DEIVISON LUIZ ARAÚJO

ESTUDO DE UMA ARQUITETURA ORIENTADA A SERVIÇOS APLICADA A UMA PLATAFORMA PARA MINERAÇÃO DE DADOS.

LAVRAS - MG

2011

DEIVISON LUIZ ARAÚJO

ESTUDO DE UMA ARQUITETURA ORIENTADA A SERVIÇOS APLICADA A UMA PLATAFORMA PARA MINERAÇÃO DE DADOS.

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso para a obtenção do título de Bacharel em Ciência da Computação.

Orientador

Prof.Dr. Ahmed Ali Abdala Esmin

LAVRAS - MG

2011

DEIVISON LUIZ ARAÚJO

ESTUDO DE UMA ARQUITETURA ORIENTADA A SERVIÇOS APLICADA A UMA PLATAFORMA PARA MINERAÇÃO DE DADOS.

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso para a obtenção do título de Bacharel em Ciência da Computação.

Aprovada em *06 de Junho de 2011*

Prof. Dr. Denilson Alves Pereira

Prof. Msc. Tiago Amador Coelho

Prof.Dr. Ahmed Ali Abdalla Esmin

Orientador

LAVRAS - MG

2011

Aos meus pais Ivair e Janete.

AGRADECIMENTOS

Agradeço a meus pais, que nunca conteram esforços para eu chegar até aqui, abdicando de sonhos pelo meu crescimento. Importantíssimos na minha vida, vão estar para sempre no meu coração.

A minha irmã Débora e aos meus avós.

A minha namorada Aline.

As grandes amigas: Michele Luiza, Letícia Lara, Melina Andrade e Evellyn Couto.

Aos meu amigos do brejão: Assis Francisco, César Victor, Rafael Chaves, Danilo Nogueira, Joede dos Passos, Tiago Souza, Jeferson Reis, Leo Shigueto, Imbrain Wilker, Régis Carvalho, Jaques Lage, Danilo Batista, Carlos Magno, Rafael Lago, Elicias Santos, Luiz Henrique.

Aos amigos da República Farol Aceso.

Aos companheiros da turma 2007/2 e de trabalho do NTE.

Aos Profs. Ahmed, Tiago e Denilson.

Aos amigos de Divinópolis/Lavras e a todos que não mencionei, mas que torceram ou ajudaram de alguma forma, muito obrigado.

RESUMO

Este trabalho faz um estudo do paradigma SOA (Arquitetura Orientada a Serviços), definido como um paradigma para organização e utilização de competências distribuídas que estão sob controle de diferentes domínios proprietários. Como uma materialização do modelo SOA, é estudado a viabiblidade do padrão de comunicação entre aplicações *Web Services*. Além disso, descreve o modelo da plataforma Web denominada *SwarmMineWeb*, uma plataforma baseada em SOA para integração com uma ferramenta de Mineração de Dados visando interoperabilidade e fácil integração com outras aplicações. Para isso, mostra os passos necessários para a concepção e o desenvolvimento tanto dos serviços, quanto de aplicações-clientes para a plataforma. Para a implementação deste modelo, algumas tecnologias foram utilizadas, tais como: HTTP (*Hyper Text Transfer Protocol*); XML (*Extensible Markup Language*); WSDL (*Web Service Description Language*) juntamente com o UDDI (*Universal Description, Discovery and Integration*); e o SOAP (*Simple Object Access Protocol*). No desenvolvimento foi usado a biblioteca WEKA como base de implementação dos algoritmos de Mineração de Dados, o Framework AXIS2 como implementação SOAP para *Web Services* e o Framework Google Web Toolkit para o desenvolvimento da interface.

Palavras-chave: Mineração de Dados; Arquitetura Orientada a Serviços; Web Services.

ABSTRACT

This work is a study of the SOA paradigm (Service-Oriented Architecture), defined as a paradigm for organizing and utilizing distributed capabilities that may be under control of different ownership domains. As an embodiment of the SOA model, is studied viablity of the standard for communication between Web Services applications. Moreover, describes a model of Web platform called SwarmMineWeb, a SOA-based platform for integration with a Data Mining tool aiming interoperability and easy integration with other applications. To do so, shows the necessary steps for the design and development of both services, and client-applications, for the platform. To implement this model, some technologies were used, such as HTTP (Hyper Text Transfer Protocol), XML (Extensible Markup Language), WSDL (Web Service Description Language) together with UDDI (Universal Description, Discovery and Integration) and SOAP (Simple Object Access Protocol). In development was used the library WEKA as basis for implementation of data mining algorithms, the framework AXIS2 as a SOAP implementation for Web Services and Google Web Toolkit Framework for the development of the interface.

Keywords: Data Mining; Service-Oriented Architecture; Web Services.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

1.1 Contextualização e Motivação

Com a disseminação e popularização do uso de sistemas computacionais, o aumento das ferramentas de coleta automática de dados e o amadurecimento das tecnologias de banco de dados houve um grande crescimento na quantidade de dados nesses bancos. Com isso, a capacidade humana de se interpretar e examinar esses dados é superada, em muito, o que gera, segundo Fayyad et.al.[9], a necessidade de novas ferramentas e técnicas para a análise automática e inteligente de banco de dados.

A partir dessa necessidade surgiu a Mineração de Dados (*Data Mining*); núcleo do processo amplo de KDD (Descoberta de Conhecimento em Bases de Dados do inglês *Knowledge Discovery in Databases*) [16], definida como a aplicação de algoritmos para extração de conhecimento em bases de dados. A aplicação desses algoritmos torna-se uma tarefa dispendiosa computacionalmente quando se trata de grandes bases de dados. O que traz a necessidade de ferramentas que tornam cada vez mais rápidas essa execução.

Além disso, várias aplicações utilizam mineração de dados como auxilio para outras ferramentas. Alguns exemplos disso são o uso em portais de relacionamento como *Orkut*, *Facebook*, *Twitter* e grandes aplicações de e-commerce como a *Amazon*, todas utilizam a mineração de dados em seus sistemas de recomendação.

Entretanto, se para cada nova aplicação for necessário implementar os algoritmos de mineração de dados novamente, muito trabalho repetitivo será feito.

Mesmo com técnicas de reuso de software, várias adaptações relativas a linguagens e plataformas são necessárias.

Para resolver esse problema de retrabalho, estão sendo desenvolvidas técnicas para tornar possível a construção de sistemas que possuam componentes distribuídos pela Web, e que são desenvolvidos independentes de linguagens de programação. Uma nova arquitetura, denominada SOA (Arquitetura Orientada a Serviço do inglês *Service-Oriented Architecture*) [21], para desenvolvimento de software tem sido proposta. SOA tem como principal objetivo o reuso intenso de componentes (serviços) para que a tarefa do desenvolvimento de uma aplicação seja primordialmente a tarefa da composição e coordenação dos serviços já implementados, aumentando o reuso e diminuindo o dispêndio de recursos.

1.2 Natureza da Pesquisa e Objetivos

Uma das primeiras propostas de trabalhos sobre disponibilização de mineração de dados como serviços na Internet foi feita por Sarawagi & Nagaralu [24]. Desde então, muitas pesquisas em torno do assunto têm sido feitas. No trabalho de Guedes et.al.[14] é proposta uma plataforma que visa escalabilidade, extensibilidade, abstração simples para os usuários e alta performance. O trabalho de Wu et.al.[31] descreve uma arquitetura orientada a serviços para inteligência de negócios (BI) que integra tecnologias de BI em um único ambiente, simplificando o tráfego de dados e permitindo análise com baixa latência. Já no trabalho de Albergaria et.al.[1] é proposto um modelo de um módulo extensível a ser acrescentado à interface de um sistema de mineração de dados, permitindo a criação de uma nova interface de mais alto nível que abstrai o conhecimento técnico necessário.

Já Dias & Moita [8], realizam um estudo sobre a adoção de características e vantagens de uma arquitetura orientada a serviços para a construção de sistemas de *Web Mining*. Li & Song [17] tratam da integração de *Web Services* em mineração de dados usando SOA. Essa integração usa a API Java Data Mining (JDM), que é a primeira tentativa de se criar um padrão de API Java para acessar ferramentas de mineração de dados a partir de aplicações Java. Por fim, Talia & Trunfio [26] discutem como as tecnologias de distribuição de dados podem dar suporte a aplicações e serviços de mineração de dados e descoberta de conhecimento. Eles destacam seus trabalhos envolvendo distribuição de dados usando o paradigma de Grid Computing.

O presente trabalho tem como principal objetivo fazer um estudo do paradigma SOA (*Service-Oriented Architecture*) juntamente com o padrão de comunicação *Web Services* (WS). Além disso, mostrar um modelo de uma plataforma integrada baseada em SOA para Mineração de Dados visando interoperabilidade e fácil integração com outras aplicações. Como materialização do paradigma SOA será usado *Web Services* com SOAP, além de mostrar outros padrões utilizados. O trabalho listará todos os passos para modelagem do software, tanto na parte servidor, tanto para criação de um cliente para os serviços criados. Além disso, mostra as vantagens e desvantagens ao modelar um software usando SOA com WS. Portanto esta pesquisa é do tipo tecnológica, uma vez que se utiliza de técnicas existentes em engenharia de software para o desenvolvimento de uma ferramenta que auxiliará na tomada de decisão na gerência e inteligência de negócios na área de Mineração de Dados, como por exemplo, na área de *e-commerce*.

A pesquisa pode ser classificada quanto à natureza como sendo pesquisa aplicada ou tecnológica; quanto aos objetivos como sendo exploratória; quanto à forma de abordar o problema como pesquisa qualitativa; e quanto aos procedimentos técnicos caracteriza-se com base de pesquisa bibliográfica e documental, estudo de caso e modelagem e simulação.

O procedimento utilizado para desenvolver o presente trabalho foi o desenvolvimento experimental de protótipos. Além disso, utilizou-se a pesquisa bibliográfica, como por exemplo, consultas a livros, artigos científicos e outros trabalhos referentes ao tema.

1.3 Materiais e Métodos

Esse trabalho foi desenvolvido no Laboratório de Inteligência Computacional e Sistemas Avançados (LICESA) situado no Departamento de Ciência da Computação (DCC) na Universidade Federal de Lavras (UFLA).

A plataforma de desenvolvimento do trabalho foi um Computador Desktop dotado da tecnologia: Processador Intel® Core 2 Duo com 2,4 Ghz, 4 Gb de memória RAM DDR2. O servidor hospedeiro de testes é um servidor virtual em uma máquina Dell PowerEdge 2900. As principais ferramentas utilizadas na pesquisa foram:

- Servidor Web Apache
- *Framework* AXIS2
- Banco de dados MySQL
- Biblioteca WEKA

- Google Web Tolkit (GWT 2.0)

A ferramenta de desenvolvimento foi o NetBeans 6.8, que acopla em si um ótimo debugger Java, além dos plugins "Axis2 Support"e "GWT4NB"que foram adicionados.

Para o desenvolvimento do trabalho foi utilizada a modelagem de software em camadas - modelo, controle e visualização (MVC) - tornando possível criar uma aplicação baseada no modelo cliente/servidor. Uma aplicação cliente/servidor deixa o pesado processamento com servidores que possuem alto recurso computacional e com o cliente apenas uma interface ou uma aplicação bem mais "leve"; reduzindo consideravelmente o tempo de resposta da aplicação. Como o sistema cliente/servidor a plataforma para Mineração de Dados ganha desempenho, deixando os algoritmos executando em uma máquina mais potente.

O *Framework* Apache AXIS2 [3] foi usado como implementação SOAP dos *Web Services*, e para geração dos documentos WSDL a partir dos serviços Java criados. Além disso, seu pequeno servidor de serviços foi adicionado ao Servidor Apache. Como implementação dos algoritmos de Mineração de Dados usou-se a biblioteca WEKA [29], pois há uma grande quantidade de algoritmos implementados e testados, utilizando as mais avançadas técnicas de otimização, é amplamente divulgada e usada no meio empresarial e acadêmico, de fácil manuseio e de fácil aprendizado para utilização.

Para o desenvolvimento de interfaces gráficas, foram usadas técnicas modernas de conteúdo dinâmico. Para tal, usou-se o framework GWT, onde o código escrito em Java utilizando sua extensa biblioteca de Widgets é compilado para um código Javascript otimizado e compatível com a maioria dos Browsers atuais.

1.4 Conclusões do Capítulo

Neste capítulo foi apresentada a contextualização e pincipais motivações para a realização deste trabalho. Mostra quais foram os principais trabalho correlatos acerca do assunto, qual a natureza e objetivos da pesquisa. Além disso, quais foram os materiais e métodos necessários no desenvolvimento do trabalho.

2 ARQUITETURA ORIENTADA A SERVIÇOS

Segundo Mackenzie et.al.[18], Arquitetura Orientada a Serviços (SOA), é um paradigma para organização e utilização de competências distribuídas que estão sob controle de diferentes domínios proprietários.

Um conceito extremamente importante para entender o SOA é o conceito de serviço. Booth et.al. [4] definem serviço como um recurso abstrato que representa a capacidade de executar tarefas que possuam uma funcionalidade coerente na visão do provedor e do consumidor. Em SOA, serviços são componentes de software que publicaram suas interfaces de uso e que independem de sistema operacional, linguagem ou plataforma. Ou seja, são interoperáveis, o princípio mais importante em SOA para Grossi [13].

Entre as vantagens do uso de SOA, as principais que levaram a sua utilização neste trabalho estão: a fácil adaptação das aplicações a tecnologias em desenvolvimento, alto encapsulamento das funcionalidades, facilidade de criar novos serviços compondo serviços já existentes, fácil integração de aplicações com outros sistemas, reaproveitamento de investimentos e redução de custos nas organizações para cooperação entre elas.

SOA é formada por alguns elementos principais: os serviços propriamente ditos, compostos pelo contrato de acesso, interface e implementação que envolve a lógica de negócios e acessos a banco de dados; o repositório onde ficarão disponíveis os serviços; o protocolo de troca de mensagens e as aplicações clientes que fazem o uso dos recursos disponibilizados pelos serviços da arquitetura. Esses elementos podem ser visualizados na Figura 1.

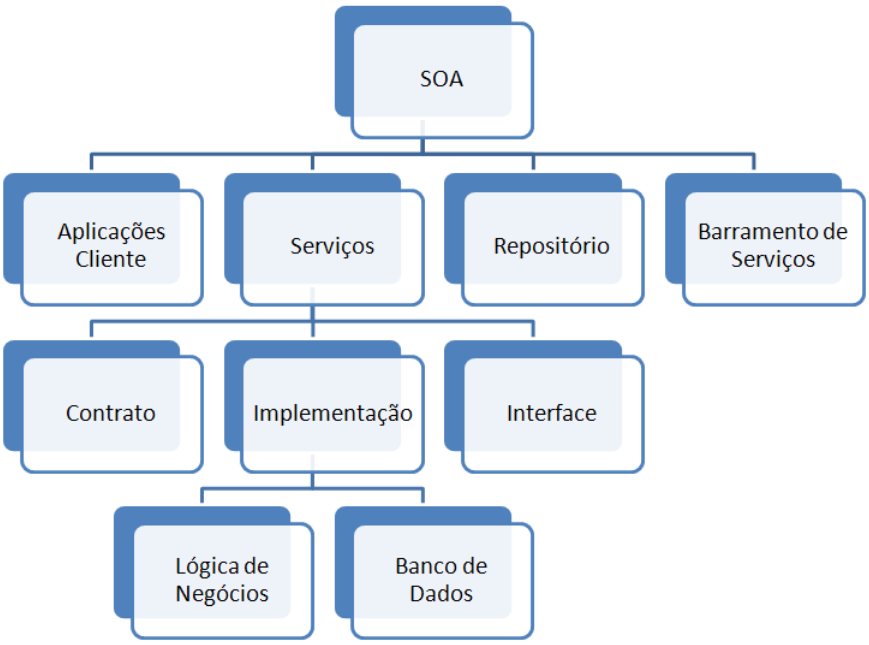

Figura 1: Elementos da arquitetura SOA.

O ponto chave ao desenvolver um sistema nos padrões de uma arquitetura orientada a serviços é definir com que padrão de comunicação serão criados os serviços. Existem algumas formas de se implementar SOA, mas a melhor maneira e mais comumente utilizada é com *Web Services*, segundo os trabalhos de Guedes et.al.[14], Dias & Moita [8] e Li & Song [17]. Apesar disso, SOA é mais do que apenas um conjunto de tecnologias e não está diretamente relacionada com qualquer tecnologia.

Outro fator importante a ser considerado é a granularidade do serviço. Serviços de grão grosso englobam várias funcionalidades para oferecer ao cliente uma maior abstração do problema. Serviços de grão fino são mais específicos

e praticamente representam métodos de acesso a recursos, deixando a cargo do consumidor a implementação da lógica e o controle da interação entre eles.

2.1 *Web Services*

Web Services (WS) é uma forma padrão de comunicação entre diferentes aplicações de software, rodando em uma variedade de plataformas e/ou *frameworks*. Os serviços Web são caracterizados por sua grande interoperabilidade e extensibilidade [4]. É uma tecnologia de integração de sistemas, empregada principalmente em ambientes heterogêneos [12]. O objetivo é desenvolver software ou componentes capazes de interagir com outros softwares, fazendo-os abertos e independentes de qualquer linguagem ou plataforma particular. Para isso, é necessário definir um conjunto de formatos de dados estruturados e legíveis por humanos.

WS surgiu com uma proposta inovadora, como uma evolução de alguns modelos de computação distribuída (RMI, DCOM, CORBA). Além de definir interfaces para componentes de software e como acessá-las através de protocolos padrões e interoperáveis, fornece mecanismos para a descoberta de novos serviços. Um dos principais ganhos com isso é a possibilidade de criar novas aplicações e serviços apenas compondo e combinando serviços já existentes.

Implementar serviços usando WS, com base nos princípios de SOA - que sejam independentes de qualquer arquitetura, fácil processamento, fácil acesso, ampla aceitação - alguns elementos devem ser considerados: um protocolo de transmissão de dados amplamente adotado; um formato de representação de dados independente de arquitetura; uma forma de se exprimir claramente a localização, funcionalidades e interface de acesso de cada serviço; e um protocolo que expresse

a semântica do processo de requisição de serviços e obtenção de respostas que se pretende expressar.

Segundo Gomes [12], atualmente existem dois padrões para o desenvolvimento de *Web Services*. O padrão SOAP, usado neste trabalho, e o padrão REST ou RESTfull.

O SOAP e os outros elementos a serem considerados em um *Web Service* estão descritos e podem ser visualizadas na Figura 2:

Figura 2: Elementos a considerar em um *Web Service*.

- O protocolo HTTP (*Hyper Text Transfer Protocol*) é um protocolo de aplicação responsável pelo tratamento de pedidos e respostas entre cliente e servidor na *World Wide Web*. Seu uso é bem padronizado e amplamente aceito, permitindo a flexibilidade necessária às aplicações *Web*, e segundo Grossi [13], é o protocolo mais indicado para ser usado como canal de transferência de dados em um ambiente distribuído, como um WS.

- O problema do formato de representação de dados é um importante problema a ser resolvido em SOA. É necessário um formato bem definido para

que haja a compreensão universal das mensagens trocadas, independentemente de plataforma ou linguagem. O formato comumente utilizado é o XML. O XML (*Extensible Markup Language*), "é um modo flexível de criar dados autodescritíveis e compartilhar tanto o formato quanto os dados na internet, em intranets e em qualquer outro lugar"segundo Ray [23]. É um formato de texto baseado em marcações, aberto, padronizado pelo *World Wide Web Consortium* (W3C - www.w3c.org), é simples e legível tanto para humanos quanto para computadores, é facilmente editável. Além do principal motivo para sua utilização no WS: é altamente portável [32]. Além disso, segundo Ray [23], o XML é apoiado mundialmente pelas maiores empresas de tecnologia e usado para troca e armazenamento de diversos tipos de dados.

- A descrição do serviço, sua publicação e localização, são fatores importantes a se considerar ao criar um WS. Para isso, o WSDL (*Web Service Description Language* [28]) foi criado. O WSDL é um formato de texto baseado em XML formado por duas partes diferentes: uma abstrata e uma concreta. A parte abstrata descreve a interface dos serviços - define operações, parâmetros e o retorno - ou seja, descreve como o serviço pode ser invocado por seus clientes. Já a parte concreta define o protocolo de comunicação e o endereço onde o serviço estará disponibilizado.

- Para poder encontrar tais descrições é necessário publicá-las em algum repositório de WS. Normalmente tais publicações são feitas utilizando o protocolo UDDI, segundo Grossi [13]. O protocolo UDDI (*Universal Description, Discovery and Integration* [20]) é um diretório global de registros de

Web Services, onde ficam disponibilizadas informações sobre as empresas fornecedoras de serviços. Sua implementação consiste em diversos *Web Services*, que provêem uma interface para que os clientes possam interagir com as informações ali armazenadas.

- Apesar do HTTP ser o responsável pela transferência de recursos na web, é necessário ter um protocolo que possa enviar e receber mensagens contendo documentos XML: é neste ponto que entra o SOAP (*Simple Object Access Protocol* [27]). O SOAP é capaz de descrever completamente uma chamada remota de procedimento, com seus parâmetros, mesmo as mais complexas estruturas de dados [13]. É um protocolo assíncrono baseado em XML para intercâmbio de informações em ambientes distribuídos [5].

2.2 Protocolo SOAP e o Framework AXIS2

SOAP é o protocolo padrão para transmissão de dados dentro da arquitetura de WS proposta pelo W3C. O SOAP é um protocolo baseado no XML e segue o modelo "*REQUEST-RESPONSE*"do HTTP [12]. A Figura 3 ilustra de maneira resumida como funciona a seqüência de interações entre os protocolos para comunicação entre uma aplicação e um *Web Service* usando SOAP. O processo é descrito por Meng et.al.[19]. Em (1) é feita a publicação do WSDL do WS em um repositório UDDI; em (2) o cliente faz a busca no repositório do serviço; em (3) efetua o download do descritor WSDL; em (4) efetua a invocação XML do serviço via protocolo SOAP e recebe a resposta XML do processamento requerido.

Segundo Gomes [12], existem duas formas de solicitar os serviços de um *Web Service* (Figura 4):

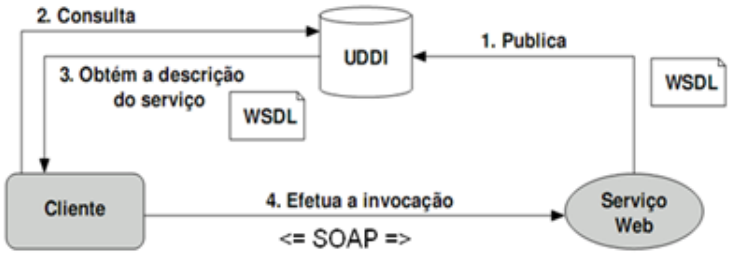

Figura 3: Funcionamento prático dos *Web Services* com SOAP.

- *One-Way Messaging*: mensagem unilateral; nela o servidor recebe a solicitação, processa e não retorna nenhuma resposta ao usuário.
- *Request-Response Messaging*: mensagem bilateral; o cliente faz a requisição, que é processada e uma resposta é enviada pelo servidor. Neste caso podem ocorrer de forma síncrona ou assíncrona.

Figura 4: Modelos *One-Way* e *Resquest-Response Messaging*.[12]

Basicamente o Apache AXIS2 é uma implementação SOAP utilizado para construção de WS. É um framework de código aberto, baseado na linguagem Java e no padrão XML [3].

Possui um servidor de aplicação pequeno e simples. Também é possível, através do AXIS2, gerar automaticamente o arquivo WSDL a partir de uma interface Java e vice-versa. Além da versão para Java, existe uma versão baseada na linguagem C++.

Apesar de ser utilizado o AXIS2 em sua versão para Java, o SOAP em si não está relacionado a nenhuma linguagem. Ele pode ser implementado em qualquer linguagem seguindo os padrões do WS. A escolha do Java como linguagem de desenvolvimento foi motivada, pela facilidade encontrada com o uso AXIS2 juntamente com a biblioteca WEKA, pois ambos já estão escritos em Java. Mas isso não implica que outras linguagens não possam ser usadas futuramente para testar novas implementações.

2.3 Conclusões do Capítulo

Neste capítulo foram abordados os principais tópicos necessários para a leitura desse documento visando um melhor entendimento sobre SOA, quais suas vantagens, características e principais questões a serem consideradas ao utilizar deste estilo de arquitetura de software. Além disso, aborda as questões acerca do seu desenvolvimento usando *Web Services* e o Framework AXIS2.

3 PROCESSO DE DESCOBERTA DO CONHECIMENTO

Segundo Han & Kamber [16], estamos nos afogando em dados, mas morrendo de sede de conhecimento. Ou seja, é eminente a necessidade de transformar tais dados em informação e conhecimento úteis para o suporte à decisão, o que têm demandado investimentos consideráveis da comunidade científica e da indústria de software [25].

Para isso, surge o processo denominado Descoberta de Conhecimento em Bancos de Dados (KDD - *Knowledge Discovery in Databases*), que segundo Fayyad et.al.[9] é o processo não trivial de identificar em dados padrões que sejam válidos, novos, potencialmente úteis e compreensíveis, visando melhorar o entendimento de um problema ou um procedimento de tomada de decisão.

As etapas que compõem um processo de KDD devem ser executadas de forma sequencial. Abaixo elas estão descritas e podem ser visualizadas na Figura 5, adaptada de Han & Kamber [16].

- Limpeza, Integração, Seleção, e Transformação (Pré-Processamento): Esta etapa é responsável pela análise e formatação dos dados para formatos adequados a extração de informações. Para realizar esta formatação é feita a limpeza e redução do volume de dados.

- Mineração de dados: Novos dados, padrões e relacionamentos são obtidos a partir da aplicação de métodos inteligentes sobre os dados já pré-processados.

- Avaliação e Apresentação: Com a execução das técnicas de extração de dados realizadas na etapa de mineração de dados, um alto volume de in-

formações é gerado, contendo dados relevantes ou não, assim nesta etapa é realizada a interpretação, avaliação e apresentação dos conhecimentos descobertos.

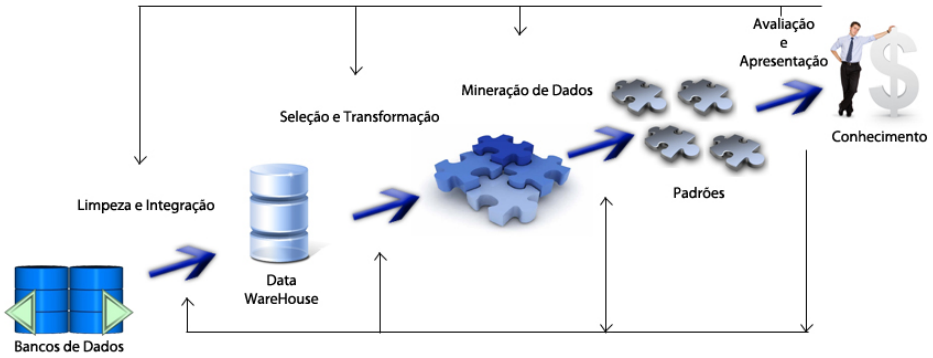

Figura 5: *Knowledge Discovery in Databases* (KDD).[16]

3.1 Mineração de Dados

Mineração de Dados extrai informações implícitas, anteriormente desconhecidas e potencialmente úteis de bases de dados. Estes conhecimentos e informações descobertos são usados por várias aplicações, incluindo análise de marketing, suporte à decisão, detecção de falhas e gerenciamento de negócios [6].

A Mineração de Dados consiste em um conjunto de técnicas reunidas da Estatística e da Inteligência Computacional com o objetivo de descobrir conhecimento novo, útil, relevante e não-trivial que por ventura esteja escondido em uma grande massa de dados [11].

O fato de serem descobertas informações ditas não triviais se dá porque não seriam encontradas ou percebidas por simples sistemas de análise, e as mesmas são de caráter desconhecido até a sua mineração.

As técnicas de Mineração de Dados consistem na especificação de métodos que permitam descobrir padrões de interesse das organizações [2]. Várias ferramentas implementam esses algoritmos de Mineração, um exemplo é a Plataforma WEKA [30, 29], utilizada como base de implementação dos algoritmos.

3.2 Técnicas de Mineração de Dados

De acordo com Dias [7], as técnicas de mineração de dados podem ser aplicadas a tarefas como classificação, estimativa, associação, segmentação e sumarização. Essas tarefas são descritas a seguir:

- Classificação: A tarefa de classificação consiste em utilizar um conjunto de exemplos pré-classificados para desenvolver um modelo que classifica novos dados;

- Estimativa: A estimativa é usada para definir um valor para alguma variável contínua desconhecida. Ela lida com resultados contínuos, enquanto que a classificação lida com resultados discretos;

- Associação: A tarefa de associação consiste em determinar quais itens tendem a co-ocorrerem (serem adquiridos juntos) em uma mesma transação. São utilizadas para encontrar uma descrição compacta para um subconjunto de dados;

- Segmentação: A segmentação é um processo de agrupamento de indivíduos conforme sua semelhança. Particiona uma população heterogênea em vários subgrupos ou clusters mais homogêneos;

- Sumarização: A tarefa de sumarização envolve métodos para encontrar uma descrição compacta para um subconjunto de dados. Um simples exemplo desta tarefa poderia ser tabular o significado e desvios padrão para todos os itens de dados.

A Tabela 1, adaptada de Fonseca [10], mostra de uma forma resumida as técnicas de mineração de dados descritas acima.

3.3 A Biblioteca WEKA

Waikato Environment for Knowledge Analysis - WEKA [29], é um pacote desenvolvido na Universidade de Waikato na Nova Zelândia, em 1993, com o intuito de agregar algoritmos para mineração de dados na área de Inteligência Artificial.

WEKA suporta várias tarefas padrões em mineração de dados, mais especificamente, o pré-processamento, agrupamento, classificação, regressão, visualização, seleção de características [22, 15]. WEKA também fornece acesso a bases de dados SQL usando JDBC e pode processar o resultado retornado por uma consulta à base de dados.

Além disso, o software é licenciado pela *General Public License* sendo, assim, possível a alteração do seu código-fonte feito na Linguagem Java. Devido a todos esses benefícios, a plataforma WEKA foi à escolhida como a base de algoritmos para o desenvolvimento deste trabalho.

Tabela 1: Técnicas de Mineração de Dados.[10]

Tarefa	Descrição	Exemplos
Classificação	Constroi um modelo de algum tipo que possa ser aplicado a dados não classificados a fim de categorizá-los em classes	Classificar pedidos de crédito; esclarecer pedido de seguros fraudulentos; identificar a melhor forma de tratamento de um paciente.
Estimativa (ou Regressão)	Usada para definir um valor para alguma variável contínua desconhecida	Estimar o número de filhos ou a renda total de uma família; estimar o valor em tempo de viada de um cliente; estimar a probabilidade de que um paciente marrerá baseando-se nos resultados de diagnósticos médicos; prever a demanda de um consumidor para um novo produto.
Associação	Usada para determinar quais itens tendem a co-ocorrerem (serem adquiridos juntos) em uma mesma transação	Determinar quais os produtos costumam ser colocados juntos em um carrinho de supermercado.
Segmentação (Agrupamento ou *Clustering*)	Processo de partição de uma população heterogênea em vários subgrupos mais homogêneos	Agrupar clientespor região do país; agrupar clientes com comportamento de compra similar; agrupar sessões de usuários Web par aprever comportamento futuro de usuário.
Sumarização	Envolve métodos para encontrar uma descrição compacta para um subconjunto de dados	Tabular o significado e desvios padrão para todos os itens de dados; derivar regras de síntese.

3.4 Conclusões do Capítulo

Neste capítulo foram abordados os principais tópicos necessários para a leitura desse documento visando um melhor entendimento sobre o Precesso de Descoberta do Conhecimento (KDD), quais suas etapas e características, abor-

dando principalmente a etapa de Mineração de Dados (MD) e suas técnicas. Além disso, apresenta a Biblioteca WEKA, usada no trabalho como fonte de implementação dos algoritmos de MD.

4 RESULTADOS E DISCUSSÃO

Esta seção faz um estudo de caso da Plataforma *SwarmMineWeb*. Mostra primeiramente como a mesma foi criada, listando os passos necessários para modelagem e desenvolvimento. Posteriormente é apresentada uma forma de se criar uma aplicação cliente para utilização dos serviços disponibilizados.

A plataforma *SwarmMineWeb* foi desenvolvida como trabalho de iniciação científica, no departamento de ciência da computação da Universidade Federal de Lavras.

Tem como objetivo fornecer *Web Services* para extração de conhecimento em bases de dados, mais especificamente no processo de aplicação dos algoritmos de mineração de dados. A mesma encontra-se disponível para utilização no endereço *http://licesa.dcc.ufla.br* (*Laboratório LICESA*). A documentação, o manual do usuário e a descrição dos serviços também podem ser encontrados nesse endereço.

4.1 Definição dos Objetivos

O foco principal de uma arquitetura orientada a serviços deve ser na estruturação do software para atender a determinados serviços e não na visão de que será criado um software que irá atender a requisitos de um usuário. Os serviços devem ser bem projetados para atender uma tarefa momentânea. Com isso, foram bem definidos quais objetivos a aplicação iria atender, para posteriormente definir os serviços que serão disponibilizados.

O primeiro passo para criação de uma plataforma nos moldes de um *Web Service* foi definir qual a funcionalidade desejável da plataforma. O foco do tra-

balho foi uma plataforma de para extração de conhecimento em bases de dados, mais especificamente no processo de aplicação dos algoritmos de Mineração de Dados. Foram deixadas de lado outras tarefas de KDD como a coleta, limpeza e pré-processamento de dados, que são tarefas mais específicas a serem aplicadas em banco de dados, o que dificulta a disponibilização da mesma como um serviço.

A plataforma é composta de dois componentes principais: os *Web Services*, que implementam algoritmos de mineração de dados, e uma interface Web com o usuário, que pode ser utilizada para modelagem de aplicações que usam os serviços da plataforma. A Figura 6 ilustra a arquitetura da *Plataforma SwarmMineWeb*.

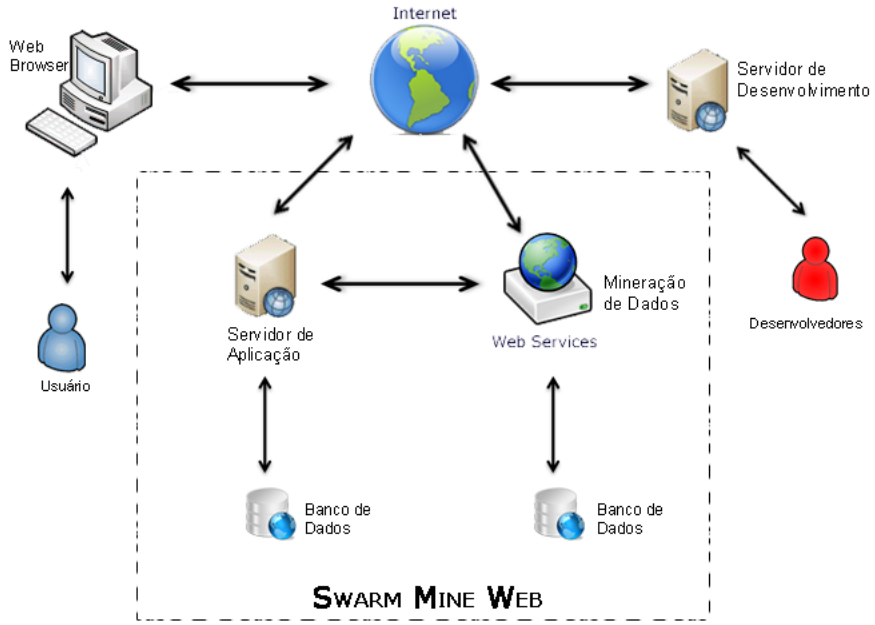

Figura 6: Arquitetura da Plataforma *SwarmMineWeb*.

O trabalho foi desenvolvido usando *Web Services*, juntamente com SOAP, como uma materialização do modelo SOA. Seus serviços foram projetados com uma granularidade fina e com suas requisições no formato "*Request-Response Messaging*", isso para que o cliente tenha total liberdade para usar os serviços da melhor maneira que lhe convenha. O componente responsável pelos serviços fica disponível na Web e pode ser encontrado através do protocolo UDDI, onde estão os contratos WSDL para utilização. Com essa padronização bem definida, qualquer desenvolvedor pode utilizar-se dos serviços facilmente.

Já a interface Web com o usuário é uma opção para que usuários possam experimentar os algoritmos e criar modelos sem a necessidade de se desenvolver uma aplicação. Esta interface usa amplamente todos os serviços da plataforma, além de fazer a composição dos mesmos utilizando-os como um único serviço. Tem uma interface bem amigável podendo ser utilizada por quaisquer usuários, sejam eles acadêmicos ou não.

4.2 Definição das Características

Com a restrição do problema a ser resolvido, a tarefa principal passa a ser definir as características da plataforma SOA criada. Para isso foi feito um estudo do funcionamento da plataforma WEKA, que foi a base de implementação de todos os algoritmos de Mineração de Dados na plataforma.

Foi feita a análise de todo o processo que leva à aplicação dos algoritmos, desde o processo de tráfego de dados, escolha do algoritmo, opções daquele algoritmo e execução. Tarefa essa, essencial para entender como é feita a Mineração

e como ela pode-se ser dividida. Feito isso, foi definido como a arquitetura será modelada para executar os passos do WEKA.

Os dados a serem utilizados pelos algoritmos são transferidos e armazenados em um banco de dados no servidor Web da plataforma, e podem ser reutilizados posteriormente, até que o usuário/desenvolvedor proprietário dos mesmos opte por descartá-los.

Da mesma forma, o usuário/desenvolvedor tem a opção de gerar modelos e salvá-los para posterior reutilização.

A plataforma oferece algoritmos relacionados às técnicas de regras de associação, classificação e agrupamento (*clustering*). O usuário/desenvolvedor escolhe o algoritmo a ser executado, juntamente com suas opções de configuração.

Para atender as funcionalidades que dependem de salvar e reutilizar modelos, dados e resultados de um determinado usuário/desenvolvedor, a utilização dos serviços requer identificação. Para isso, um modo de identificação de usuário foi incorporado ao sistema. O usuário/desenvolvedor deve efetuar um cadastro prévio e efetuar login/logoff ao usar o sistema.

4.3 Definição dos Serviços

Posteriormente, foi feita a definição dos serviços disponibilizados. Cada serviço incorpora uma funcionalidade separada e o conjunto ou composição desses serviços manipulados corretamente forma toda plataforma para Mineração de Dados. Neste ponto é importante definir a granularidade dos serviços disponibilizados. No caso da plataforma, foi definido que seria usado um modelo de granularidade fina. Com isso, quem for usar os serviços terá uma liberdade maior

no controle da interação entre cada serviço. A Tabela 2 apresenta os serviços disponibilizados e uma breve descrição dos mesmos.

4.4 Interface Web com o usuário

A interface Web é uma alternativa para que usuários possam utilizar os serviços de mineração de dados da *SwamMineWeb* sem a necessidade de se escrever código em uma linguagem de programação. Ela fornece através de uma interface amigável abstrações simples para os algoritmos implementados na plataforma. Na atual versão, ela pode ser vista como uma interface Web para parte da biblioteca Weka.

A Figura 7 apresenta um *snapshot* da interface para execução do algoritmo de agrupamento (*clustering*) denominado *Simple K-means*. O usuário pode enviar um arquivo de dados, configurar as opções desejadas do algoritmo, processá-lo e obter os resultados. Para usufruir da aplicação o usuário precisa somente se cadastrar. A parte esquerda da tela apresenta os algoritmos à disposição do usuário.

4.5 Implementação

Para implementação dos serviços Web foi utilizada a linguagem Java. Ela foi escolhida devido a sua maturidade na construção de serviços Web e por possuir ferramentas que facilitam o processo de desenvolvimento. Entretanto, o uso do Java não tira a característica de interoperabilidade, ou seja, a possibilidade da utilização dos serviços em outras linguagens.

Os *Web Services* foram criados utilizando-se o *Framework AXIS2*. Este framework incorpora um servidor SOAP que opera com o servidor *Web Tomcat*.

Tabela 2: Serviços disponibilizados pela *SwarmMineWeb*.

Serviços Disponibilizados	Descrição resumida
String cadastro (login, senha, nome, sobrenome, email)	Cadastra um usuário no sistema retornando uma mensagem de confirmação ou um código de erro.
usuario login (login, senha)	Efetua login e retorna o usuário.
String[] mostrarArquivosSalvos (usuario)	Retorna os nomes dos arquivos de dados do usuário.
String uploadArquivo (nomeArquivo, arquivoBase64, usuario)	Envia um arquivo de dados em formato base64 retornando uma mensagem de confirmação ou um código de erro.
String descartarArquivo (nomeArquivo, usuario)	Descartar o arquivo selecionado pelo usuário. Retorna uma mensagem de confirmação ou um código de erro.
String[] getAlgoritmos (usuario)	Retorna os nomes dos algoritmos suportados.
Algoritmo getAlgoritmo(nomeAlgoritmo, usuario)	Retorna o algoritmo selecionado.
String[][] getOptionsDesc (nomeAlgoritmo, usuario)	Retorna a descrição de cada opção do algoritmo selecionado.
String juntarOpcoes (opcoes, usuario)	Junta as opções do algoritmo, passando para uma String.
String associator (Algoritmo, nomeArquivo, usuario)	Executa um determinado algoritmo de associação. Retorna a string contendo o resultado da mineração.
String classifier (Algoritmo, nomeArquivo, opcao, nomeArquivo2, percent, usuario)	Executa um determinado algoritmo de classificação. Retorna a string contendo o resultado da mineração.
String clusterer (Algoritmo, nomeArquivo, opcao, nomeArquivo2, percent, usuario)	Executa um determinado algoritmo de agrupamento (clustering). Retorna a string contendo o resultado da mineração.
String salvarModelo (usuario)	Salva o último modelo criado pelo usuário. Retorna uma mensagem de confirmação ou um código de erro.
String descartarModelo (Modelo, usuario)	Descartar o modelo selecionado pelo usuário. Retorna uma mensagem de confirmação ou um código de erro.
String utilizarModelo (nomeArquivo, Modelo, usuario)	Utiliza um modelo salvo sobre um arquivo. Retorna o resultado da mineração sobre o arquivo.

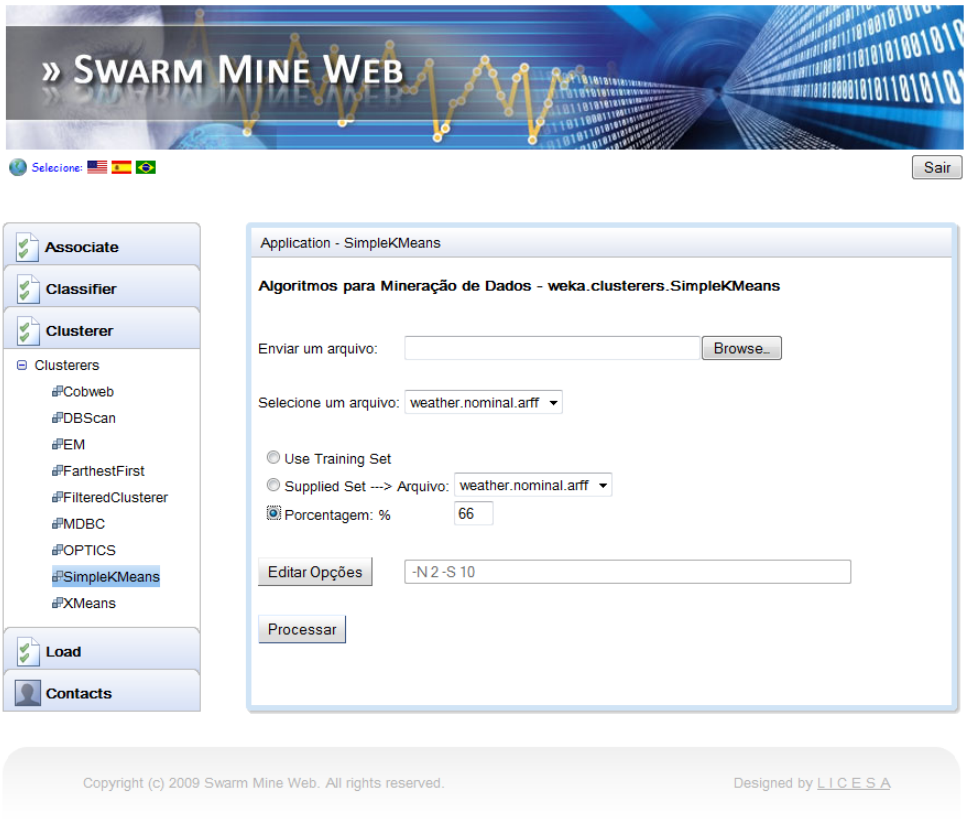

Figura 7: *Snapshot* da interface Web com o usuário.

Também, possui funcionalidades para transformar uma classe Java em um descritor WSDL, além de transformar objetos em arquivos XML automaticamente.

A interface foi feita utilizando o *Framework Google Web Toolkit* (GWT). Seu código, escrito em Java compilado para *Javascript* otimizado, agrega características da Web 2.0, com um conteúdo dinâmico e compatível com a maioria dos navegadores disponíveis atualmente.

4.6 Criando Aplicações-Cliente

Esta seção descreve passo-a-passo como desenvolver aplicações cliente que utilizam os serviços Web na plataforma *SwarmMineWeb*. Um exemplo é apresentado na linguagem Java.

Usando como ambiente de desenvolvimento o NetBeans 6.8, primeiramente deve ser feita a instalação dos plugins necessários. Os três plugins utilizados foram o "Axis2 Support", "GWT4NB"e "WADL Designer".

Após instalar os plugins, deve-se criar o projeto desejado, seja Web ou desktop. No exemplo a seguir, foi criado um projeto Java Desktop. Feito isso, clique no seu projeto com o botão direito do mouse e selecione a opção "Novo / Outro". Na parte categorias selecione "Serviço Web"e nos tipos "Cliente para serviço Web"conforme ilustrado na Figura 8.

Na proxima janela coloque o endereço do arquivo WSDL, selecione o estilo desejado para o cliente como "JAX-WS Style"e confirme as configurações do serviço como mostrado na Figura 9.

Com a criação do cliente para acesso ao serviço web, serão geradas automaticamente as classes necessárias para a utilização da plataforma.

Por último devem ser criadas as classes desejadas para utilizar os serviços, conforme o exemplo mostrado na Figura 10.

Neste exemplo, nas linhas de 1 a 4 são feitas as importações das classes que serão utilizadas. As mesmas fazem parte das classes geradas automaticamente ao criar um cliente para serviço web, conforme descrito acima. Nas linhas 8 e 9, é criada a instância da classe "ServicosPortType"que faz o acesso aos serviços da plataforma. Na linha 11, é feito o login no sistema, passando-se como parâmetros

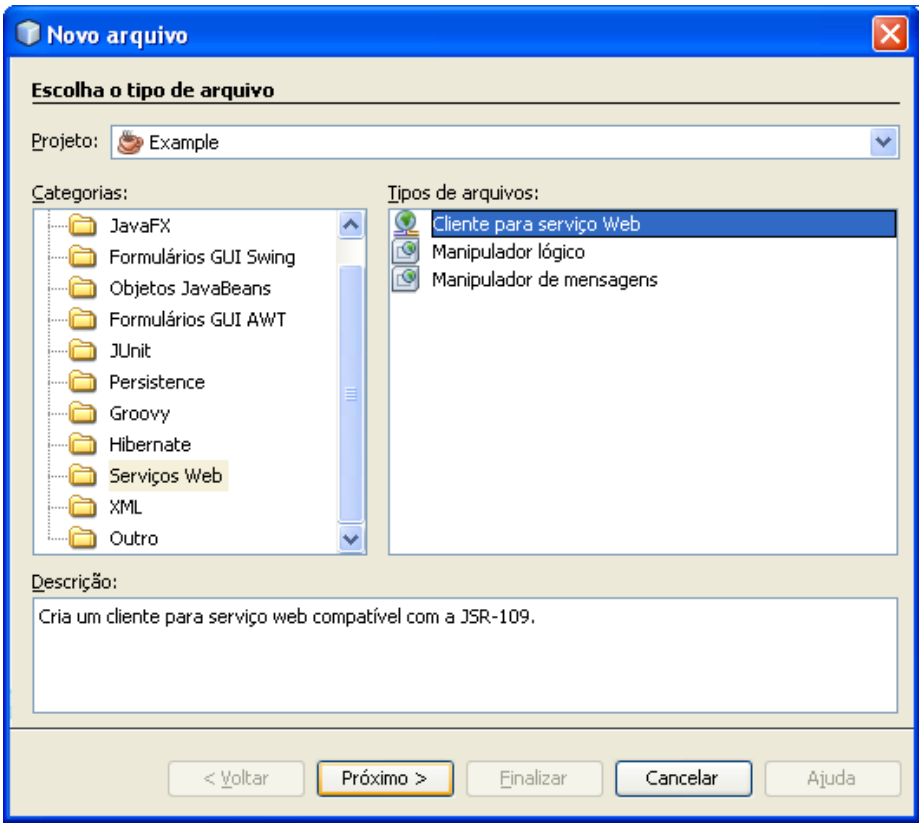

Figura 8: Criando um cliente para serviço web.

o login do usuário e senha, e retornado o objeto User que será usado nos outros serviços. Após feito o login, deve ser escolhido algum algoritmo entre os disponíveis. Nesse exemplo, foi escolhido o algoritmo Apriori, com seus parâmetros padrão, como mostra a linha 12. Para realizar a mineração de dados em si, pode ser usado algum dos três métodos disponíveis: associator, classifier ou clusterer. Para utilizar o algoritmo Apriori, método a ser acessado é o associator (linha 13), passando como parâmetro o algoritmo retornado anteriormente e o nome do ar-

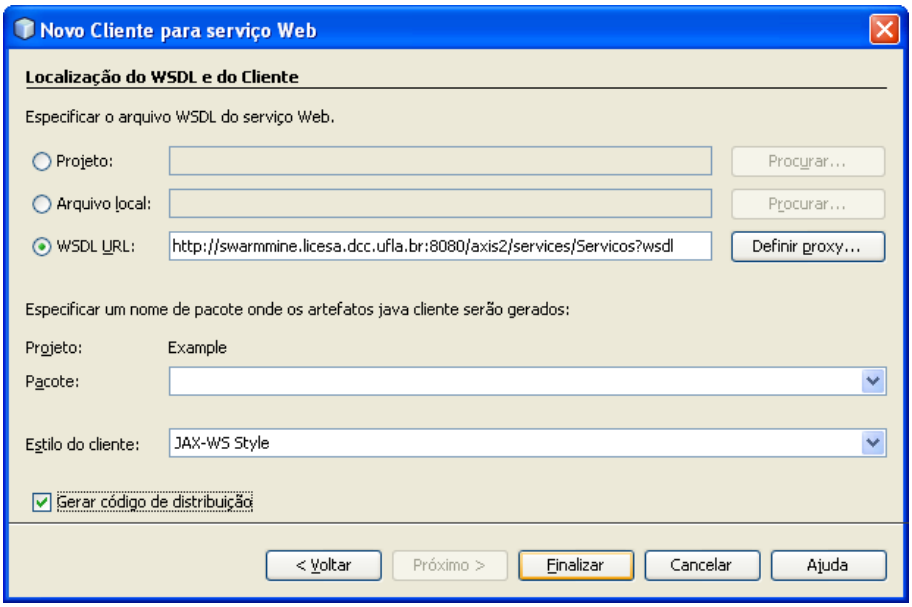

Figura 9: Janela para configuração dos serviços.

```
1  import org.smws.Servicos;
2  import org.smws.ServicosPortType;
3  import org.smws.model.xsd.Algoritmo;
4  import org.smws.model.xsd.User;
5
6  public class ExampleAssociate {
7      public static void main(String[] args) {
8          Servicos serv = new Servicos();
9          ServicosPortType ws = serv.getServicosHttpSoap11Endpoint();
10
11         User u = ws.login("usuario", "senha");
12         Algoritmo alg = ws.getAlgoritmo("weka.associations.Apriori", u);
13         String print = ws.associator(alg, "weather.nominal.arff", u);
14         while (print.contains("<BR>")) {
15             print = print.replace("<BR>", "\n");
16         }
17         System.out.println("Resultado:\n" + print);
18     }
19 }
```

Figura 10: Código de exemplo.

quivo de dados. Neste caso, considera-se que o arquivo de dados já foi enviado ao sistema. O resultado da execução do algoritmo é retornado em forma de uma cadeia de caracteres com tags HTML. No exemplo, o resultado é impresso na saída padrão (linhas 14 a 17), como mostra a Figura 11.

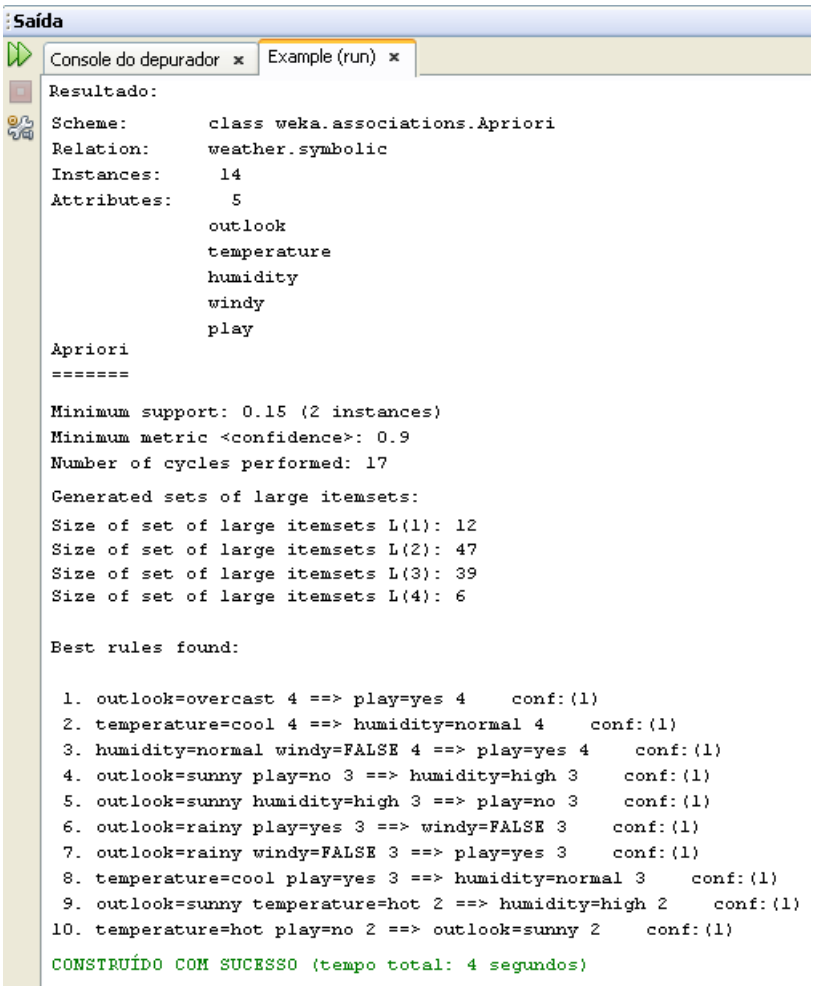

Figura 11: Resultado da Mineração.

5 CONCUSÃO E TRABALHOS FUTUROS

Este trabalho teve como objetivo fazer um estudo do paradigma SOA juntamente com o padrão de comunicação *Web Services*, mostrando um modelo de uma plataforma integrada baseada em SOA para Mineração de Dados visando interoperabilidade e fácil integração com outras aplicações. A motivação principal foi o crescente interesse, das mais variadas áreas do conhecimento, em encontrar padrões interessantes em suas bases de dados. Mas, para isso, a tarefa de Mineração deve ser a mais eficiente possível, em termos de tempo computacional.

Pode-se concluir que os requisitos traçados foram atendidos satisfatoriamente. A plataforma foi desenvolvida com uma aplicação Web nos moldes cliente/servidor, podendo assim o usuário usufruir de uma plataforma para KDD online, com abstrações simples. Com isso, a aplicação das complicadas técnicas de Mineração de Dados pode se concentrar em servidores com alto poder de processamento; já com as máquinas de usuário, ou até mesmo outras aplicações, ficam apenas com o trabalho de envio dos dados, com a apresentação dos resultados, análise, entre outros.

Para oferecer as tarefas de Mineração de Dados como serviços web, foi estudada a viabilidade do paradigma de Arquitetura Orientada a Serviços(SOA). Aonde se chegou à conclusão que SOA agregaria características como: a facilidade de compor serviços, a reusabilidade e principalmente a interoperabilidade. Foi encontrado como bom método para implementar SOA, os *Web Services*, já que este, atendeu bem a todos requisitos.

A plataforma de estudo neste trabalho, a *SwarmMineWeb*, foi feita utilizando algumas ferramentas altamente difundidas. Para implementação dos algorit-

mos de Mineração de Dados, foi utilizada a plataforma WEKA. Para a criação dos *Web Services*, foi utilizada a biblioteca AXIS2. E para criação de uma interface web utilizou-se os recursos do *Framework Google Web Toolkit*.

A modelagem do *SwarmMineWeb* juntamente com a experiência do uso das ferramentas mostradas, demonstraram o grande potencial e a facilidade do uso de *Web Services* como materialização de uma Arquitetura Orientada a Serviços; concluindo que SOA é uma ótima alternativa para criação de sistemas Web que visam principalmente o alto reuso de software independentemente de linguagem e arquitetura.

Como trabalho futuro, é pretendido analisar o desempenho da plataforma usando aplicações reais. Será verificado se a mesma suporta o processamento de computação intensiva em grandes quantidades de dados através de paralelismo maciço. Para isso, será necessário analisar o quanto a plataforma suporta, para posteriormente aplicar técnicas de replicação de servidores e paralelizar a execução de uma tarefa ou então feita a divisão do tratamento de requisições que será processado em cada servidor. Além disso, algumas melhorias podem ser agregadas a plataforma, como um melhor sistema de autenticação dentro da plataforma.

Referências

[1] ALBERGARIA, E., MOURÃO, F., PRATES, R., AND MEIRA JR, W. Modelo de interface extensível como solução para desafios de interação em sistemas de mineração de dados. *XXVIII Congresso SBC* (Julho 2008), 151.

[2] AMO, S. Técnicas de mineração de dados. In *Anais do XXIV Congresso da SBC. Integração e diferenças Regionais: O papel da Computação* (2004), vol. 2, Sociedade Brasileira de Computação.

[3] APACHE. Apache axis2. *http://ws.apache.org/axis2/* (2009).

[4] BOOTH, D., HAAS, H., MCCABE, F., NEWCOMER, E., CHAMPION, M., FERRIS, C., AND ORCHARD, D. Web services architecture. *World Wide Web Consortium - http://www.w3.org/TR/ws-arch* (Fevereiro 2004).

[5] CHAPPELL, D., AND JEWELL, T. Using java in service-oriented architecture. *O'Reilly and Associates Inc* (2002).

[6] CHEN, Y., CHIANG, M., AND KO, M. Discovering time-interval sequential patterns in sequence databases. *Expert Systems with Applications 25*, 3 (2003), 343–354.

[7] DIAS, M. *Um modelo de formalização do processo de desenvolvimento de sistemas de descoberta de conhecimento em banco de dados*. PhD thesis, Universidade Federal de Santa Catarina, 2001.

[8] DIAS, T., AND MOITA, G. Um estudo exploratório sobre uma arquitetura orientada a serviços para sistemas de mineração de dados na web. *VI LAPTEC* (2007). VI Congress of Logic Applied to Technology.

[9] FAYYAD, U., PIATETSKY-SHAPIRO, G., AND SMYTH, P. From data mining to knowledge discovery in databases. *AI magazine 17*, 3 (1996), 37.

[10] FONSECA, E. Track4mine: Uma nova plataforma inteligente de coleta, análise e mineração de dados e interações de usuários na web. *Universidade Federal de Lavras* (2009). Trabalho de Conclusão de Curso.

[11] GOLDSCHMIDT, R., AND PASSOS, E. *Data Mining: Um Guia Prático-Conceitos, Técnicas, Ferramentas, Orientações e Aplicações.*, 1 ed. 2005. Editora Campus, Rio de Janeiro.

[12] GOMES, D. *Web Services SOAP em Java.*, 1 ed. 2010. Editora Novatec.

[13] GROSSI, B. Estudo do modelo de computação orientada a serviços e sua aplicação a um sistema de mineração de dados. Universidade Federal de Minas Gerais, Junho 2005.

[14] GUEDES, D., MEIRA JR, W., AND FERREIRA, R. Anteater: A service-oriented architecture for high-performance data mining. *IEEE Internet Computing* (2006), 36–43.

[15] HALL, M., FRANK, E., HOLMES, G., PFAHRINGER, B., REUTEMANN, P., AND WITTEN, I. The weka data mining software: An update. *ACM SIGKDD Explorations Newsletter 11*, 1 (2009), 10–18.

[16] HAN, J., AND KAMBER, M. *Data mining: concepts and techniques*, 3 ed. Morgan Kaufmann, 2011.

[17] LI, J., AND SONG, B. Web services integration on data mining based on soa. In *Proceedings of the International Symposium on Intelligence Information Processing and Trusted Computing* (2010), IEEE, pp. 532–534.

[18] MACKENZIE, C., LASKEY, K., MCCABE, F., BROWN, P., AND METZ, R. Reference model for service oriented architecture 1.0. *Architecture*, Outubro (2006), 1–31.

[19] MENG, J., MEI, S., AND YAN, Z. Restful web services: a solution for distributed data integration. *Proceedings of the International Conference on Computational Intelligence and Software Engineering* (2009), 1–4.

[20] PAPAZOGLOU, M., AND GEORGAKOPOULOS, D. Service-oriented computing. *Communications of the ACM 46*, 10 (2003), 25–28.

[21] PERREY, R., AND LYCETT, M. Service-oriented architecture. *IEEE CS Press* (2003), 116–119.

[22] PIMENTA, G., CORRÊA, G., FERNANDES, H., AND SERUFO, J. Weka manual. Universidade Federal de Minas Gerais.

[23] RAY, E. *Aprendendo XML*. 2001. Rio de janeiro: Editora Campus.

[24] SARAWAGI, S., AND NAGARALU, S. Data mining models as services on the internet. *SIGKDD Explorations 2*, 1 (2000), 24–28.

[25] SILVA, M. Mineração de dados: Conceitos, aplicações e experimentos com weka. *Sociedade Brasileira de Computação 1* (2004).

[26] TALIA, D., AND TRUNFIO, P. How distributed data mining tasks can thrive as knowledge services. *Communications of the ACM 53*, 7 (2010), 132–137.

[27] W3C. Soap version 1.2 part 1. *World Wide Web Consortium. Disponível em: http://www.w3.org/TR/soap12-part1/.* (2007).

[28] W3C. Web services description language (wsdl) version 2.0 part 0: Primer. *World Wide Web Consortium. Disponível em: http://www.w3.org/TR/wsdl20-primer/.* (2007).

[29] WAIKATO. Weka 3 - machine learning software in java. *University of WAIKATO. Disponível em: http://www.cs.waikato.ac.nz/ml/weka* (2010).

[30] WITTEN, I., AND FRANK, E. *Data mining: practical machine learning tools and techniques*, 3 ed. Morgan Kaufmann, San Francisco, 2011.

[31] WU, L., BARASH, G., AND BARTOLINI, C. A service-oriented architecture for business intelligence. In *Service-Oriented Computing and Applications, 2007. SOCA'07. IEEE International Conference on* (2007), IEEE, pp. 279–285.

[32] YA-QIN, F., AND WEN-YONG, F. Xml in web data mining application. *Proceedings of WASE International Conference on Information Engineering* (2010), 53–56.