída.

Dada a descrição da arquitetura do sistema que pretendemos caracterizar, com seus componentes e mecanismos de interação, devemos definir as dimensões da caracterização. A descrição dessas dimensões é apresentada na Seção a seguir.

3.2 Dimensões de Caracterização

Uma metodologia de caracterização de comportamento de usuários dos serviços descritos na Seção 3.1 deve tratar três tipos de requisitos: padrão de acesso dos usuários; padrão das requisições dos usuários; e a natureza dessas requisições. Assim a metodologia deve considerar tanto os componentes interativos quanto não-interativos da arquitetura, além do relacionamento entre eles. É a partir da caracterização de cada um desses componentes que se torna possível obter todas as informações necessárias para a modelagem da carga imposta pelos usuários.

A dimensão padrão de acesso refere-se à modelagem das informações temporais de carga dos usuários, ou seja, visa determinar a freqüência das interações entre usuários e sistemas e determinar o quanto cada uma dessas interação dura. Essa informação é obtida a partir dos serviços interativos, pois são eles os responsáveis por toda interação do sistema com os usuários

O padrão de requisições dos usuários é a dimensão que se refere à modelagem do que o usuário faz enquanto está interagindo com o sistema, ou seja, quais são as funcionalidades do sistema solicitadas e como essas solicitações são feitas ao longo da sessão de interação usuário-sistema. As funcionalidades requisitadas podem ser tanto síncronas quanto assíncronas e as mesmas podem estar sobrepostas em uma mesma sessão. Para a obtenção dessas informações, os sistemas interativos e em lote, ou seja, os componentes interativos e não-interativos, devem ser analisados em conjunto.

A terceira e última dimensão se refere à análise da natureza das requisições dos usuários, ou seja, modelar os detalhes das requisições assíncronas dos usuários. Nesse tipo de serviço, as tarefas mais importantes a serem analisadas são as assíncronas por serem tarefas que normalmente demandam mais tempo e processamento. No entanto, para caracterizar essas tarefas deve-se antes entender suas características. As informações necessárias para a caracterização dessa dimensão são obtidas a partir dos registros de atividade dos servidores em lotes.

Determinadas quais as dimensões de caracterização que devem ser consideradas pela metodologia, a etapa seguinte é descrever como deve ser feita a coleta dos dados necessários para essa caracterização. Dessa forma, a Seção a seguir apresenta detalhadamente como essa coleta deve ser feita.

3.3 Coleta de Dados

A coleta de dados é feita de forma descentralizada, ou seja, cada um dos sistemas que compõem a arquitetura é responsável pelo registro das requisições feitas a ele. Nos sistemas interativos os logs podem ser feitos a partir dos logs tradicionais gerados pelos servidores Web com algumas alterações para que as requisições a outros sistemas também sejam arquivadas, assim como a natureza delas. Já os sistemas em lote devem passar por um processo de instrumentação completa para que todas as requisições que cheguem a eles ou saiam deles seja registradas.

Em ambos os casos, a noção de sessão é registrada desde a coleta de dados. Uma sessão de usuário é uma seqüência contígua de requisições submetidas por um usuário, onde duas requisições simultâneas não estão espaçadas temporalmente acima de

um limiar pré-estabelecido. Esta noção é largamente empregada em sistemas Web e será estendida aqui no sentido de incorporar as requisições assíncronas, que serão consideradas como parte da sessão onde elas foram requisitadas, embora possam se estender por períodos de inatividade do usuário ou mesmo outras sessões. A justificativa para essa estratégia é que, apesar da sua dispersão temporal, o momento da requisição é o único no qual o usuário atua diretamente no ciclo de vida da requisição assíncrona.

Toda solicitação de um usuário é feita através dos sistemas interativos, assim os mesmos são os responsáveis pelo registro delas. Cada uma dessas requisições deve ser registrada contendo a identificação da sessão de origem, o dia, mês, ano, hora, minuto e segundo (a que denominamos *timestamp*) da requisição, além dos dados que caracterizam a natureza da requisição, por exemplo, no caso de uma solicitação de uma tarefa de mineração de dados, deve-se registrar qual o algoritmo utilizado, os parâmetros dos algoritmos escolhidos, além da base de dados e atributos que foram selecionados. Uma importante informação que deve ser registrada é o identificador do usuário associado à sessão de origem da tarefa no sistema como um todo.

Cada requisição de usuário pode fazer com que o sistema interativo gere uma ou mais requisições aos sistemas em lote. Cada vez que um sistema interativo gera uma solicitação a um sistema em lote, o mesmo deve registrar a mesma em seus logs. Além disso, o servidor em lote também deve fazer o registro de que foi acionado. No sistema interativo de onde originou-se a solicitação, o registro deve conter o identificador da sessão de origem, um identificador da requisição que a originou, o identificador do servidor de destino da requisição e o *timestamp*, além dos detalhes e parâmetros da requisição. No servidor em lote de destino deve-se registrar o identificador da sessão de origem, um identificador da requisição que a originou, o identificador do servidor de origem da requisição e o *timestamp*. Requisições entre os sistemas em lote também podem ocorrer e o registro dessas requisições devem ser feitas da mesma forma.

Para cada requisição submetida, seja dos usuários aos sistemas interativos, seja entre os sistemas que compõe a arquitetura (interativos e em lote), existe uma resposta associada, ou seja, toda requisição deve ter uma resposta. Da mesma forma que as requisições são registradas nos logs dos servidores correspondentes, as respostas a essas também devem ser registradas, onde os mesmos identificadores utilizados nos registros das requisições devem ser utilizados nas respectivas respostas.

Neste tipo de arquitetura, por se tratarem da integração de servidores interativos em lote, pode ocorrer que um determinada sessão de usuário seja finalizada, mesmo que ainda existam processos sendo executados nos servidores dos sistemas em lote.

Independente do fim de uma sessão, os processos continuam sendo executados até o fim dos mesmos. Assim os registros precisam continuar sendo feitos referenciando a sessão finalizada até que as requisições assíncronas estejam concluídas.

Nesse processo de coleta de dados, os identificadores exercem um papel fundamental pois serão eles os responsáveis em estabelecer as ligações entre logs de diferentes servidores, como veremos na Seção a seguir. Além disso, os relógios dos servidores que compõem cada um dos sistemas interativos ou em lote, devem estar sincronizados para manter a consistência temporal dos dados, como também será discutido a seguir.

3.3.1 Preparação dos Dados

Cada servidor da arquitetura, seja ele interativo ou em lote, possui um log (registro) das requisições feitas a ele, assim como as respostas a cada uma delas. Possui também registrados as requisições que ele faz aos outros servidores[1]. Assim, após a coleta dos dados, um passo fundamental é a sua consolidação, ou seja, a partir de logs coletados de forma independente e distribuída entre os servidores que compõem a arquitetura, devemos criar um log único que agrupe todos os registros.

O primeiro passo desse processo de agrupamento é realizar a verificação de integridade dos registros para que não haja inconsistências, ou seja, deve-se verificar se todo registro de requisição que é enviada de um sistema para outro possui o seu correspondente informando quando a requisição chegou no sistema de destino, se todo registro de resposta a uma requisição em um sistema possui o correspondente no sistema que originou a requisição etc. Essa verificação de integridade é feita através dos identificadores mencionados na Seção anterior, que permitem acompanhar toda a trajetória de uma requisição. Caso a integridade dos logs seja violada, os mesmos devem ser descartados.

Feita esse verificação de integridade, o próximo passo é o agrupamento propriamente dito, colocando todos os registros de todos os sistemas em um único arquivo e posteriormente ordenando esse arquivo único pelo *timestamp* dos registros. Daí vem a necessidade mencionada na Seção anterior de que os servidores de cada um dos sistemas devem estar com seus relógios sincronizados, caso contrário inconsistências podem ocorrer, como por exemplo, uma requisição chegar ao sistema de destino antes que a mesma tenha sido enviada pelo servidor de origem, ou o registro de duração de execução de uma tarefa ser menor ou maior do que realmente foi. Neste

[1] Embora estejamos os servidores interativos e em lote como componentes atômicos, é possível e usual que haja requisições entre servidores de mesma natureza, a exemplo de arquiteturas de servidores Web organizados em camadas.

último caso, a inconsistência poderia não ser percebida e levar a uma caracterização equivocada.

Outra exemplo importante de transtorno que uma dissicronização traria seria com relação a análise do comportamento, no qual os logs poderiam registrar que uma tarefa de um determinado usuário ficou menos tempo em execução do que realmente ficou.

Finalizado esse processo de preparação dos dados, o resultado é um único log que contém todos os registros de todos os sistemas, agrupados de forma consistente. Através desse log é possível obter todo o histórico de todas as requisições de todos usuários que submeteram requisições aos servidores de forma direta ou indireta. Esse é o log utilizado na extração dos dados necessários para a caracterização de carga do sistema.

3.4 Metodologia de Caracterização

Definida a arquitetura do sistema, as dimensões de caracterização, e a estratégia de coleta e preparação dos dados podemos apresentar a metodologia de caracterização propriamente dita. Apesar de existirem diversos sistemas que poderiam se encaixar perfeitamente no modelo de arquitetura apresentado, como microscopia virtual apresentada em [ABB+98] e análise de placenta [PH05], a definição de nossa metodologia está focada especificamente em sistemas de mineração de dados. No entanto, como veremos a seguir, a metodologia é aplicável a qualquer tipo de serviço Web que apresente características similares, desde que as devidas adaptações sejam feitas.

3.4.1 Visão Geral

Nossa metodologia é modelada de forma hierárquica em três níveis, onde cada nível é responsável pela caracterização de uma das dimensões definidas na Seção 3.2: sessões de usuários, comportamento de usuários e tarefas de usuários. O nível de sessão de usuário compreende métricas como o processo de chegada de usuários ao sistema, duração de sessão etc. O nível de comportamento de usuário foca nas ações desempenhadas pelos usuários durante cada uma das suas sessões (cada requisição que eles submeteram, por exemplo). É importante que esse nível de caracterização explicite uma característica importante de sistemas mineração de dados: a sobreposição temporal entre requisições síncronas e assíncronas, ou seja, o uso simultâneo de servidores interativos e em lote. O nível final, tarefas do usuário, qualifica e quan-

tifica o processo de chegada dos trabalhos de processamento (tarefas), da natureza delas, dos parâmetros escolhidos para cada execução etc.

Os dois primeiros níveis se aplicam a qualquer sistema que possua características semelhantes aos sistemas de mineração de dados. O terceiro nível é que depende diretamente da natureza do sistema, ou seja, cada sistema possui tarefas próprias, com características próprias. Assim, é neste nível que as métricas devem ser alteradas para permitir caracterizar aplicações diferentes.

Os critérios de cada nível da metodologia podem ser aplicados sobre todas as sessões de usuários ou apenas sobre um sub-grupo dessas sessões. Através de análise de similaridade nos padrões de requisições de usuários para um dado serviço, diferentes grupos de sessões de usuários podem ser identificados e a cada um desses grupos podem ser caracterizados de forma independente. Nós apresentamos um exemplo dessa estratégia posteriormente em nossos resultados. Nas seções a seguir, nós discutimos cada uma dos três níveis da metodologia de caracterização em maior detalhe.

3.4.2 Sessões de Usuários

Este nível do nosso modelo de caracterização modela o padrão de acesso dos usuários do sistema. O objetivo nesse nível é encontrar modelos que representem uma informação temporal relacionada à carga criada pelos usuários que iniciam uma sessão no sistema, determinando quão freqüentemente eles acessam o sistema e quão longa são essas sessões. Duas métricas diferentes e complementares são propostas aqui:

- **tempo entre chegada de sessões:** a distribuição que modela o tempo entre a chegada de duas sessões consecutivas ao sistema;
- **duração das sessões:** corresponde ao tempo durante o qual os usuários estão ativos, isto é, submetendo requisições.

Além dessas, é também importante caracterizar o número das requisições submetidas ao sistema durante cada sessão. Isso quantifica o número de ações que cada usuário executa enquanto está conectado ao sistema, e como essas ações são distribuídas temporalmente. Em nossa metodologia nós propomos o uso de um histograma de distribuição das requisições por cada sessão, mostrando como as sessões são agrupadas em termos do número de tarefas em cada sessão.

A caracterização de sessões de usuários é uma ferramenta importante para definir, na perspectiva dos usuários, a intensidade de uso e como ela varia ao longo do tempo.

Essa informação permite avaliar os hábitos dos usuários, por exemplo determinando períodos durante os quais houve uso intenso, e é fundamental para a construção de ferramentas de geração de carga, por exemplo.

3.4.3 Comportamento de Usuários

Uma etapa essencial na metodologia de caracterização é a análise dos padrões de requisições dos usuários. Isso significa a análise de cada uma das requisições que ele requisita a cada funcionalidade do sistema e em como esses pedidos se relacionam durante uma sessão. Uma técnica de modelagem que é amplamente utilizada nestes casos é o *Customer Behavior Model Graph*, ou CBMG [MA00, HTMRG+04, NAR+04]. CBMGs são grafos de transição de estado em que cada nó ou estado representa alguma funcionalidade do sistema que pode ser requisitada pelo usuário e as arestas representam as transições entre as funcionalidades disponíveis. Cada aresta recebe um valor que representa a probabilidade de um usuário deixar um determinado estado e acionar outro, o que acontece quando ele decide solicitar uma nova funcionalidade, uma vez que a precedente já foi executada.

A metodologia CBMG é uma representação semanticamente rica e condensada do comportamento dos usuários, onde diferentes grupos de usuários podem ser representados por CBMGs individuais que se diferem nos estados considerados e pelas probabilidades das transições entre eles. Quando esta separação nos grupos é desejada, técnicas de agrupamento são usadas para determinar as melhores soluções para agrupar sessões.

Entretanto, CBMGs são limitados à representação de comportamentos de usuários que são sempre seqüenciais, ou seja, onde apenas uma funcionalidade é acessada por vez, e transições acontecem quando o usuário finaliza uma determinada funcionalidade e decide iniciar um outra. Os sistemas alvo dessa metodologia de caracterização, entre os quais se incluem sistemas de mineração de dados, estão sujeitos a comportamento de usuários que são essencialmente seqüenciais como a criação de tarefas e a visualização de seus resultados, mas também a atuações assíncronas, uma vez que os processos de mineração podem ser executados independentemente das sessões dos usuários. Os usuários podem decidir interagir de forma completamente seqüencial, esperando que cada tarefa seja finalizada antes de requisitar um nova funcionalidade, mas eles podem também optar em trabalhar com outra tarefa enquanto esperam uma determinada tarefa finalizar a execução. Do ponto de vista de representação do comportamento, devemos ser capazes de caracterizar tanto transições referentes ao comportamento dos usuários, quando uma nova requisição é feita a alguma outra funcionalidade, quanto transições devido a tarefas que foram

executadas (freqüentemente independentes) em paralelo. Isso significa que a técnica do CBMG, se aplicada em sua forma convencional, não pode ser usada para modelar tais aplicações. Esta foi a motivação para a criação de uma nova representação que permita modelar o comportamento quando os usuários iniciam múltiplas atividades em paralelo. Nós denominamos esse modelo de USIMG, *User-System Interaction Model Graph*, o qual é introduzido neste trabalho. No USIMG, cada vértice representa um estado no sistema, que pode compreender apenas uma ação interativa, a exemplo do CBMG, ou uma ação em lote, ou ambas, o que é representado pelo chamado estado composto, que agrega duas ou mais ações que estejam acontecendo no sistema como um todo sob os auspícios de um dado usuário.

Para criar os USIMGs, o primeiro passo é identificar as funcionalidades do sistema e determinar os marcos que indicam o início e o fim de cada uma delas. Cada uma dessas funcionalidades identificadas formam o conjunto de estados que denominamos de "estados simples". Uma vez feito isso, o próximo passo é ordenar cronologicamente o início e o fim de cada uma das funcionalidades. Nós podemos assim identificar dinamicamente os estados que nós denominamos de "estados compostos" observando o momento quando múltiplas atividades foram iniciadas, mas que ainda não foram finalizadas. Esses estados compostos são assim uma composição de dois ou mais estados simples que estão ativos simultaneamente. A Figura 3.2 ilustra o processo para a criação dos USIMGs. Mais uma vez destacamos a importância da sincronização dos relógios entre os servidores, caso contrário, a identificação equivocada do início e/ou do fim de uma requisição do usuário pode acarretar a geração de estados que não representem de forma real o comportamento do mesmo.

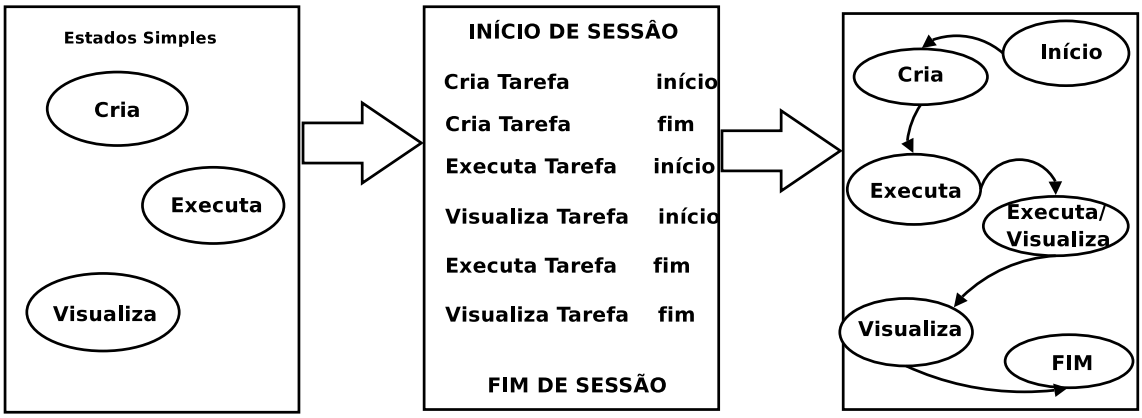


Figura 3.2: Processo de Geração dos USIMGs

O algoritmo de construção dos USIMGs é bastante simples e se baseia em uma lista encadeada e suas operações. Para o algoritmo, é passada uma lista com as

requisições dos usuários encontradas no logs, ordenadas cronologicamente. Primeiramente insere-se o estado de início da sessão no USIMG. Posteriormente, para cada elemento contido nessa lista, verifica-se qual o seu tipo, ou seja, início ou fim. Casa seja o início de uma requisição, insere-se a mesma em uma outra lista que contém o estado atual do usuário. Feita a inserção, insere-se o conteúdo dessa lista no USIMG, pois esse conteúdo é agora um novo estado do usuário. Além disso, insere-se uma aresta no USIMG do último estado do mesmo para o estado que acabou de ser inserido. Se o tipo da requisição for o fim, ou seja, a requisição foi finalizada, remove-se a mesma da lista de estado que mantém o estado atual do usuário. Se essa lista não estiver vazia, significa que o usuário finalizou uma determinada tarefa mas continua trabalhando com outras, ou seja, um novo estado, representado pelo conteúdo atual da lista de estados. Basta, então, inserir novamente o conteúdo dessa lista no USIMG, além da aresta do último estado do USIMG para o estado que acabou de ser inserido. Se após a remoção a mesma permanecer vazia, significa que o usuário simplesmente finalizou todas suas tarefas pendentes, ou seja, não foi para nenhum novo estado. Ao final do tratamento de todas as requisições, basta acrescentar o estado de fim de sessão e uma aresta do último estado do USIMG para esse estado final. A seguir nós apresentamos o pseudo-código desse algoritmo de cálculo dos estados dos USIMGs:

CalculaEstadosUSIMG(*lista_requisicoes*)

```
1.  estado_anterior ="início de sessão";
2.  InsereEstadoUSIMG(estado_anterior);
2.  para toda (requisicao_i ∈ lista_requisicoes) faça
3.      se (requisicao_i.tipo ="início") então
4.          InsereLista(lista_estado_atual, requisicao_i);
5.          InsereEstadoUSIMG(lista_estado_atual);
6.          InsereArestaUSIMG(estado_anterior, lista_estado_atual);
7.          estado_anterior = lista_estado_atual;
8.      senão se (requisicao.tipo = "fim") então
9.          RemoveLista(lista_estado_atual, requisicao_i);
10.         se (TamanhoLista(lista_estado) > 0) então
11.             InsereEstadoUSIMG(lista_estado_atual);
12.             InsereArestaUSIMG(estado_anterior,
                                  lista_estado_atual);
13.         estado_anterior = lista_estado_atual;
```

```
14.          fim se
15.      fim se
16.  fim para
17.  estado_atual ="fim de sessão";
18.  InsereEstadoUSIMG(estado_atual);
19.  InsereArestaUSIMG(estado_anterior, estado_atual);
```

Quando todos os estados que podem acontecer durante uma sessão estiverem definidos, técnicas de agrupamento podem ser aplicadas para identificar grupos de sessões que apresentam uma similaridade no comportamento, devidamente consistentes para serem separados em grupos diferentes. Esses novos grupos podem ser representados por um USIMG individualizado, que pode ser utilizado para caracterizá-los.

Utilizando os modelos encontrados nesse nível da caracterização, podemos gerar carga sintética que representa de forma fiel o comportamento dos usuários caracterizados, incluindo as operações simultâneas que forem identificadas.

3.4.4 Tarefas de Usuários

Feita a caracterização do padrão de acesso dos usuários ao sistema e o padrão das requisições, o último nível da metodologia de caracterização é modelar os detalhes das operações iniciadas pelos usuários. No contexto de mineração de dados, as requisições importantes são as execuções dos algoritmos de mineração de dados, ou tarefas. Cada uma dessas tarefas pode ser caracterizada por três informações distintas: o algoritmo selecionado, os parâmetros necessários para a execução de cada algoritmo, e o banco de dados selecionados pelos usuários.

Nesse caso, nossa metodologia propõe os seguintes mecanismos de caracterização:

- **processo de chegada de tarefas:** é representado pela distribuição do tempo entre chegadas consecutivas de novas tarefas a serem processadas pelo sistema. Como feito em casos anteriores, isso é feito através da análise da distribuição de freqüência do tempo entre chegadas.

- **algoritmos de mineração:** nós devemos determinar quão freqüentemente cada algoritmo é selecionado pelos usuários. Isso pode ser feito através da construção de um histograma da popularidade dos algoritmos disponíveis nos logs.

- **parâmetros:** para cada algoritmo, a popularidade individual dos valores de seus parâmetros devem ser determinados, novamente através do uso de um histograma/distribuição de popularidade. Se um algoritmo possui mais de um parâmetro, eles devem ser considerados de forma independente e possíveis correlações entre eles devem ser investigadas. Outra informação que deve ser analisada é como os parâmetros mudam durante uma sessão interativa, na qual um usuário pode executar o mesmo algoritmo múltiplas vezes. Isso pode ser feito separando os parâmetros em classes utilizando critérios temporais ou de popularidade, entre outros.

- **base de dados:** deve haver também uma caracterização de como os usuários acessam as bases de dados disponíveis para ele para mineração. Isso pode ser feito determinando a popularidade das bases de forma geral ou separadas por algoritmo nas tarefas executadas. Deve-se também analisar a popularidade dos atributos de cada base separadamente, assim como o número de atributos utilizados em cada mineração.

Através da aplicação desses critérios de caracterização, será possível entender melhor como os usuários acessam o sistema e então será possível melhorar o desenvolvimento de novas funcionalidades, prevendo o comportamento futuro, além de tornar possível a criação de um gerador de carga que poderá ser utilizado para avaliar o desempenho do sistema sob diferentes situações, bem como avaliar o impacto das alterações na arquitetura do sistema, por exemplo.

3.5 Sumário

Neste capítulo apresentamos a nossa proposta de metodologia de caracterização de sistemas que possuam componentes síncronos e assíncronos. Esta proposta estende trabalhos anteriores, em particular aqueles dedicados a sistemas síncronos (p.ex., Web), incorpora mecanismos de caracterização de carga de processamento em lote, e introduz uma nova representação que permite relacionar o comportamento dos usuários e seus efeitos em servidores interativos e em lote. No próximo capítulo apresentamos a aplicação dessa metodologia a um serviço de mineração de dados real.

Capítulo 4

Estudo de Caso

Neste capítulo apresentamos um estudo de caso realizado aplicando a metodologia de caracterização descrita no Capítulo 3 a um serviço de mineração de dados denominado Tamanduá.

Primeiramente, na Seção 4.1 descrevemos a arquitetura desse ambiente, identificando seus componentes e o funcionamento deles. Posteriormente, na Seção 4.2 apresentamos os resultados obtidos na aplicação de cada um dos níveis da metodologia de caracterização nesse ambiente. Encerramos o capítulo com um sumário dos resultados encontrados.

4.1 Arquitetura Tamanduá

A arquitetura Tamanduá é baseada no ciclo de vida padrão de descoberta de conhecimento em bases de dados [HK00] e é ilustrada na figura 4.1. Todas as interações associadas com requisições aos serviços de mineração de dados são realizadas através da Web, com base numa interface, que acessa os serviços de mineração através do uso de tecnologias *Web Services*.

Cada usuário interage com o sistema através de um servidor de aplicação que por sua vez interage com os diversos serviços necessários para satisfazer as requisições dos usuários. Esse servidor é responsável pela autenticação do usuário e pela implementação dos controles de acesso adequados. Diversos servidores de aplicação podem existir, tanto para fins de distribuição de carga quanto para implementação de lógicas de serviço diferentes em função da origem dos usuários. Essa solução garante uma forma simples de identificação de serviços e expansão da arquitetura [KGL+03].

O processo de mineração começa com a aquisição de dados, quando dados de diversas fontes e formatos são integrados em repositórios de dados consistentes controlados pelos servidores de dados. Os dados propriamente ditos podem se encontrar

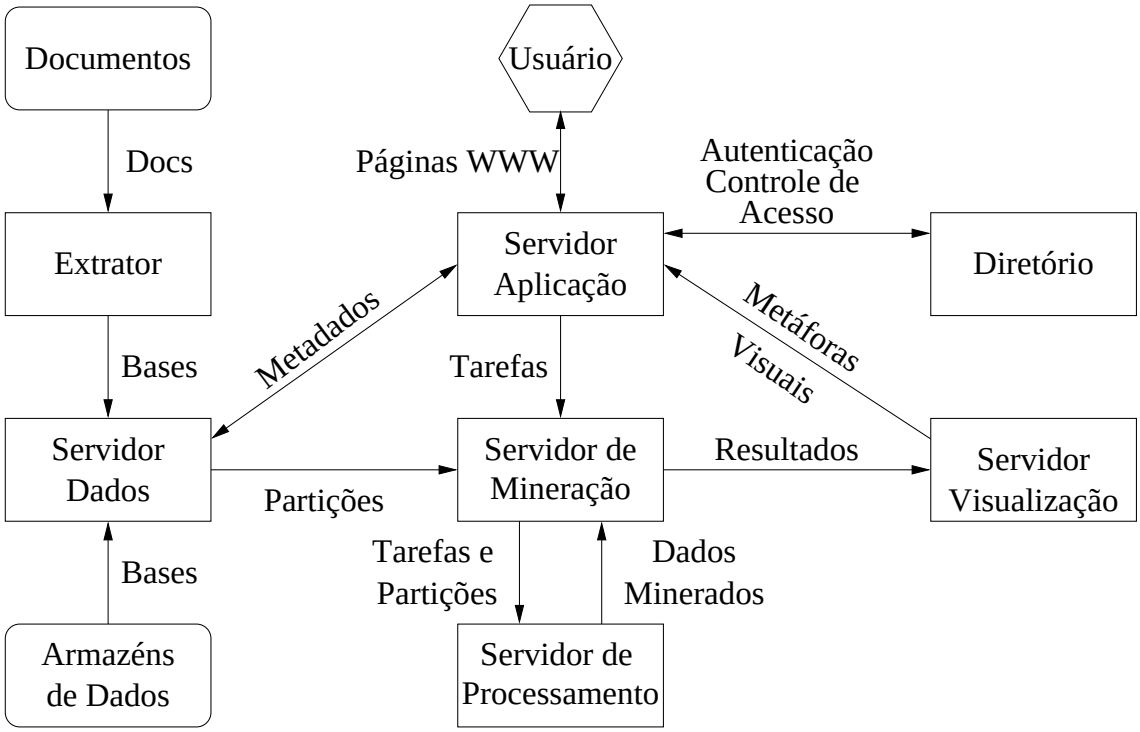


Figura 4.1: A arquitetura Tamanduá

em armazéns de dados já existentes, bancos de dados convencionais, ou podem ser obtidos a partir de fontes não estruturadas, como páginas da WWW. Nesse último caso, a arquitetura prevê recursos para o desenvolvimento de ferramentas extratoras capazes de retirar a informação relevante das páginas para gerar uma base de dados adequada.

Os servidores de dados do Tamanduá mantêm informações sobre as bases de dados que podem ser mineradas. Através de uma interface de serviço pré-definida, cada servidor de dados pode anunciar a sua existência, fornecendo tanto os dados propriamente ditos quanto meta-dados que descrevam cada base disponível. Todos os servidores implementam uma interface de consultas através da qual os usuários podem identificar as características das bases disponíveis, selecionar atributos de interesse e determinar intervalos de dados que devem ser considerados durante o processo de mineração [Pat02].

O usuário escolhe uma base sobre a qual pretende minerar através das informações obtidas de seus meta-dados. Depois de determinar os dados a serem minerados, o usuário especifica a tarefa de mineração a ser executada, que inclui também o algoritmo a ser utilizado, com seus parâmetros. Esse algoritmo deve ser escolhido entre aqueles oferecidos pelos servidores de mineração disponíveis.

Apesar do servidor de mineração do Tamanduá já dispor de diversos algoritmos de mineração implementados, até o momento apenas um algoritmo de mineração de regras de associação está disponível para os usuários. Uma regra de associação representa uma relação entre dois ou mais itens de uma base de dados. Por exemplo, analisando a bases de vendas de uma papelaria, podemos encontrar uma regra que diz que 80% dos clientes que compram lápis e borracha também compram caderno, e que 50% das vendas dessa papelaria envolvem esses três itens.

Esse algoritmo de mineração de regras de associação disponibilizado é o o Apriori [AIS93, VJF+04], cujos parâmetros são suporte e confiança. A confiança representa o valor de previsão (ou probabilidade condicional) de uma regra minerada. No exemplo de regra do parágrafo anterior, se um cliente compra lápis e borracha podemos arriscar que ele também irá comprar caderno com uma chance de acerto de 80%. O suporte representa a freqüência de ocorrência do evento formado pelos itens da regra e dá uma medida da sua significância estatística. Na regra exemplo do parágrafo anterior, o suporte de 50% indica que 50% de todas as transações realizadas na papelaria incluíram os itens lápis, borracha e caderno. Assim, deve-se informar para o algoritmo Apriori quais os valores mínimos de suporte e confiança que uma regra deve apresentar para ser considerada interessante, sendo que esses valores variam no intervalo de 0 a 100%.

Os algoritmos de mineração de dados são computacionalmente intensivos, tendo complexidade exponencial em muitos casos, o que é agravado pelo grande volume de dados que se tem que analisar. A plataforma Tamanduá emprega versões paralelas desses algoritmos que se caracterizam pela alta escalabilidade. Essa característica é importante por minimizar as demandas computacionais localizadas, permitindo o reaproveitamento de parque existente em momentos de ociosidade. A eficiência desses algoritmos paralelos está associada à estratégia de paralelização baseada em fluxos identificados de dados [VMF+04, FJG+05], e que tem resultado em eficiências próximas de 100% para configurações de grande porte, com dezenas de máquinas, em virtude da assincronia inerente ao algoritmo e exploração da sobreposição entre computação e comunicação.

O processamento da tarefa é realizado em um ambiente de cluster, utilizando técnicas de algoritmos paralelos baseados no paradigma *Filter Labeled-Stream* [VMF+04, FJG+05]. Minerar conjuntos de dados enormes e potencialmente dinâmicos é uma tarefa muito intensiva computacionalmente. Portanto, a implementação de algoritmos eficientes de mineração de dados em um ambiente de computação paralela de alto desempenho é crucial para melhorar os tempos de resposta. Cada servidor de mineração coordena a utilização de um cluster, anunciando para os servidores de

aplicação os algoritmos nele implementados e recebendo as requisições por processamento feitas pelos clientes.

Uma vez escolhida pelo cliente a base de dados a ser minerada e o algoritmo a ser aplicado sobre os dados, o servidor de aplicação requisita o serviço ao servidor de mineração. Este, por sua vez, coordena a transferência dos dados do servidor de dados para o servidor de processamento, inclusive gerenciando um mecanismo de cache para bases de dados já transferidas anteriormente.

Durante a execução da tarefa, o servidor de mineração monitora a execução e é capaz de fornecer informações sobre o seu andamento. Quando a tarefa é completada, a aplicação pode exibir o resultado para o usuário, solicitando a um servidor de visualização que crie as abstrações visuais do modelo de acordo com a tarefa e parâmetros informados pelo usuário. A possibilidade de usar serviços de visualização distintos permite a distribuição de carga relacionada ao processo de exibição de resultados e a escolha de diferentes servidores em função do tipo de visualização desejado e das metáforas visuais disponíveis em cada servidor [NS02].

A implantação desse paradigma fez surgir novas necessidades que distinguem o Tamanduá dos demais modelos de mineração de dados. Para operar em uma arquitetura orientada a serviços (*Service Oriented Arquitecture*, SOA) [Arc04], as aplicações (ou serviços) têm que definir, de forma declarativa, suas exigências e capacidades em um formato pré-definido entre as máquinas [CKM+03]. A popularidade dos *Web Services* como um método para comunicação entre programas tornou essa solução uma escolha natural para a implementação de SOA, provendo acesso a serviços através da Web, baseado em padrões bem definidos e amplamente difundidos.

Uma SOA se caracteriza principalmente pela disponibilização de serviços que, descritos de uma maneira interpretável por máquinas, possam ser combinados para resolver algum problema. No Tamanduá, esses serviços são o repositório de dados, o servidor de mineração e o servidor de visualização e o problema que se deseja resolver é a mineração de uma base de dados e a descoberta do conhecimento a partir da visualização dos resultados obtidos.

A arquitetura Tamanduá pode ser mapeada na arquitetura alvo de caracterização da metodologia proposta, descrita na Seção 3.1. No Tamanduá nós temos quatro servidores interativos (servidor de dados, de visualização, de aplicação e de mineração) e um servidor em lote (servidor de processamento), onde todas as requisições de usuários são feitas através de um servidor interativo central (servidor de aplicação) que repassa as mesmas para os demais servidores interativos. Assim, os quatro servidores interativos são os responsáveis pelas requisições síncronas dos usuários como visualização de resultados, visualização de base de dados e criação de tarefas, entre

outras, e o servidor de processamento é responsável pelas requisições assíncronas que são as execuções das tarefas de mineração de dados. De acordo com a metodologia, a dimensão de padrão de acesso irá modelar as interações dos usuários com os servidores de aplicação, de dados e visualização, a dimensão de requisições modela tanto as requisições a esses servidores quanto ao servidor de mineração de dados. Por fim, a dimensão de análise da natureza das requisições irá tratar exclusivamente da natureza das tarefas de mineração submetidas ao servidor de mineração.

4.2 Resultados

Nesta Seção mostramos os resultados da caracterização do Tamanduá, uma aplicação de mineração de dados descrita na Seção anterior e que vem sendo utilizado por estudantes de cursos de mineração de dados, que utilizavam o Tamanduá em aulas práticas e no desenvolvimento de trabalhos, e por organizações governamentais que têm como objetivo identificar fraudes em compras do governo e discrepâncias entre os salários de funcionários públicos, entre outros.

Como dito anteriormente, a arquitetura do Tamanduá é dividida em um servidor de mineração, um servidor de dados e um servidor de visualização, e todas as requisições dos usuários são submetidas através de um servidor de aplicação, que dispara as ações necessárias a serem executadas pelos outros servidores. A coleta de dados para análise foi feita de forma descentralizada, conforme proposto pela metodologia na Seção 3.3. Os logs do servidor de aplicação foram gerados a partir de logs tradicionais de um servidor Web e o restante dos servidores foram devidamente instrumentados para que os logs pudessem ser coletados seguindo a sugestão da metodologia. Todas as requisições dos usuários foram registradas, assim como as respostas das mesmas, e o relógio de todos os servidores foram sincronizados utilizando NTP(*Network Time Protocol*) [Mil92], o que na prática pouco importou uma vez que, na versão atual do Tamanduá, todos os servidores estão sendo executados na mesma máquina. Uma descrição dos logs coletados em cada um dos servidores do Tamanduá, assim como um amostra de tais logs, segue em anexo no Apêndice A. Os logs utilizados neste trabalho foram coletados no período de março a maio de 2005.

Seguindo a metodologia, o passo seguinte à coleta dos dados é a preparação dos mesmos, descrita na Seção 3.3.1. Nessa preparação agrupamos os logs coletados nos diversos servidores diferentes em um único log, verificando a integridade dos registros coletados e ordenando os mesmos de forma cronológica, ou seja, pela ordem de ocorrência. A partir desse único log, que contém todos os registros de todos os

servidores, foi feita a caracterização apresentada nesta seção. Uma descrição mais detalhada desse único log e uma amostra do mesmo segue em anexo no Apêndice B.

Começamos a apresentação dos resultados descrevendo a carga que foi usada em nossa caracterização. Em seguida, discutimos as múltiplas dimensões do modelo de caracterização hierárquica que foi utilizado para analisar a carga de trabalho, incluindo as sessões dos usuários, o comportamento dos usuários e as tarefas que eles criam.

4.2.1 Visão Geral da Carga de Trabalho

Período	Março-Maio 2005
Total # usuários	145
Total # sessões	932
Total # tarefas	1035
Média (CV) – duração da sessão (minutos)	29,50 (1,70)
Média (CV) – # sessões por usuário	6,43 (1,53)
Média (CV) – # tarefas por sessão	1,12 (1,73)

Tabela 4.1: Sumário da Carga (CV = Coeficiente de Variação)

Uma visão geral da carga de trabalho utilizada em nossa caracterização é apresentada na Tabela 4.1. Neste período um total de 932 sessões de usuários foram realizadas, para um grupo de 145 usuários, o que totaliza um média de 6,4 sessões por usuários. O coeficiente de variação (CV) [1], no entanto, indica que existe uma quantidade significativa de usuários que desviam significativamente da média. O mesmo acontece com a duração das sessões e o número de tarefas por sessão, o que sugere uma grande variabilidade no comportamento do usuário. Este fato é considerado durante nossa caracterização, no sentido em que procuramos quantificar a variabilidade dos valores em cada um dos critérios avaliados.

4.2.2 Sessões de Usuários

Primeiro, consideramos as sessões dos usuários. Para estudar o processo de chegada das sessões, primeiramente observamos que a interação dos usuários ao longo do período de log coletado era composta de sub-períodos de interação intensa, interação pouco intensa e períodos de nenhuma interação. Os períodos de interação intensa estão associados a cursos que foram ministrados ou aulas práticas, os momentos de nenhuma interação representam os finais de semana e madrugadas e os

[1] Coeficiente de Variação é a razão entre o Desvio Padrão e a Média

momentos de interação pouco intensa representam o restante do log, nos quais o uso do sistema era esporádico. Se considerássemos todo o período na análise do processo de chegada teríamos que a média seria fortemente influenciada por esses períodos de pouca ou nenhuma interação e o modelo não seria representativo de uso intenso do sistema. Dessa forma, para a análise do processo de chegada, escolhemos um período de interação intensa, no qual a taxa de chegada de sessões era razoavelmente estável e alta[2], evitando assim, os efeitos indesejáveis de agregação de dados.

Na Tabela 4.2 apresentamos um sumário da carga de trabalho considerada para análise do processo de chegada de sessão. Essa carga representa três horas de uma aula prática no dia 29 de março de 2005. Apenas nesse período observamos que foram disparadas 17% das tarefas contidas em todo log, além de representar 8% de todas as sessões, o que mostra que se trata de uma carga bastante representativa. Em todo log coletado, encontramos outros 13 períodos como esse de aulas práticas ou cursos, nos quais a interação eram intensa, representando, juntamente com o período utilizado na analise, 61,6% de todas as sessões e 63,3% de todas as tarefas, o que, além de mostrar que o sistema Tamanduá possui períodos de rajadas de uso, induzidos por curso ou aula práticas, reforça a significância da carga utilizada na análise do processo de chegada.

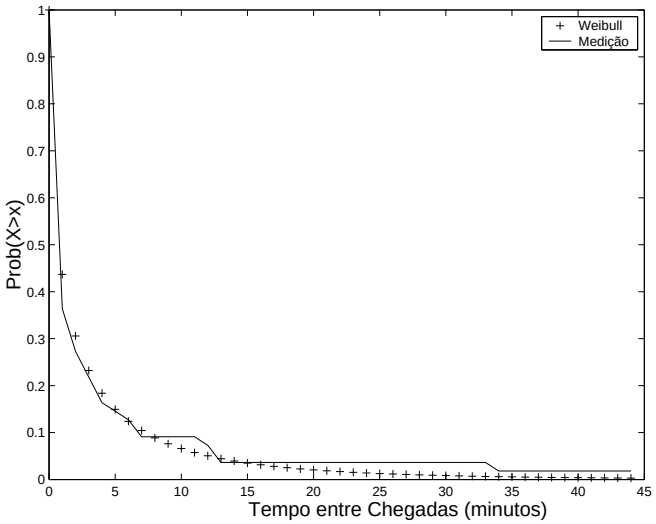
Período	29 Março 2005 19hs às 22hs
Total # usuários	34
Total # sessões	78
Total # tarefas	176
Média (CV) – duração da sessão (minutos)	26,88 (0,93)
Média (CV) – # sessões por usuário	2,36 (0,78)
Média (CV) – # tarefas por sessão	2,25 (1,34)

Tabela 4.2: Sumário da Carga Considerada no Processos de Chegada de Sessão e de Tarefas (CV = Coeficiente of Variação)

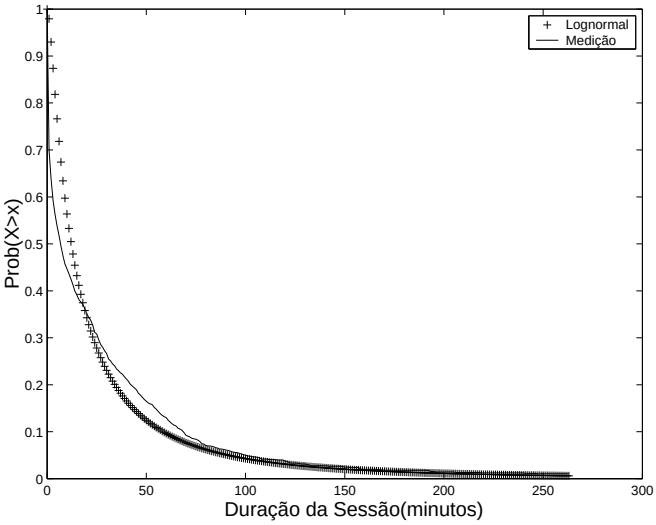
Feita a análise desse período de estabilidade escolhido, pudemos concluir que o tempo entre chegadas de sessões de usuários pode ser representada por uma distribuição Weibull [Wei05c], como ilustrado pela Figura 4.2(a).

Além de analisarmos o processo de chegada, conforme propõe a metodologia, a duração das sessões também foi caracterizada. Diferente do processo de chegada de sessões, no qual um período específico do log foi selecionado para análise, na caracterização da duração das sessões, todas as sessões foram consideradas. O motivo é

[2]Isto é, com muitas novas sessões sendo iniciadas.



(a) Distribuição do Tempo entre Chegadas de Sessões (minutos)



(b) Distribuição da Duração das Sessões (minutos)

Figura 4.2: Caracterização das Sessões de Usuários

que não há problemas de agregação de dados nesse caso, ou seja, mesmo as sessões iniciadas em momentos de pouca ativação possuem uma período no qual há uma interação de usuário. A duração média das sessões é baixa, em torno de 30 minutos, e a maior duração encontrada não é maior que 4 horas. Encontramos que a duração das sessões pode ser modelada por uma distribuição LogNormal [Wei05b], como ilustrado na Figura 4.2(b); os parâmetros dessa distribuição são mostrados na Tabela 4.3.

O próximo passo da caracterização das sessões dos usuários é analisar o número de tarefas de mineração criadas por sessão. Para isso foi feito um histograma do número de tarefas por sessão apresentado na Figura 4.3. Podemos observar que a alta

Distribuição	*Lognormal*		*Weibull*		*Power Law*	
	σ	μ	α	β	a	b
Distr. do Tempo entre Chegada de Sessão	-	-	1.23	2.50	-	-
Distr. da Duração de Sessão	0.83	0.52	-	-	-	-
Distr. do Número de Tarefas por Sessão	-	-	-	-	177.90	-1.29
Distr. do Tempo entre Chegada de Tarefas	-	-	0.89	0.51	-	-

$$\text{Lognormal (PDF): } p_X(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{\frac{-(ln(x)-\mu)^2}{2\sigma^2}}$$

$$\text{Weibull (PDF): } p_X(x) = \alpha\beta^{-\alpha}x^{\alpha-1}e^{-(\frac{x}{\beta})^{\alpha}}$$

$$\text{Power Law: } f(x) = ax^b$$

Tabela 4.3: Sumário das Distribuições

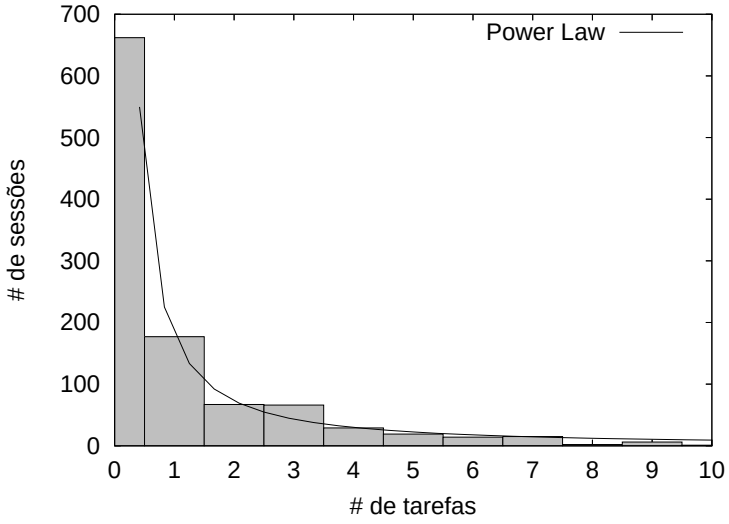


Figura 4.3: Distribuição do Número de Tarefas por Sessão

variância indicada pela Tabela 4.1 da Seção 4.2.1 é confirmado pelo histograma, que mostra que algumas sessões disparam um grande número de tarefas, enquanto outras disparam algumas poucas ou mesmo nenhuma. A partir desses dados, conseguimos aproximar os dados por uma distribuição *Power Law* que também pode ser observada pela Figura 4.3. Essa modelagem é muito importante pois facilita a construção de um gerador de carga sintética.

A modelagem dessas três caracterizações foi feita com base na análise de diversas distribuições populares e documentadas. Essa análise utilizou um programa *Matlab* [Com05] que aproxima os dados medidos com distribuições populares, além disso, esse programa implementa o método dos mínimos quadrados [Wei05a] para

auxiliar nas análises. Assim, através da avaliação dessas aproximações, observando a distância dos mínimos quadrados de cada curva, além de uma inspeção visual das mesmas, encontramos os modelos que melhor representavam os dados medidos.

Através dos resultados conseguimos confirmar a grande variabilidade das características relativas às sessões, ao mesmo tempo em que conseguimos modelá-las através de distribuições estatísticas, permitindo a geração de carga sintética de forma bastante fácil.

4.2.3 Comportamento de Usuário

Após a caracterização das sessões, partimos para a caracterização do comportamento do usuário em cada sessão. O objetivo foi identificar e modelar as interações dos usuários com o sistema. Para isso aplicamos cada um dos passos para a identificação do USIMG, como descrito na Seção 3.4.3, ao log no qual foram unificados os registros de todos os servidores do sistema Tamanduá. Em seguida modelamos o comportamento do usuário em cada sessão, e chegamos ao USIMG mostrado na Figura 4.4.

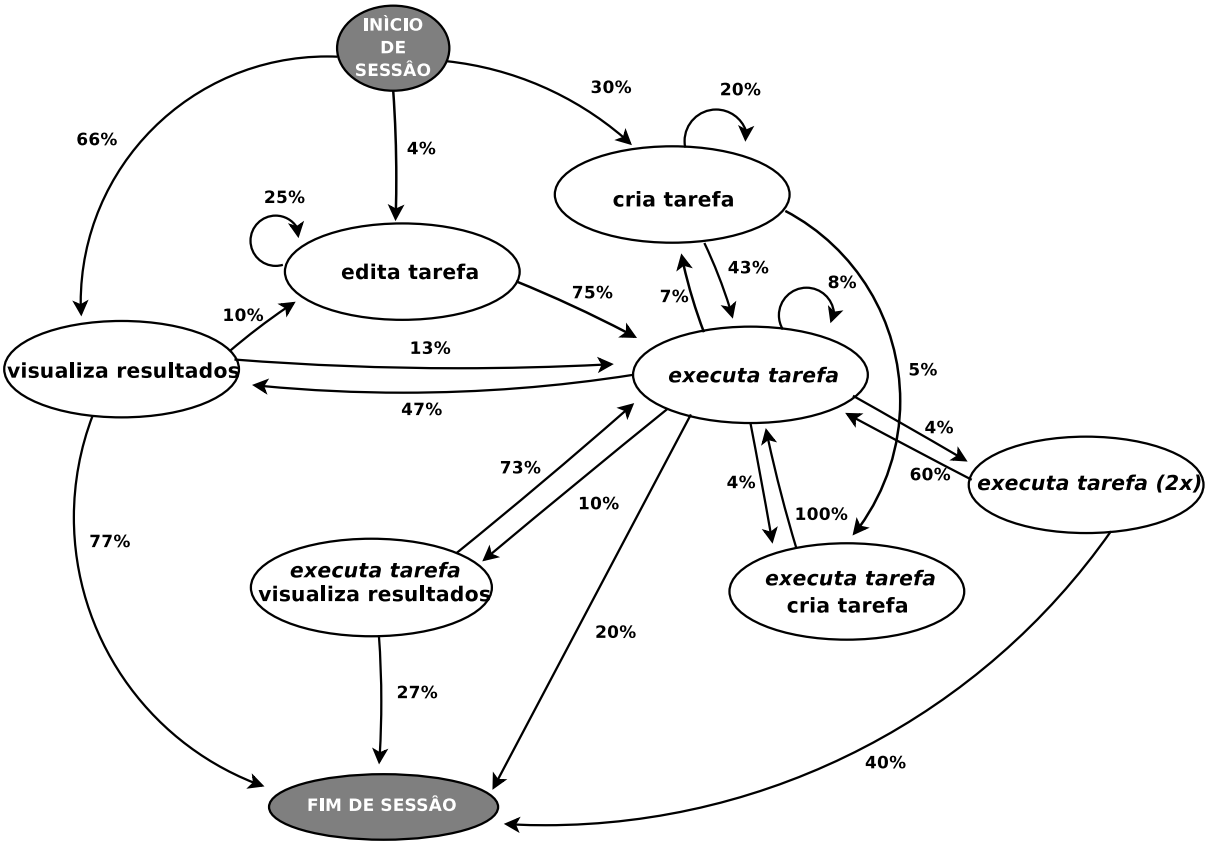


Figura 4.4: USIMG de todas as Sessões

Estado	Porcentagem
Visualiza Resultados	40%
Executa Tarefa	22%
Cria Tarefa	20%
Edita Tarefa	7%
Executa Tarefa (2x)	5%
Visualiza Resultados / Executa Tarefa	3%
Cria Tarefa / Executa Tarefa	3%

Tabela 4.4: Freqüência das Requisições - todas sessões

No USIMG, cada estado representa uma ou mais funcionalidades do sistema que estão sendo requisitadas pelo usuário. Se o usuário requisita mais de uma funcionalidade ao mesmo tempo, o estado é uma combinação dessas funcionalidades. As arestas representam as transições, observadas entre os estados, e são rotuladas com a probabilidade de um usuário executar tal transição. Através do USIMG podemos ver que algumas funcionalidades são de fato requisitadas simultaneamente, o que não poderia ser modelado através dos CBMGs tradicionais. Além disso, baseado no USIMG seria possível criar uma carga de trabalho mais precisa e fiel, representando inclusive as atividades simultâneas dos usuários.

Analisando a Figura 4.4 podemos ver que, ao iniciar uma sessão, o usuário tem uma probabilidade de 66% de solicitar a visualização de resultados anteriores e apenas 34% de chance de requisitar a edição ou criação de uma nova tarefa, embora ele possa criar ou editar tarefas em um outro momento de sua sessão. O USIMG permite modelar cenários onde múltiplos eventos simultâneos sejam disparados pelo usuários enquanto uma tarefa já está executando, sem que essa execução tenha terminado. Isso ocorre em estados como *executa tarefa/cria tarefa*, no grafo da Figura 4.4. Um caso interessante é que o USIMG é capaz de capturar a execução de múltiplas tarefas simultaneamente, como representado pelo estado *executa tarefa (2x)*. Na Tabela 4.4, apresentamos a freqüência de cada estado do USIMG da Figura 4.4. A maior parte do tempo está associado aos estados seqüenciais (visualização de resultados, tarefa em execução e criação de tarefa).

Uma questão importante é sobre a diversidade de comportamentos que um único USIMG pode representar. Para avaliar estes aspectos podemos agrupar as sessões semelhantes, como discutido no Capítulo 3. Assim, além de identificar o USIMG para todas as sessões do log, utilizamos técnicas de agrupamento para separar as sessões dos usuários em grupos que possuam comportamento similar. Utilizamos o algoritmo K-means [Har75] e o número de grupos foi determinado através do βCV, como descrito em [MA00]. Identificamos dois grupos, cujas características

são apresentadas na Tabela 4.5.

	Grupo 0	Grupo 1
% de sessões	68%	32%
% de tarefas	10%	90%
Média (CV) duração de sessão (minutos)	12,19 (2,06)	54,28 (1,25)

Tabela 4.5: Sumário das Características do USIMG

Aplicamos as mesmas técnicas de caracterização para os grupos de sessões encontrados através do agrupamento, o que produziu um USIMG para cada grupo. Esses grafos estão mostrados nas Figuras 4.5 e 4.6. Note que existem alguns estados que aparecem nos grupos mas não apareciam no USIMG único mostrado anteriormente. Isso aconteceu porque decidimos considerar apenas as transições entre os estados mais significativas em cada caso(transições acima de 4%, inclusive), assim, conseqüentemente ocultamos estados irrelevantes que possuem poucas requisições, como por exemplo o estado *Executa Tarefa (3x)* no grafo único de todas as sessões. Observando o grafo de cada um dos dois grupos, notamos que eles separaram as sessões de uma forma bastante interessante.

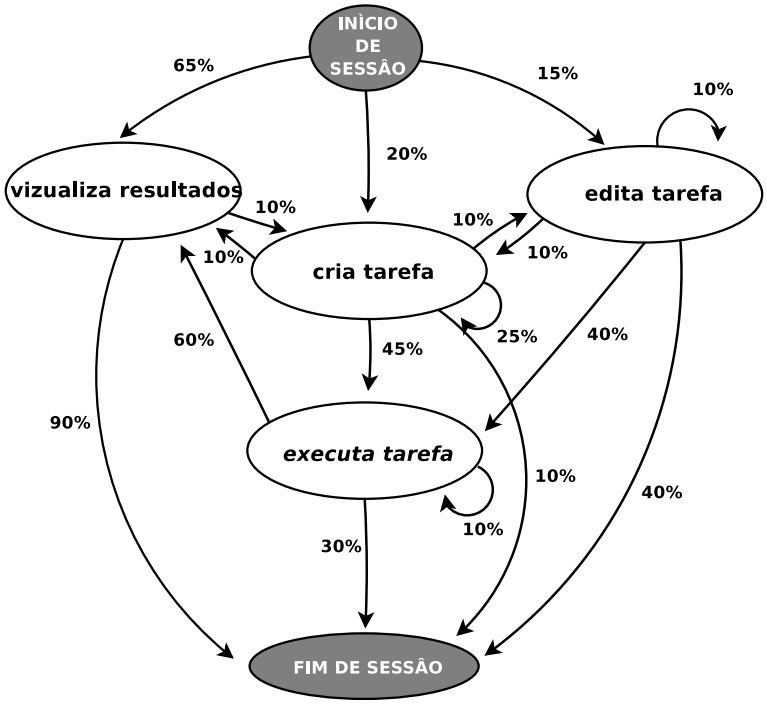


Figura 4.5: USIMG das Sessões do Grupo 0

A primeira classe de sessões, representada pelo grafo 4.5, identifica as sessões que não possuem paralelismo ou estados compostos. Isso significa que o usuário executa

Requisições	Porcentagem
Visualiza Resultados	75%
Cria Tarefa	10%
Edita Tarefa	8%
Executa Tarefa	7%

Tabela 4.6: Freqüência das Requisições - Grupo 0

cada passo seqüencialmente, esperando que uma requisição se complete para iniciar uma nova requisição. Isso forma um ciclo simples: o usuário cria uma tarefa, submete a mesma para execução, aguarda o fim da execução e visualiza os resultados. Esse grupo representa aproximadamente dois terços das sessões, mas, levando em conta sua natureza "seqüencial", elas respondem por apenas 10% das tarefas executadas no sistema, já que os usuários que possuem sessões nesse grupo demora mais a disparar tarefas (Tabela 4.5). Além disso, são sessões que duram em média apenas 12 minutos. Podemos associar esse grupo aos usuários que não têm grande experiência com mineração de dados e/ou utilização do Tamanduá.

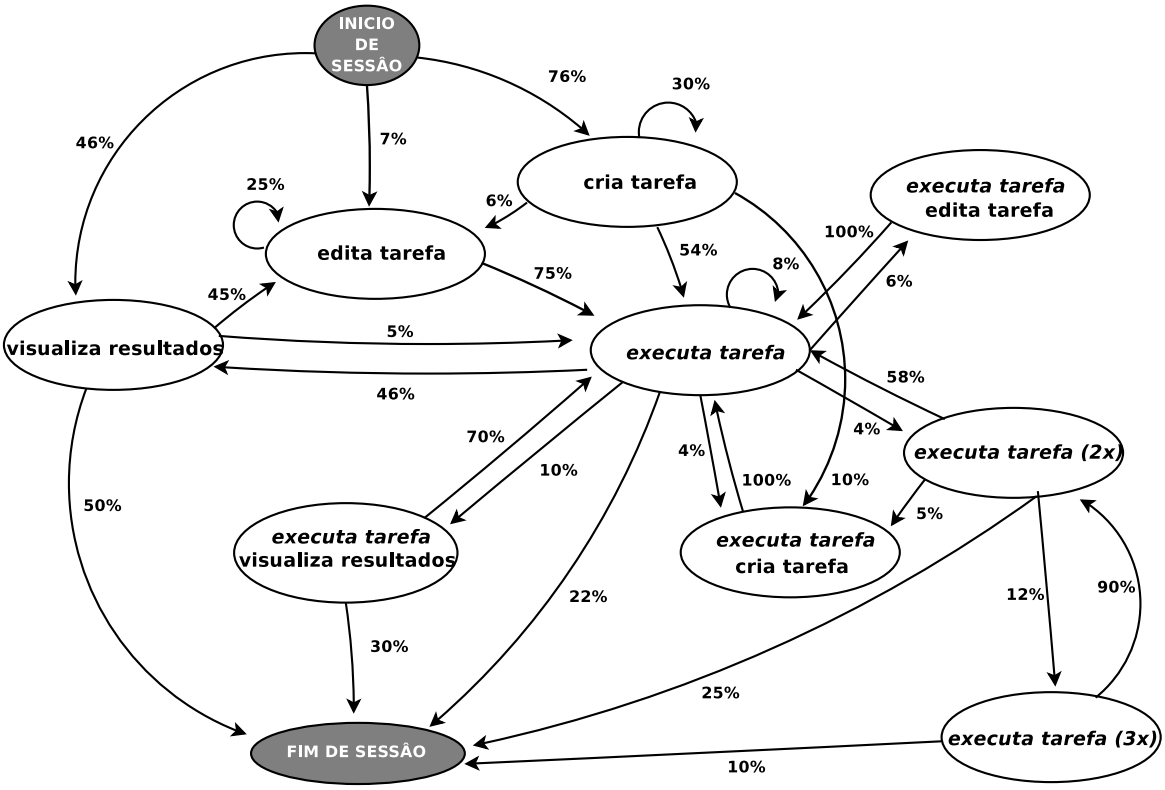


Figura 4.6: USIMG das Sessões do Grupo 1

Requisições	Porcentagem
Executa Tarefa	27%
Cria Tarefa	21%
Visualiza Resultados	20%
Edita Tarefa	8%
Executa Tarefa (2x)	8%
Edita Tarefa / Executa Tarefa	5%
Visualiza Resultados / Executa Tarefa	5%
Cria Tarefa / Executa Tarefa	4%
Executa Tarefa (3x)	2%

Tabela 4.7: Freqüência das Requisições - Grupo 1

O segundo grupo de sessões (Figura 4.6) inclui as sessões em que existe uma notável superposição entre interatividade e processamento em lote (batch). Isso significa que enquanto o usuário espera que uma tarefa termine sua execução, ele utiliza este tempo para outras atividades, como visualizar uma tarefa anterior, criar uma nova tarefa ou mesmo executar outras. Este grupo contém apenas 32% sessões, mas 90% das tarefas registradas são executadas dentro dessas sessões (Tabela 4.5) e a duração das mesmas é de 54 minutos em média, 4,5 vezes maiores que as sessões do outro grupo.

A freqüência de cada estado nos grupos 0 e 1 é mostrada nas Tabelas 4.6 e 4.7. No caso do grupo 0, que foi classificado como o grupo das sessões sem paralelismo, 100% das requisições correspondem aos estados seqüenciais (*Visualiza Resultados* e *Cria Tarefa*). Por outro lado, no grupo 1, classificado como o grupo das sessões com paralelismo, a freqüência dos estados com sobreposição de atividades do usuário(*Edita Tarefa / Executa Tarefa*, *Visualiza Resultados / Executa Tarefa*, *Cria Tarefa / Executa Tarefa* e *Executa Tarefa (3x)*) chega a 16% do total de requisições, o que não pode ser desprezado.

Esses resultados mostram a utilidade do processo de caracterização ser executado para grupos distintos de usuários. Considerando a diversidade encontrada nos modelos, outros resultados interessantes ainda podem ser encontrados. Em geral, esses resultados mostram que precisamos de melhores ferramentas de caracterização e técnicas para avaliar *Web Services*, e mostram também como a diversidade de usuários afeta a carga de trabalho imposta ao sistema, em particular as ações não-interativas. Esses resultados podem ser utilizados para direcionar pesquisas em estratégias de escalonamento e questões relacionadas, como desempenho, disponibilidade e previsibilidade das aplicações, tópicos que têm se tornado relevantes no escopo dos *Web Services* em geral.

4.2.4 Tarefas de Usuários

Finalmente, caracterizamos as tarefas criadas pelos usuários. Assim como existem períodos em que a taxa de chegada de sessão é praticamente nula, existem períodos em que não há execução de nenhuma tarefa de mineração, dessa forma, para o processo de chegada de tarefas, escolhemos o mesmo período estável utilizado na avaliação do processo de chegada de sessões de usuários descrito na Seção 4.2.2, e um sumário da carga desse período também pode ser encontrada na Tabela 4.2. Mais uma vez isso garante que evitemos efeitos indesejáveis da agregação de dados.

Novamente, como no processo de chegada de sessões, o processo de chegada de tarefas pode ser modelado como uma distribuição Weibull, como mostrado na Figura 4.7 e cujos valores dos parâmetros podem ser vistos na Tabela 4.3 da Seção 4.2.2.

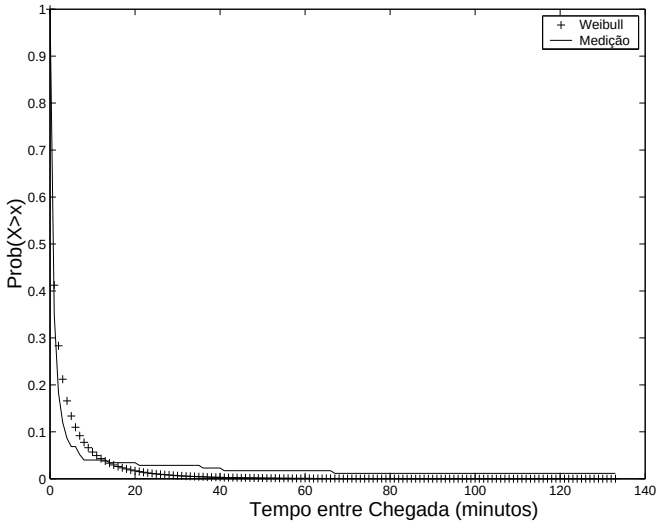
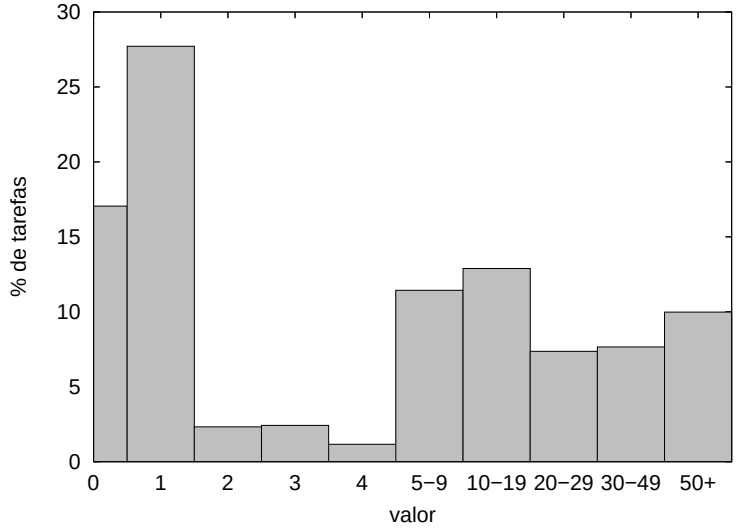


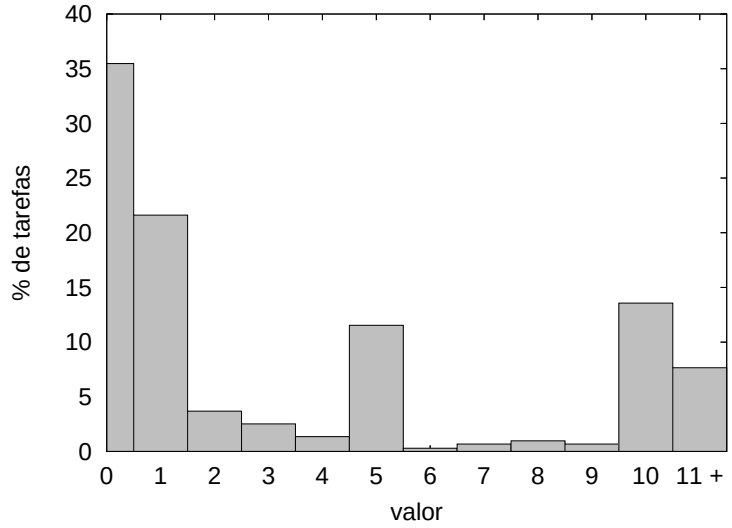
Figura 4.7: Distribuição do Tempo entre Chegada de Tarefas (minutos)

Terminada a caracterização do processo de chegada de tarefas, o próximo passo na metodologia de caracterização proposta é analisar a popularidade dos diversos algoritmos disponíveis no sistema. O Tamanduá, embora disponha de diversos algoritmos já implementados, disponibiliza apenas um algoritmo para uso dos usuários, até o momento: o Apriori [AIS93, VJF+04]. Por essa razão, toda a caracterização a partir deste ponto será realizada para apenas esse algoritmo. No caso de múltiplos algoritmos, analisaríamos a popularidade de cada um deles e também a co-ocorrência entre eles, tanto dentro de uma mesma sessão, quanto para o mesmo usuário, além de realizar todos os passos a seguir para cada um deles.

Seguindo a metodologia, foi feita a caracterização dos parâmetros das tarefas de mineração submetidas pelos usuários. Primeiramente observamos a popularidade de



(a) Confiança



(b) Suporte

Figura 4.8: Popularidade dos Parâmetros do Algoritmo Apriori

cada um dos parâmetros do Apriori independentemente (suporte e confiança, que possuem intervalo de 0 a 100%). Para isso construímos o histograma de popularidade para os dois parâmetros, que são mostrados na Figura 4.8. Observando os dados, notamos que na maioria das tarefas os usuários escolhem baixos valores para ambos os parâmetros: cerca de 50% das tarefas usam valores de confiança inferiores a 4% e aproximadamente 65% usam suporte menor que 4%. Os valores para a confiança são distribuídos mais uniformemente que os do suporte; isso se deve ao fato de que a confiança é mais usada como medida de quão interessante uma regra pode ser, enquanto o suporte se refere ao número de regras de associação encontrados. Esses valores baixos de suporte mostram que as bases de dados disponíveis aos usuários são

densas, além de uma variedade de atributos tal que apenas valores baixos de suporte conseguem capturar as regras existentes. Não encontramos nenhuma distribuição que modelasse bem esses dados. No entanto, *scripts* para geração de carga podem usar diretamente os dados de freqüência (ou popularidade).

Após analisar cada parâmetro isoladamente, investigamos a correlação entre eles (Tabela 4.8). De acordo com a tabela, aproximadamente 50% das tarefas possuem tanto o suporte quanto a confiança menor que 5. Novamente, a confiança possui um distribuição mais ampla e com mais valores distintos que o suporte. A diagonal na tabela de relações mostra uma correlação entre o aumento do suporte e confiança. Para nós, isso é uma indicação de que alguns usuários não entendem muito bem os conceitos por trás de cada parâmetro utilizado e simplesmente utilizam o mesmo valor para ambos os parâmetros. A tabela de relações entre parâmetros complementa os resultados encontrados a partir dos gráficos da Figura 4.8, para fins de geração de carga.

Confiança / Suporte	0	5	10	15	20	20+
0	48,2%	2,56%	2,97%	0%	0,17%	0,34%
5	2,39%	6,48%	1,62%	0%	0,09%	0%
10	3,58%	2,22%	6,14%	0%	0,17%	0%
15	0,34%	0%	0%	0,51%	0%	0%
20	3,75%	0,60%	0,68%	0,26%	0,43%	0,17%
20+	4,95%	2,30%	2,22%	0,09%	2,56%	4,18%

Tabela 4.8: Relação entre os Parâmetros do Algoritmo Apriori

A Tabela 4.9 mostra uma descrição das principais bases de dados utilizadas, informando o número de tuplas e de atributos de cada uma e a porcentagem de tarefas que utilizaram cada uma delas. Observando a porcentagem de tarefas que usam cada base, temos que 17% das tarefas utilizam as outras 9 bases de dados disponíveis. Essas outras bases de dados não foram consideradas em nossas análise por serem muito pouco utilizadas. O restante dos resultados apresentados foram realizados considerando as tarefas para cada uma dessas três bases de dados de forma independente.

Finalizando à caracterização dos parâmetros das tarefas, estudamos como os usuários refinam uma tarefa de mineração de dados. A Tabela 4.10 mostra como a popularidade dos valores do suporte variam em momentos diferentes: quando uma tarefa é criada, quando ela é refinada pela primeira vez (após os primeiros resultados serem visualizados), e o terceiro momento, quando a tarefa é refinada pela segunda vez. Isso foi feito para duas das bases de dados apresentadas na Tabela 4.9. A tabela

Base de Dados	# de Tuplas	# de Atributos	% de Tarefas
Base de Dados X	14506	16	52
Base de Dados Y	27794	18	21
Base de Dados Z	86611	26	10

Tabela 4.9: Descrição das Base de Dados

mostra que os usuários tendem a reduzir o valor do suporte ao refinar uma tarefa de mineração de dados, isso porque os mesmos não conhecem as características das bases, realizando uma calibração experimental até que as regras fosse surgindo. Isso aumenta a carga no sistema, porque o suporte é determinante no número de regras de associação processados pelo algoritmo (quanto menor o suporte, maior a carga no sistema). Um fenômeno semelhante pôde ser observado nos valores de confiança. Neste contexto vislumbramos a utilização de técnicas de análise de seqüências a fim de quantificar este comportamento.

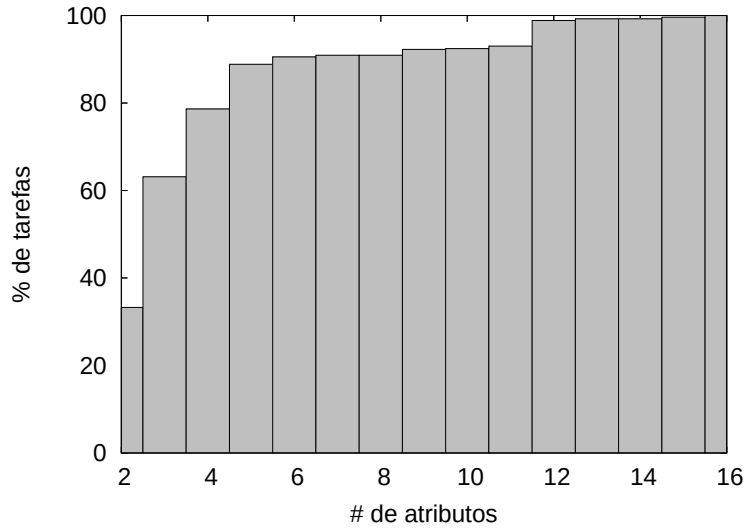
	Base de Dados X				*Base de Dados Y*			
Suporte	*0-5*	*5-10*	*10-20*	*20+*	*0-5*	*5-10*	*10-20*	*20+*
1^o Instante	65,92%	18,15%	12,96%	2,97%	48,80%	19,64%	27,97%	3,59%
2^o Instante	76,70%	9,70%	7,76%	5,84%	60,52%	26,61%	7,89%	4,98%
3^o Instante	89,58%	4,16%	4,16%	2,1%	75%	9,33%	8,33%	7,34%

Tabela 4.10: Refinamento das Tarefas

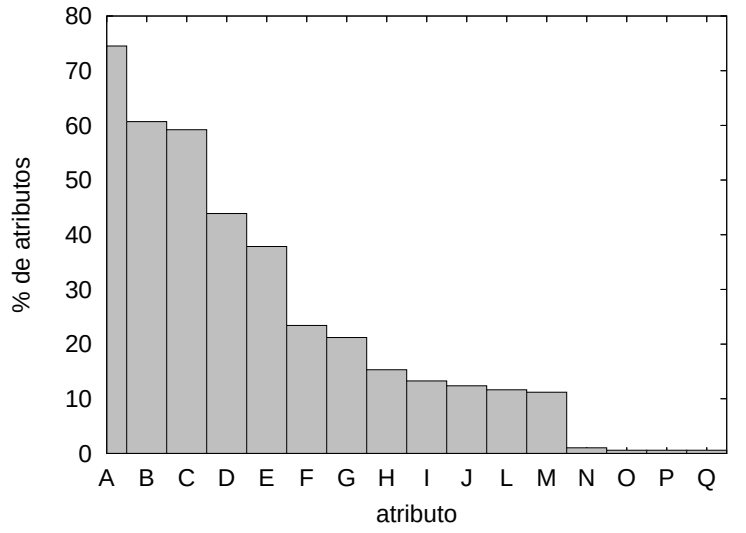
Além dessa informação ser muito útil para um gerador de carga mais refinado e mais pertinente, ela é interessante para o planejamento do sistema como um todo, para escalonamento de tarefas e estratégias de alocação de recursos.

Complementando a caracterização das tarefas analisamos como os usuários utilizam as bases de dados disponíveis e seus atributos. Para isso utilizamos as três bases de dados descritas na tabela 4.9. Separando as tarefas relacionadas a cada uma dessas bases, primeiro verificamos o número de atributos escolhidos pelos usuários para serem utilizados nas bases a serem mineradas e depois examinamos a popularidade dos atributos escolhidos para essas tarefas.

As Figuras 4.9, 4.10 e 4.11 mostram os resultados para cada uma das três principais base de dados. Os histogramas cumulativos para o número de atributos utilizados em cada tarefa são mostrados nas Figuras 4.9(a), 4.10(a) e 4.11(a). Os resultados mostram que 80% das tarefas de mineração utilizam no máximo 8 atributos. Esse dado é particularmente importante, pois o algoritmo Apriori é altamente sensível ao número de atributos. Além disso, os resultados mostram que os usuários



(a) Atributos por Tarefa(Acumulado)



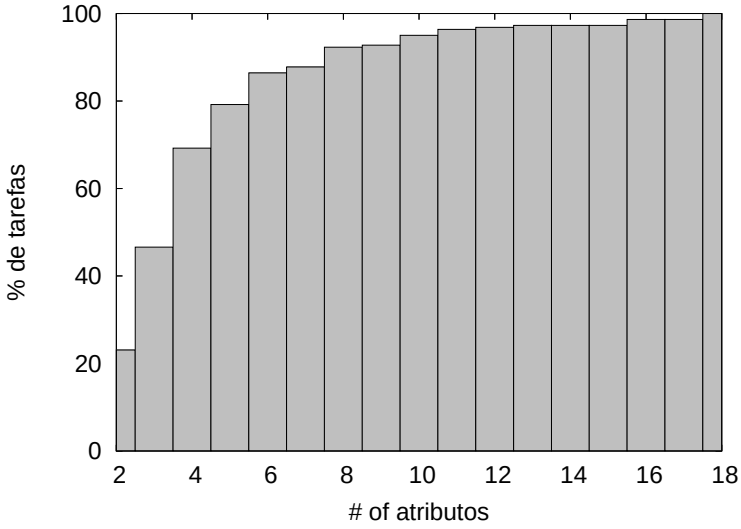
(b) Popularidade de Atributos

Figura 4.9: Base de Dados X

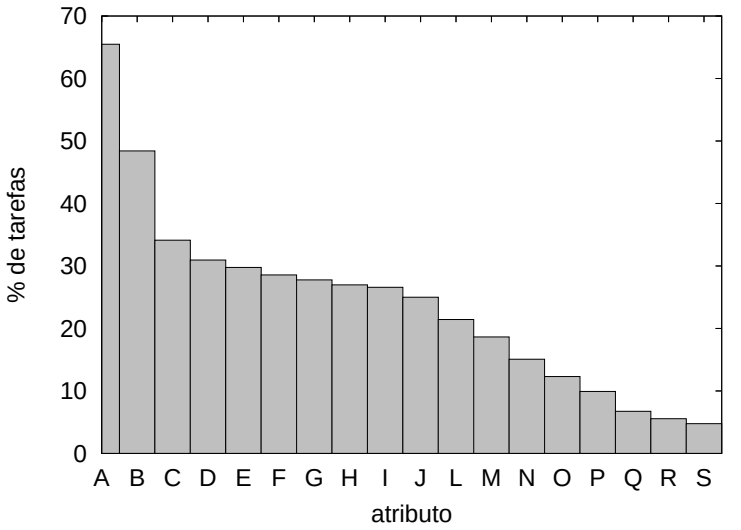
selecionam atributos baseados nas especificidades de cada tarefa de mineração, isto é, dependendo do objetivo da tarefa, os usuários escolhem diferentes atributos, como será mostrado no próximo parágrafo.

As Figuras 4.9(b), 4.10(b) e 4.11(b) mostram os histogramas de popularidade para os atributos utilizados nas tarefas de mineração de dados pra cada uma das mesmas bases de dados. Elas mostram que os 5 atributos mais populares estão presente em mais de 40% das tarefas de mineração de dados em todas as bases. Apesar de ser claro que existem alguns atributos que estão presentes na maior parte das tarefas, podemos apontar também alguma diversidade na natureza das tarefas.

Baseando-se nessa análise, acreditamos que existe uma significativa localidade



(a) Atributos por Tarefa(Acumulado)

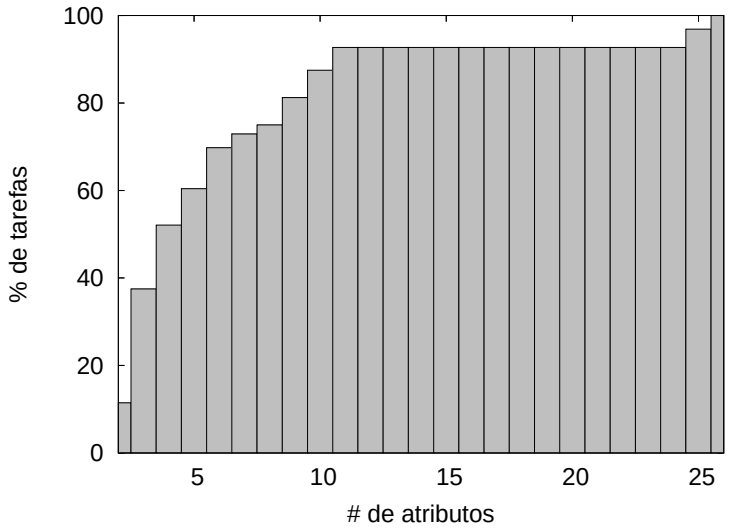


(b) Popularidade de Atributos

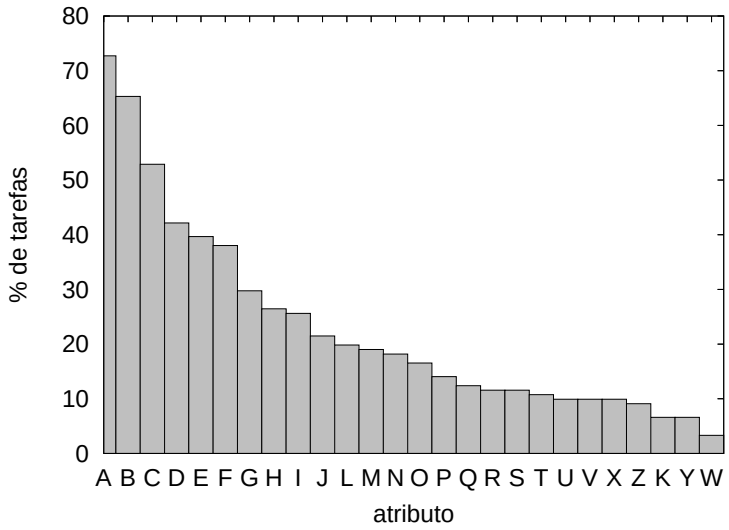
Figura 4.10: Base de Dados Y

de referência em termos dos atributos, o que é uma forte evidência de que um sistema de *cache* poderia melhorar o desempenho do sistema consideravelmente, evitando transferências freqüentes de dados do servidor de dados para o servidor de mineração. Esta caracterização de carga pode ser utilizada como um ponto de partida no desenvolvimento de soluções de *cache*, para gerar carga e para avaliar essas políticas.

Em resumo, caracterizamos os parâmetros das tarefas de mineração, o que mostrou a necessidade de estratégias de *cache* e escalonamento, assim como o comportamento do usuário, que começa minerando dados em um nível mais geral, verificando as evidências mais significativas, e posteriormente refina sua busca minerando os



(a) Atributos por Tarefa(Acumulado)



(b) Popularidade de Atributos

Figura 4.11: Base de Dados Z

dados em um maior nível de detalhe.

4.3 Sumário

Neste capítulo apresentamos os resultados obtidos com a aplicação da nossa proposta de metodologia de caracterização a um sistema real de mineração de dados, o Tamanduá. Grande parte da carga pode ser modelada a partir de distribuições estatísticas, como o processo de chegada de sessão e de tarefa que puderam ser aproximadas a distribuições Weibull, além do USIMG que modelou o comportamento dos

usuários. Através desses modelos, associados aos as demais distribuições de freqüência e popularidade, um gerador de carga pode ser construído para que mudanças nos sistema possam ser avaliados. Além disso, conseguimos agrupar as sessões de usuários em dois grupos: sessões sem paralelismo e sessões com paralelismo. Associando esses dados com as Tabelas 4.8 4.10, temos fortes indícios que o comportamento dos usuários muda muito à medida em que os mesmos vão se familiarizando com o sistema. E por fim, caracterizamos como são utilizadas as bases de dados e seus atributos nas tarefas dos usuários, mostrando que o uso de técnicas de *cache* podem melhorar muito o desempenho de sistema.

Capítulo 5

Conclusões e Trabalhos Futuros

5.1 Conclusões

Diversos estudos de caracterização de serviços Web têm sido publicados ultimamente. Entretanto, a maioria deles foca apenas em pedidos interativos, isto é, pedidos que seguem o protocolo tradicional da requisição-resposta do HTTP. Uma nova tendência para tais serviços é aquela cujos os serviços podem ser providos de forma assíncrona por serem computacionalmente intensivos e não faz sentido obstruir um outro pedido até que a resposta do primeiro esteja pronta. Um exemplo para tais serviços são os serviços de mineração de dados, nos quais a execução das tarefas de mineração é provida de forma assíncrona, enquanto a criação de uma tarefa e a visualização dos resultados acontecem interativamente. Além disso, esta assincronia permite que haja sobreposição entre as várias atividades, em particular na execução de tarefa e em requisições interativas. Compreender a carga de tal aplicação é um passo fundamental para o processo de planejamento de capacidade e de análise de desempenho, bem como do projeto e a implementação de novos mecanismos de sistema, tais como escalonadores e *caching*. Neste trabalho nós propomos e detalhamos uma metodologia de caracterização de carga para tais sistemas com foco em sistemas de mineração de dados. Nossa metodologia é dividida em três níveis onde nós analisamos primeiramente as sessões do usuário, a seguir o comportamento dos usuários dentro de suas sessões, e finalmente os pedidos que compõem uma sessão. Em termos de técnicas de caracterização, nós propomos o USIMG (*User-System Interaction Model Graph*), que é capaz de modelar a execução de atividades simultâneas que podem acontecer durante o uso do sistema.

Nós aplicamos nossa metodologia na caracterização de carga de um sistema atual de serviços de mineração de dados, Tamanduá. Conseguimos modelar grande parte da carga utilizando distribuições estatísticas. Em particular, nós modelamos a du-

ração das sessões de usuários como uma distribuição LogNormal e o tempo entre chegada de sessões e tarefas como uma distribuição Weibull. Nós também modelamos o número de tarefas por sessão utilizando uma *Power Law*. Nós caracterizamos o comportamento dos usuários utilizando nosso modelo proposto, USIMG, no qual foi possível a verificação do impacto da sobreposição entre requisições interativas e não-interativas. Nós também demonstramos o potencial em se aplicar o modelo USIMG conjugado com outras técnicas, em particular técnicas de agrupamento. Mais especificamente, nós encontramos dois grupos muito distintos. O primeiro grupo é composto de usuários que utilizam o sistema de forma seqüencial, isto é, eles sempre esperam até a execução de uma tarefa ser finalizada e correspondem a 68% das sessões, mas apenas 10% das tarefas. O segundo grupo de usuários apresenta diversas situações onde a sobreposição entre requisições interativas e não-interativas acontecem, e como conseqüência disso faziam uso mais intenso do sistema, representando apenas 32% das sessões dos usuários mas 90% das tarefas. Finalmente, nós modelamos os parâmetros relacionados às requisições individuais, em particular as requisições de execução de tarefas, onde encontramos correlações interessantes e tendências, como as encontradas na seqüência na escolha dos parâmetros na mineração por cada usuário. Esses resultados mostram a importância de um bom entendimento das características da carga em atividades que envolvem uma análise de desempenho do sistema.

Nós acreditamos que as contribuições deste trabalho vão além da metodologia e da caracterização de um serviço de mineração de dados se tornando uma direção a ser seguida na caracterização de outros serviços Web que compreendam tarefas interativas e não-interativas. Mais ainda, questões interessantes sobre a evolução dos usuários e seu comportamento já surgiram e se mostram como uma direção promissora de pesquisa, com discutida na próxima Seção.

5.2 Trabalhos Futuros

Durante a elaboração e execução deste trabalho, pudemos vislumbrar um conjunto grande de outras oportunidades de trabalho que podem ser exploradas a partir das contribuições e resultados alcançados nesta dissertação. Nas seções a seguir descrevemos um pouco de cada uma dessas oportunidades.

5.2.1 Caracterização de Outros Algoritmos

Neste trabalho, a caracterização foi realizada utilizando apenas um único algoritmo de mineração de regras de associação disponível na aplicação caracterizada, Tamanduá. No entanto, existe ainda uma variedade muito grande de outros algoritmos não só de mineração de regras de associação, como também de agrupamento e de classificação, além das técnicas de preparação e transformação dos dados.

Cada um desses grupos, apesar de possuírem algumas características em comum, como, por exemplo, serem computacionalmente intensivos e terem o desempenho definido por parâmetros de execução e por uma base de dados, a natureza desses parâmetros e o resultado de suas execuções são bastante distintos. Enquanto as técnicas de transformação de dados preparam os mesmos para uma mineração, sem resultados palpáveis para os usuários, técnicas de agrupamento buscam agrupar os dados com base em algum critério de similaridade (que pode ser escolhido pelo usuário) e as técnicas de classificação procuram descrever modelos que representem as características de um conjunto de dados.

Assim, para que as características específicas de cada um desses grupos de algoritmos e técnicas possam ser estudadas e analisadas, é necessário que a metodologia seja alterada de forma que novos conjuntos de métricas sejam adicionadas ao nível de tarefas de usuários (Seção 3.4.4).

5.2.2 Caracterização de Outros Grupos de Usuários

Atualmente, a ferramenta caracterizada vem sendo utilizada por dois grupos de usuários distintos: estudantes de cursos de mineração de dados; e funcionários de organizações governamentais que procuram fraudes em compras do governo. No entanto, dados os volumes de dados crescentes envolvidos em diversas outras áreas como geoprocessamento e biologia molecular, entre outras, temos que a ferramenta pode ser utilizada nessas áreas como importante apoio no suporte a decisão.

Dessa forma, usuários de outros perfis também podem utilizar o sistema, e uma nova caracterização da ferramenta para esses novos grupos de usuários se faz necessária, uma vez que o interesse dos mesmos pode divergir bastante. Um trabalho interessante nesse sentido seria, portanto, uma análise comparativa entre os grupos de usuários do sistema, observando seus interesses e perfis de uso.

5.2.3 Aplicação da Metodologia em Outros Contextos

A metodologia de caracterização apresentada neste trabalho foi criada com base em uma arquitetura geral de aplicações, apresentada na Seção 3.1, e em arquiteturas

Web Services em geral, no entanto com o foco mais específico em aplicações de mineração de dados.

Web Service está se tornando a arquitetura padrão para o desenvolvimento de aplicações que de alguma forma gerenciam processos computacionalmente intensivos [Dim05] ou de compartilhamento de dados [Ser05]. Existem hoje diversas aplicações que utilizam tecnologias *Web Services* para prover seus serviços, como simulação metereológica, análise de risco em aplicações financeiras [Col05], análise de placentas, microscopia virtual etc. Na Seção 2.1 descrevemos algumas delas.

Assim, trabalhos de caracterização dessas aplicações poderiam ser feitos aplicando a metodologia descrita neste trabalho. Dados os três níveis de caracterização propostos na metodologia, temos que a mesma pode ser muito bem aplicada em outros contextos e outros sistemas com as mesmas características, desde que as métricas apresentadas no nível 3 sejam alteradas no intuito de abordar as características específicas das tarefas de cada aplicação.

5.2.4 Gerador de Cargas Sintéticas

Os resultados obtidos na aplicação da metodologia na caracterização do Tamanduá podem ser aplicados a geradores de carga que retratam exatamente a carga gerada por usuários reais. Além disso, com base nessa caracterização, novas técnicas de escalabilidade para o sistema podem ser propostas e essas propostas podem ser avaliadas utilizando esses geradores de carga.

Em [GGC+05] temos um exemplo real de como uma caracterização pode ser utilizada em uma proposta de melhoria. Esse trabalho apresenta uma proposta de escalonador de processo, avaliado com base em um gerador de cargas que utilizou a caracterização apresentada neste trabalho. Assim como esse, vários outros projetos de melhoria de escalabilidade podem ser realizados baseados nessa caracterização, como, por exemplo, distribuição das bases de dados, pré-alocação de recursos etc.

5.2.5 Análise Temporal do Comportamento dos Usuários

Durante a caracterização do Tamanduá identificamos uma variabilidade no comportamento dos usuários ao longo do tempo no que se refere às tarefas de mineração de dados solicitadas. No entanto, à medida em que os usuários se tornaram mais proficientes no sistema, suas requisições também se tornaram mais elaboradas, apresentando um comportamento semelhante independente do usuário ou do momento no qual os acessos foram feitos.

Mais especificamente, as primeiras tarefas submetidas pelos usuários são tarefas pequenas, utilizando valores altos para suporte e confiança e muitos atributos de uma base de dados. Essa postura se explica porque no início os usuários não entendem muito bem os conceitos de mineração de dados que estão por trás da aplicação. Mas à medida em que esses conceitos vão sendo absorvidos, as tarefas vão se tornando mais elaboradas. Os valores dos parâmetros são refinados e o número de atributos escolhidos para mineração é menor e os atributos mais relacionados. Dessa forma, conseguimos identificar uma série de fatores que podem influenciar no comportamento dos usuários, por exemplo, à medida que os usuários vão se familiarizando com a aplicação, o uso da mesma passa a ser mais intenso. Da mesma forma, à medida em que a aplicação se torna mais eficiente, o utilização do usuário se torna mais elaborada.

Assim, uma boa oportunidade de trabalho a ser realizado é a caracterização levando em consideração os aspectos que influenciam no comportamento dos usuários, apesar de não ser um processo trivial, principalmente quando o objetivo é modelagem desse comportamento. Um exemplo de tal aspecto é o tempo, ou seja, qual o processo evolutivo ao longo do tempo. Uma caracterização feita hoje pode apresentar resultados diferentes se a mesma for feita daqui a alguns meses.

No entanto, esse aspecto exige bastante cuidado por relacionar intrinsecamente todos os outros aspectos que influenciam no comportamento. Esse aspecto pode ser tratado utilizando, por exemplo conceitos de sistemas dinâmicos [1] [Bar97, SJR04, Rou81] e outras técnicas como as utilizadas em mineração de padrões seqüenciais e temporais [HK00].

A partir de uma caracterização de comportamento de usuários levando-se em consideração seu processo evolutivo ao longo do tempo, é possível encontrar modelos mais consistentes onde esse comportamento se torne previsível, permitindo a aplicação de técnicas de aumento de desempenho.

[1] Entende-se por sistema dinâmico qualquer processo que evolua com o tempo.

Apêndice A

Exemplos dos Logs dos Servidores

data	hora	usuario	id sessão	requisição	info
2005-05-31	08:53:34,431	lcrocha	98F3B40	ListaTarefas	-
2005-05-31	08:53:36,600	lcrocha	98F3B40	CriaTarefa	-
2005-05-31	08:53:55,736	lcrocha	98F3B40	SalvaTarefa1	-
2005-05-31	08:53:59,335	lcrocha	98F3B40	SelecionaBase	X
2005-05-31	08:54:11,325	lcrocha	98F3B40	SalvaTarefa2	X [A,B]
2005-05-31	08:54:13,999	lcrocha	98F3B40	SelecionaAlgoritmo	apriori
2005-05-31	08:54:20,280	lcrocha	98F3B40	SalvaTarefa3	(Sup.,0.1),(Conf.,1)
2005-05-31	08:54:20,294	lcrocha	98F3B40	ListaTarefas	-

Tabela A.1: Log do Servidor de Aplicação do Tamanduá

data	hora	id sessão	requisição	id tarefa	tipo
2005-03-29	10:46:47,850	F9A1A84	getParameters	-	ini
2005-03-29	10:47:06,697	F9A1A84	executeTask	1112104026678	ini
2005-03-29	10:47:06,698	F9A1A84	getSupportedAlgorithms	-	ini
2005-03-29	10:47:06,705	F9A1A84	getSupportedAlgorithms	-	fim
2005-03-29	10:53:01,534	F9A1A84	executeTask	1112104026678	fim

Tabela A.2: Log do Servidor de Mineração do Tamanduá

data	hora	id sessão	requisição	tipo
2005-03-29	16:45:41,074	8537F4B	getDatabases	ini
2005-03-29	16:45:41,087	8537F4B	getDatabases	fim
2005-03-29	16:45:41,133	8537F4B	getMetadata	ini
2005-03-29	16:45:41,147	8537F4B	getMetadata	fim

Tabela A.3: Log do Servidor de Dados do Tamanduá

data	hora	id sessão	requisição	id tarefa	tipo
2005-03-29	17:29:30,027	64CD390	detail	1112128130019	ini
2005-03-29	17:29:30,032	64CD390	detail	1112128130019	fim
2005-03-29	17:43:23,242	A33EE22	getMetadata	1112107726122	ini
2005-03-29	17:43:23,252	A33EE22	getMetadata	1112107726122	fim
2005-03-29	17:43:25,265	A33EE22	filter	1112107726122	ini
2005-03-29	17:43:29,718	A33EE22	filter	1112107726122	fim

Tabela A.4: Log do Servidor de Visualização do Tamanduá

Apêndice B

Exemplo do Log Unificado

data	hora	tipo de log	id sessão	login	requisição	tipo	id tarefa
2005-03-29	10:45:53.163	INTERFACE	F9A1A8	lcrocha	ListaTarefas	-	-
2005-03-29	10:46:01.120	INTERFACE	F9A1A8	lcrocha	RemoveTarefa	-	-
2005-03-29	10:46:01.564	MINERACAO	F9A1A8	lcrocha	removeTask	ini	11120
2005-03-29	10:46:01.601	MINERACAO	F9A1A8	lcrocha	removeTask	fim	11120
2005-03-29	10:46:01.703	INTERFACE	F9A1A8	lcrocha	ListaTarefas	-	-
2005-03-29	10:46:02.843	INTERFACE	F9A1A8	lcrocha	CriaTarefa	-	-
2005-03-29	10:46:14.707	INTERFACE	F9A1A8	lcrocha	SalvaTarefa1	-	-
2005-03-29	10:46:14.713	INTERFACE	F9A1A8	lcrocha	SelecionaBase	-	-
2005-03-29	10:46:45.098	INTERFACE	F9A1A8	lcrocha	SalvaTarefa2	-	-
2005-03-29	10:49:09.703	INTERFACE	F9A1A8	lcrocha	ExecutaTarefa	-	-
2005-03-29	10:49:09.772	MINERACAO	F9A1A8	lcrocha	executeTask	ini	11121
2005-03-29	10:49:09.772	MINERACAO	F9A1A8	lcrocha	getAlgorithm	ini	-
2005-03-29	10:49:09.779	MINERACAO	F9A1A8	lcrocha	getAlgorithm	fim	-
2005-03-29	10:49:09.780	MINERACAO	F9A1A8	lcrocha	getParameters	ini	-
2005-03-29	10:49:09.788	MINERACAO	F9A1A8	lcrocha	getParameters	fim	-
2005-03-29	10:49:09.788	MINERACAO	F9A1A8	lcrocha	getArgument	ini	-
2005-03-29	10:49:09.789	MINERACAO	F9A1A8	lcrocha	getArgument	fim	-
2005-03-29	10:49:09.790	MINERACAO	F9A1A8	lcrocha	executeTask	fim	11121
2005-03-29	11:51:41.353	INTERFACE	F9A1A8	lcrocha	Detalha	-	11121
2005-03-29	11:51:42.331	VISUALIZA	F9A1A8	lcrocha	detail	ini	11121
2005-03-29	11:51:42.334	VISUALIZA	F9A1A8	lcrocha	detail	fim	11121
2005-03-29	11:51:42.361	INTERFACE	F9A1A8	lcrocha	Detalha	-	11121
2005-03-29	11:51:58.562	VISUALIZA	F9A1A8	lcrocha	detail	ini	11121
2005-03-29	11:51:58.563	VISUALIZA	F9A1A8	lcrocha	detail	fim	11121
2005-03-29	11:51:58.572	INTERFACE	F9A1A8	lcrocha	Detalha	-	11121
2005-03-29	11:56:47.780	VISUALIZA	F9A1A8	lcrocha	detail	ini	11121
2005-03-29	11:51:58.563	VISUALIZA	F9A1A8	lcrocha	detail	fim	11121

Tabela B.1: Log do Servidor Unificado do Tamanduá

Referências Bibliográficas

[ABB+98] Asmara Afework, Michael D. Beynon, Fabian Bustamante, Angelo Demarzo, Renato Ferreira, Robert Miller, Mark Silberman, Joel Saltz, Alan Sussman, and Hubert Tsang. Digital dynamic telepathology – the virtual microscope. Technical Report CS-TR-3892, University of Maryland, 1998.

[AEM+04] S. R. Amendolia, F. Estrella, R. McClatchey, D. Rogulin, and T. Solomonides. Managing pan-european mammography images and data using a service oriented architecture. In *IDEAS-DH '04: Proceedings of the IDEAS Workshop on Medical Information Systems: The Digital Hospital (IDEAS-DH'04)*, páginas 99–108, Washington, DC, USA, 2004. IEEE Computer Society.

[AGLG04] Nayef Abu-Ghazaleh, Michael J. Lewis, and Madhusudhan Govindaraju. Differential serialization for optimized soap performance. In *HPDC*, páginas 55–64. IEEE Computer Society, 2004.

[AIS93] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, May 1993.

[AJ99] Martin Arlitt and Tai Jin. Workload characterization of the 1998 world cup web site. Technical Report HPL-1999-35(R.1), HP Laboratories Pablo Alto, 1999.

[AKR01] Martin Arlitt, Diwakar Krishnamurthy, and Jerry Rolia. Characterizing the scalability of a large web-based shopping system. *ACM Trans. Inter. Tech.*, 1(1):44–69, 2001.

[AM02] Virgílio A. F. Almeida and Daniel A. Menascé. Capacity planning: An essential tool for managing web services. *IT Professional, IEEE*, 4(4):33–38, 2002.

[Arc04] Web Services Architecture. Web services architecture, 2004. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211.

[AW97] Martin F. Arlitt and Carey L. Williamson. Internet web servers: workload characterization and performance implications. *IEEE/ACM Trans. Netw.*, 5(5):631–645, 1997.

[Bar97] Fernando J. Barros. Modeling formalisms for dynamic structure systems. *ACM Trans. Model. Comput. Simul.*, 7(4):501–515, 1997.

[BHTW03a] Peter Brezany, Juergen Hofer, A Min Tjoa, and Alexander Woehrer. Gridminer: An infrastructure for data mining on computational grids. In *Conference and Exhibition on Advanced Computing, Grid Applications and eResearch*. APAC, 2003.

[BHTW03b] Peter Brezany, Juergen Hofer, A Min Tjoa, and Alexander Woehrer. Towards an open service architecture for data mining on the grid. In *Conference on Database and Expert Systems Applications*. IEEE Computer Science Press, 2003.

[BMPZ02] P. Baglietto, M. Maresca, A. Parodi, and N. Zingirian. Deployment of service oriented architecture for a business community. In *EDOC '02: Proceedings of the Sixth International ENTERPRISE DISTRIBUTED OBJECT COMPUTING Conference (EDOC'02)*, página 293, Washington, DC, USA, 2002. IEEE Computer Society.

[Bor83] Christine L. Borgman. End user behavior on an online information retrieval system: a computer monitoring study. In *SIGIR '83: Proceedings of the 6th annual international ACM SIGIR conference on Research and development in information retrieval*, páginas 162–176. ACM Press, 1983.

[CGB02] Kenneth Chiu, Madhusudhan Govindaraju, and Randall Bramley. Investigating the limits of soap performance for scientific computing. In *HPDC '02: Proceedings of the 11 th IEEE International Symposium on High Performance Distributed Computing HPDC-11 20002 (HPDC'02)*, páginas 246–254. IEEE Computer Society, 2002.

[CJ90] P. Compton and R. Jansen. A philosophical basis for knowledge acquisition. *Knowl. Acquis.*, 2(3):241–257, 1990.

[CKM+03] Francisco Curbera, Rania Khalaf, Nirmal Mukhi, Stefan Tai, and Sanjiva Weerawarana. The next steps in web services. *Communications of the ACM*, 46(10):29–34, 2003.

[Col05] Thomas F. Coleman. Financial engineering on computational clusters, 2005. http://www.fenews.com/fen40/special-feature/coleman.html.

[Com05] MathWorks Company. Matlab-the language of technical computing, 2005. http://www.mathworks.com/products/matlab/.

[CS94] Maria Calzarossa and Giusetembroe Serazzi. Construction and use of multiclass workload models. *Performance Evaluation*, 19(4):341–352, 1994.

[CT03] Mario Cannataro and Domenico Talia. The knowledge grid. *Communications of the ACM*, 46(1):89–93, 2003.

[CTT01] Mario Cannataro, Domenico Talia, and Paolo Trunfio. Knowledge grid: High performance knowledge discovery services on the grid. In Craig A. Lee, editor, *GRID*, volume 2242 of *Lecture Notes in Computer Science*, páginas 38–50. Primavera, 2001.

[CTT02] Mario Cannataro, Domenico Talia, and Paolo Trunfio. Design of distributed data mining applications on the knowledge grid. In *National Science Foundation Workshop on Next Generation Data Mining*, 2002.

[DCJ+05] Gang Dang, Zhi-Quan Cheng, Shi-Yao Jin, Tao Yang, and Tong Wu. A service-oriented architecture for tele-immersion. In *EEE '05: Proceedings of the 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'05)*, páginas 646–649, Washington, DC, USA, 2005. IEEE Computer Society.

[dig02] digiMine. digimine, 2002. http://www.digimine.com/.

[Dim05] Labro Dimitriou. Distributed parallel computing with web services, 2005. http://webservices.sys-con.com/read/48036.htm.

[Dis02] Information Discovery. Information discovery, 2002. http://datamining.aa.psiweb.com/.

[DKB02] D. Dhyani, W. Keong, and N. Bhowmick. A survey of web metrics, 2002.

[DP02] Dan Davis and Manish Parashar. Latency performance of soap implementations. In *CCGRID*, páginas 407–412. IEEE Computer Society, 2002.

[FF03] Christopher Ferris and Joel Farrell. What are web services? *Commun. ACM*, 46(6):31, 2003.

[FGW02] Usama Fayyad, Georges G. Grinstein, and Andreas Wierse, editors. *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann, 2002.

[FJG+05] R. Ferreira, W. Meira Jr., Dorgival Guedes, L. Drummond, B. Coutinho, G. Teodoro, T. Tavares, R. Araújo, and G. Ferreira. Anthill: A scalable run-time environment for data mining applications. In *Proc. of the Symp. on Computer Architecture and High Performance Computing, SBAC*, Rio de Janeiro, Brazil, Outubro 2005. IEEE.

[GGC+05] Luís Fabrício W. Góes, Pedro H. C. Guerra, Bruno Rocha Coutinho, Leonardo Rocha, Wagner Meira Jr., Renato Ferreira, Dorgival O. Guedes, and Walfredo Cirne. Anthillsched: A scheduling strategy for irregular and iterative i/o-intensive parallel jobs. In *Job Scheduling Strategies for Parallel Processing*, The Cambridge Marriott - Kendall Square, Cambridge, MA, Jun. 2005.

[GSC+00] Madhusudhan Govindaraju, Aleksander Slominski, Venkatesh Choppella, Randall Bramley, and Dennis Gannon. Requirements for and evaluation of rmi protocols for scientific computing. In *Supercomputing '00: Proceedings of the 2000 ACM/IEEE conference on Supercomputing (CDROM)*, página 61. IEEE Computer Society, 2000.

[GSC+04] Madhusudhan Govindaraju, Aleksander Slominski, Kenneth Chiu, Pu Liu, Robert van Engelen, and Michael J. Lewis. Toward characterizing the performance of soap toolkits. In *GRID*, páginas 365–372. IEEE Computer Society, 2004.

[Har75] J. Hartigan. Clustering algorithms. *John Wiley and Sons, Inc.*, 1975.

[HK00] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000.

[HTMRG+04] Nt. Humberto T. Marques, Leonardo C. D. Rocha, Pedro H. C. Guerra, Jussara M. Almeida, Jr. Wagner Meira, and Virgilio A. F. Almeida. Characterizing broadband user behavior. In *NRBC '04: Proceedings of the 2004 ACM workshop on Next-generation residential broadband challenges*, páginas 11–18. ACM Press, 2004.

[JvO04] Ion Juvina and Herre van Oostendorp. Predicting user preferences: from semantic to pragmatic metrics of web navigation behavior. In *Proceedings of the conference on Dutch directions in HCI*, página 10. ACM Press, 2004.

[KBTH04] Guenter Kickinger, Peter Brezany, A. Min Tjoa, and Juergen Hofer. Grid knowledge discovery processes and an architecture for their composition. In *IASTED Conference 2004*, fevereiro 2004.

[KGL+03] K.H.Bennett, N. E. Gold, P. J. Layzell, F. Zhu, O. P. Brereton, D. Budgen, J. Keane, I. Kotsiopoulos, M. Turner, J. Xu, O. Almilaji, J. C. Chen, and A. Owrak. A broker architecture for integrating data using a web services environment. In *Service-Oriented Computing, ICSOC 2003*, volume 2910 of *LNCS*, Trento, Italy, Dezembro 2003.

[KS03] Christopher Kohlhoff and Robert Steele. Evaluating soap for high performance business applications: Real-time trading systems. In *WWW*, 2003.

[KZL01a] Shonali Krishnaswamy, Arkady B. Zaslavsky, and Seng Wai Loke. Federated data mining services and a supporting xml-based language. In *HICSS*, 2001.

[KZL01b] Shonali Krishnaswamy, Arkady B. Zaslavsky, and Seng Wai Loke. Towards data mining services on the internet with a multiple service provider model: An xml based approach. *J. Electron. Commerce Res.*, 2(3):103–130, 2001.

[KZL03] Shonali Krishnaswamy, Arkady B. Zaslavsky, and Seng Wai Loke. *Architectural Issues of Web-enabled Electronic Business*, chapter Chapter 7: Internet Delivery of Distributed Data Mining Services: Architectures, Issues and Prospects, páginas 114–128. Idea Group Publishing, 2003.

[MA00] D. Menásce and V. Almeida. *Scaling for E-business: technologies, models, performance and capacity planning.* Prentice Hall, Upper Saddle River - NJ, 2000.

[MA02] D. Menásce and V. Almeida. *Capacity Planning for Web Services.* Prentice Hall, Upper Saddle River - NJ, 2002.

[MHHK99] G. Mahinthakumar, Forrest M. Hoffman, William W. Hargrove, and Nicholas T. Karonis. Multivariate geographic clustering in a metacomputing environment using globus. In *Supercomputing '99: Proceedings of the 1999 ACM/IEEE conference on Supercomputing (CDROM)*, página 5. ACM Press, 1999.

[Mic02] Sun Microsystems. Java remote method invocation, 2002. http://java.sun.com/products/jdk/rmi/index.jsp.

[Mil92] David L. Mills. RFC 1305: Network Time Protocol, 1992.

[NAR+04] Humberto T. Marques Neto, Jussara M. Almeida, Leonardo C. D. Rocha, Wagner Meira Jr., Pedro H. C. Guerra, and Virgilio A. F. Almeida. A characterization of broadband user behavior and their e-business activities. *SIGMETRICS Perform. Eval. Rev.*, 32(3):3–13, 2004.

[NS02] Thomas Nocke and Heidrun Schumann. Meta data for visual data mining. In *Proceedings Computer Graphics and Imaging, CGIM'02, Kaua'i, Hawaii, USA*, 2002.

[Pat02] Norman Paton. Database access and integration services on the grid. Technical Report UKeS-2002-3, National e-Science Centre, 2002.

[PH05] Tony C. Pan and Kun Huang. Virtual mouse placenta: Tissue layer segmentation. In *Proceedings of the 27th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC2005)*, setembro 2005.

[Pit98] James E. Pitkow. Summary of WWW characterizations. *Computer Networks and ISDN Systems*, 30(1–7):551–558, 1998.

[RMP90] R. G. Reynolds, J. I. Maletic, and S. E. Porvin. Pm: A system to support the automatic acquisition of programming knowledge. *IEEE Transactions on Knowledge and Data Engineering*, 2(3):273–282, 1990.

[Rou81] William B. Rouse. Human-computer interaction in the control of dynamic systems. *ACM Comput. Surv.*, 13(1):71–99, 1981.

[SCP04] Paul Stodghill, Rob Cronin, and Keshav Pingali. Performance analysis of a multi-physics simulation system based on web services. citeseer.ist.psu.edu/578395.html, 2004.

[SD02] Tom Soukup and Ian Davidson. *Visual Data Mining: Techniques and Tools for Data Visualization and Mining.* Wiley, 2002.

[Ser05] Delivering Environmental Web Services. Delivering environmental web services, 2005. http://www.resc.rdg.ac.uk/projects.php.

[Shi02] Clay Shirky. Web services and context horizons. *IEEE Computer*, 35(9):98–100, 2002.

[SJR04] Satinder Singh, Michael R. James, and Matthew R. Rudary. Predictive state representations: a new theory for modeling dynamical systems. In *AUAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*, páginas 512–519, Arlington, Virginia, United States, 2004. AUAI Press.

[SN00] Sunita Sarawagi and Sree Hari Nagaralu. Data mining models as services on the internet. *SIGKDD Explor. Newsl.*, 2(1):24–28, 2000.

[Tam05] Projeto Tamanduá. Projeto tamanduá, 2005. http://tamandua.speed.dcc.ufmg.br.

[TKC+04] Kerry L. Taylor, Christine M. O Keefe, John Colton, Rohan Baxter, Ross Sparks, Uma Srinivasan, Mark A. Cameron, and Laurent Lefort. A service oriented architecture for a health research data network. In *SSDBM '04: Proceedings of the 16th International Conference on Scientific and Statistical Database Management(SSDBM'04)*, página 443, Washington, DC, USA, 2004. IEEE Computer Society.

[VJF+04] Adriano Veloso, Wagner Meira Jr., Renato Ferreira, Dorgival Guedes Neto, and Srinivasan Parthasarobegioathy. Asynchronous and anticipatory filter-stream based parallel algorithm for frequent itemset mining. In *Knowledge Discovery in Databases: PKDD 2004, 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Proceedings*, volume 3202 of *Lecture Notes in*

Computer Science, páginas 422–433. Primavera-Verlag GmbH, setembroember 2004.

[VMF+04] A. Veloso, W. Meira Jr., R. Ferreira, D. Guedes, and S. Parthasarathy. Asynchronous and anticipatory filter-stream based parallel algorithm for frequent itemset mining. *European Conference on Principles of Data Mining and Knowledge Discovery*, 2004.

[Wei05a] Eric W Weisstein. Least squares fitting, 2005. http://mathworld.wolfram.com/LeastSquaresFitting.html.

[Wei05b] Eric W Weisstein. Log normal distributionn, 2005. http://mathworld.wolfram.com/LogNormalDistribution.html.

[Wei05c] Eric W Weisstein. Weibull distribution, 2005. http://mathworld.wolfram.com/WeibullDistribution.html.

[WF00] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers Inc., 2000.

[WMSR04] Ted J. Wasserman, Patrick Martin, David B. Skillicorn, and Haider Rizvi. Developing a characterization of business intelligence workloads for sizing new database systems. In Il-Yeol Song and Karen C. Davis, editors, *DOLAP*, páginas 7–13. ACM, 2004.

[Wor05] Scientific Data Intensive Computing Workshop. Scientific data intensive computing workshop, 2005. http://research.microsoft.com/workshops/scidata04/.