Universidade Federal de Alagoas
Instituto de Computação

Dissertação de Mestrado

Um *framework* para mineração de dados educacionais baseado em serviços Web semânticos

TARSIS MARINHO DE SOUZA
tarsis.ms@gmail.com

Maceió, 29 de Abril de 2011

TARSIS MARINHO DE SOUZA

Um *framework* para mineração de dados educacionais baseado em serviços Web semânticos

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre pelo Curso de Mestrado em Modelagem Computacional de Conhecimento do Instituto de Computação da Universidade Federal de Alagoas.

Orientadores:

Prof. Dr. Evandro de Barros Costa

Prof. Dr. Patrick Henrique da Silva Brito

Maceió, 29 de Abril de 2011

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecária Responsável: Helena Cristina Pimentel do Vale

S729u Souza, Tarsis Marinho de.
Um framework para mineração de dados educacionais baseado em serviços semânticos / Tarsis Marinho de Souza. – 2011.
66 f. : il.

Orientador: Evandro de Barros Costa.
Co-Orientador: Patrick Henrique da Silva Brito.
Dissertação (mestrado em Modelagem Computacional de Conhecimento) – Universidade Federal de Alagoas. Instituto de Computação. Maceió, 2011.

Bibliografia: f. 62-66.

1. Mineração de dados educacionais. 2. Framework (Arquitetura de Software). 3. Serviços semânticos. I. Título.

CDU: 004.78:37.018.43

UNIVERSIDADE FEDERAL DE ALAGOAS/UFAL
Programa Multidisciplinar de Pós-Graduação em
Modelagem Computacional de Conhecimento
Avenida Lourival Melo Mota, Km 14, Bloco 09, Cidade Universitária
CEP 57.072-900 – Maceió – AL – Brasil
Telefone: (082) 3214-1364

MODELAGEM COMPUTACIONAL DE CONHECIMENTO

Membros da Comissão Julgadora da Dissertação de Mestrado de Társis Marinho de Souza, intitulada: "Um *Framework* para Mineração de Dados Educacionais Baseado em Serviços Web Semânticos", apresentada ao Programa de Pós-Graduação em Modelagem Computacional de Conhecimento da Universidade Federal de Alagoas em 29 de abril de 2011, às 09h30min, na sala de aula do Mestrado em Modelagem Computacional de Conhecimento.

COMISSÃO JULGADORA

Prof. Dr. Evandro de Barros Costa
UFAL – Instituto de Computação
Orientador

Prof. Dr. Patrick Henrique da Silva Brito
UFAL – Instituto de Computação
Orientador

Prof. Dr. Ig Ibert Bittencourt Santana Pinto
UFAL – Instituto de Computação
Examinador

Prof. Dr. Aydano Pamponet Machado
UFAL – Instituto de Computação
Examinador

Prof. Dr. Seiji Isotani
USP – Instituto de Ciências Matemáticas e de Computação
Examinador

Maceió, abril de 2011.

Agradecimentos

Primeiramente, à Deus por permitir concluir mais uma etapa tão importante em minha vida.

Aos meus pais, Juraci e Miriam, e a minha irmã, Talita, pelo incentivo, apoio incondicional e por nunca medirem esforços para que eu pudesse chegar e concluir mais este objetivo.

Aos meus familiares, especialmente, a minha outra família: Dimitris, Hulda, Dimitri e Júlio Cesar, que me acolheram com tanto carinho e amor e sem os quais, definitivamente, não teria conseguido concluir mais esta etapa.

Aos meus orientadores, Prof. Evandro Costa e Prof. Patrick Brito, pelo apoio, paciência, compreensão e por estarem sempre disponíveis a ajudar em todos os momentos.

Aos grandes amigos que fiz durante o mestrado e que contribuíram muito para que eu pudesse concluir mais esta etapa, especialmente, a Carol, Elvys, Henrique, Higor, Jean, Marcos e Vitor. Não poderia deixar de agradecer ao meu amigo Heitor que contribuiu muito no desenvolvimento deste trabalho.

E, finalmente, à FAPEAL, pelo apoio financeiro.

Resumo

Este trabalho apresenta um *framework* para mineração de dados educacionais baseado em serviços web semânticos como uma solução para os desafios encontrados na área, como: padronização de métodos e dados, integração entre ambientes educacionais e ferramentas de mineração e ferramentas mais intuitivas para o uso de educadores e não especialistas. A solução proposta também inclui um *wizard*, que auxilia na extensão e instanciação do *framework* proposto em diferentes produtos. Para validar a solução proposta, foi realizado um estudo de caso utilizando dados de um ambiente *e-Learning* real utilizado por estudantes de um curso da modalidade a distância.

Palavras-chave: Mineração de dados educacional, *Framework* e Serviços Web semânticos.

Abstract

This paper presents a framework for educational data mining based on semantic web services as a solution to the challenges found in this area such as data and methods standardization, educational environments integration and easier to use mining tools for both educators and non-specialist users. The solution also includes a wizard, which aims to facilitate the extension and instantiation of the proposed framework in different products. In order to validate the proposed solution, a case study was also performed using data from a real e-learning environment used by students from distance mode courses.

Keywords: Educational data mining, Framework and Semantic Web services.

Lista de Figuras

Sumário

Capítulo 1

Introdução

1.1 Contextualização e Problemática

Ambientes *e-Learning*, geralmente, armazenam um grande volume de dados sobre seus usuários, sendo esses coletados automaticamente pelos servidores Web e que gravam todas as ações dos seus usuários enquanto os mesmos navegam pelo ambiente e/ou realizam atividades. Esses dados podem ser uma grande e importante fonte de informação para professores e desenvolvedores de ambientes *e-Learning* para, por exemplo, conhecer detalhes de cada estudante e poder entender melhor suas particularidades por meio da construção de seus perfis.

No entanto, dado o volume de dados armazenado, a análise desses dados em busca dessas informações é uma tarefa complexa e dispendiosa para professores e desenvolvedores. Assim, faz-se necessário o uso de uma abordagem que os auxilie nesse processo. Neste contexto, a mineração de dados apresenta-se como uma boa solução, contribuindo para esta tarefa de análise de dados.

Esses dados podem ser, adequadamente, analisados com a adoção de técnicas de mineração de dados e usados como um suporte na geração de informação relevante tanto para os desenvolvedores quanto para professores. A utilização de métodos de mineração de dados e aprendizagem de máquina para explorar dados provenientes de um contexto educacional é denominada *Educational Data Mining* [8].

A mineração de dados no contexto educacional tem sido utilizada para, por exemplo, identificar as dificuldades que os alunos podem ter durante o processo de aprendizagem [40], melhorar a estrutura dos ambientes *e-Learning* e seus cursos [64, 57], proporcionar aos alunos apoio individualizado[34, 65, 58], entre outros.

Para realizar a mineração de dados em ambientes *e-Learning* pode-se utilizar diferentes abordagens. As principais abordagens encontradas na literatura são: i) Utilizar ferramentas de mineração não direcionadas a educação, o usuário pode utilizar *frameworks* de mineração de dados ou desenvolver/adaptar algum algoritmo de mineração de dados.

Nesta abordagem, o usuário precisa definir os objetivos, definir e implementar/utilizar técnicas de mineração, implementar o pós-processamento, refinar os resultados e adaptar o ambiente. Para utilizar tal abordagem o usuário precisa desenvolver sua própria solução e integrar esta solução ao ambiente *e-Learning*; ii) O uso de ferramentas de mineração de dados direcionadas ao contexto educacional, o usuário pode utilizar ferramentas de mineração de dados para obter informações educacionais relevantes. Nesta abordagem, o próprio usuário utilizará os resultados obtidos pela ferramenta de forma não-automática, ou seja, o usuário deve analisar os resultados obtidos - como a descoberta de padrões pedagógicos relevantes - para tomar decisões, como recomendar um determinado conteúdo a um estudante. Mas utilizar esta abordagem tem um alto custo para o usuário, pois os resultados da mineração não podem ser aplicados diretamente ao ambiente. Na literatura, contudo, pode-se encontrar ferramentas de mineração de dados educacional que permitem aplicar os resultados obtidos diretamente aos ambientes *e-Learning*. No entanto, estas ferramentas são desenvolvidas para ambientes específicos e são uteis apenas a usuários de ambientes semelhantes. Os principais usuários envolvidos nas abordagens citadas são: desenvolvedor de ambientes *e-Learning* e/ou professor.

Romero e Ventura [49] destacam desafios para mineração de dados educacional: i) Padronização dos métodos e dados, onde as ferramentas e técnicas são úteis apenas aos seus desenvolvedores. Não existem ferramentas gerais ou reutilização de ferramentas e/ou técnicas que possam ser aplicadas a qualquer ambiente educacional. Assim, a padronização dos dados e técnicas são necessárias; ii) Integração com ambientes *e-Learning*, Ferramentas precisam ser integradas a ambientes educacionais de forma simples. Todas as tarefas relacionadas à mineração devem ser executadas como uma única atividade, para que *feedbacks* e resultados obtidos possam ser aplicados diretamente nos ambientes; iii) Ferramentas de mineração mais fáceis para uso de educadores e usuários não especialistas. Pois, dentre sua maioria, tais ferramentas são complexas demais para educadores e usuários não especialistas e vão além do escopo que esses usuários podem querer fazer. Essas ferramentas deveriam ter uma interface mais intuitiva e fácil de usar, sem a necessidades de definir parâmetros para algoritmos de mineração para simplificar a configuração, execução e a visualização dos resultados.

1.2 Objetivos

Este trabalho propõe um *framework* baseado em serviços Web semânticos para prover mineração de dados educacional a ambientes *e-Learning*. Este *framework* disponibiliza serviços Web semânticos que encapsulam algoritmos de pré-processamento, algoritmos de mineração e pós-processamento. Com isso, pretende-se alcançar os seguinte objetivos específicos:

- Reuso de ferramentas e/ou técnicas de mineração de dados, incluindo algoritmos de pré-processsamento, mineração e pós-processamento;

  Redução do custo de desenvolvimento de aplicações para o uso da mineração em ambientes *e-Learning*

- Fácil integração com ambientes *e-Learning*;

  Permitir que resultados alcançados com a mineração possam ser aplicados diretamente aos ambientes;

  Possibilitar que qualquer ambiente *e-Learning* possa utilizar as técnicas de mineração de dados para fins educacionais;

- Interface simples;

  Simplificar o processo de extensão do *framework* proposto;

  Redução dos custos do processo de instanciação do *framework*

1.3 Estrutura da Dissertação

O presente trabalho está organizado como se segue. No capítulo 2 é apresentado o referencial teórico, onde são apresentados os conceitos utilizados para o desenvolvimento deste trabalho. No capítulo 3 são apresentados os trabalhos relacionados, a fim de fazer um comparativo entre as principais soluções disponíveis e a solução proposta. A solução proposta é apresentada em detalhes no capítulo 4, onde são apresentados os requisitos da solução, bem como sua arquitetura e as decisões de projeto tomadas na construção para sua construção. Buscando validar a solução proposta, é descrito no capítulo 5 o estudo de caso realizado que descreve passo-a-passo o processo de extensão e instanciação do *framework* proposto para execução da mineração de dados educacionais, realizada com dados provenientes de um ambiente *e-Learning* real, para avaliar a motivação dos estudantes, além de promover a discussão acerca dos desafios na mineração de dados educacionais, destacando as contribuições da solução proposta para esses desafios. Por fim, são apresentadas as considerações finais.

Capítulo 2

Referencial Teórico

Neste capítulo são apresentados, sucintamente, os principais tópicos que formam a base teórica e contribuem para o entendimento do trabalho aqui proposto. Assim, são discutidos conceitos relacionados a Mineração de dados, *Frameworks*, Representação de variabilidades e Web semântica.

2.1 Mineração de Dados

Mineração de Dados[1] é um passo no processo de Descoberta de Conhecimento em Base de Dados (KDD) [2], que consiste na aplicação da análise de dados e algoritmos de descoberta, produzindo uma enumeração de padrões ou modelos sobre os dados [17].

Entretanto, O termo mineração de dados tornou-se comum e vem sendo muito utilizado. Para muitos pesquisadores, o termo mineração de dados é usado como um sinônimo para *Descoberta de conhecimento*, além de está sendo utilizado para descrever um dos passos do processo de *Descoberta de conhecimento* [30].

Mineração de dados é também conhecido por muitos outros nomes, incluindo: *including knowledge extraction, information discovery, information harvesting, data archeology* e *data pattern processing* [19].

No presente trabalho, será utilizado o termo *mineração de dados* para se referir ao processo de extração de conhecimento, além de também e a etapa de mineração do referido processo por ser um termo comum e muito utilizado entre pesquisadores e usuários.

2.1.1 Processo de Descoberta de Conhecimento

O processo de extração de conhecimento requer a execução de diversas etapas. No entanto, não existe um consenso sobre o número de etapas e suas nomenclaturas. Kurgan e Musilek

[1] Do Inglês *Data Mining*
[2] Do Inglês *Knowledge Discovery in Data Bases*

[31] realizaram um levantamento histórico e apontam os principais modelos de processos e o modelo escolhido e utilizado neste trabalho foi o modelo proposto por Fayyad [18].

Na figura 2.1.1 é apresentado o processo de descoberta de conhecimento e a seguir suas respectivas descrições.

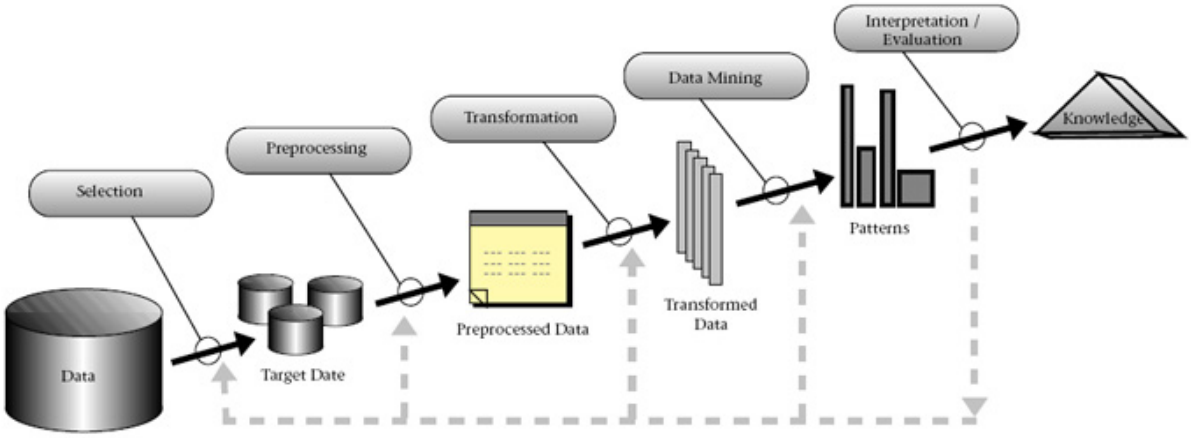

Figura 2.1: Processo de Descoberta de Conhecimento
Fonte: Extraída de [18]

O processo consiste no seguintes passos:

- *Data selection* - Etapa de seleção dos dados relevantes para a análise pretendida
- *Preprocessing* - Etapa que inclui a limpeza dos dados para remover ruídos e inconsistências
- *Data transformation* - Etapa de transformação dos dados para forma apropriada para realização da mineração
- *Data mining* - Principal etapa do processo de descoberta de conhecimento, onde os dados são analisados em busca de padrões relevantes
- *Pattern interpretation/evaluation* - Etapa de interpretação e avaliação dos padrões encontrados

2.1.2 Natureza dos dados

A mineração de dados pode ser aplicada em diversos tipos de bases de dados, como: Banco de dados relacionais, *data warehouse*, dados provenientes da web, entre outros. O foco deste trabalho é a realização da mineração de dados provenientes da web. Desta forma, a mineração web é detalhada abaixo.

Mineração web

Mineração web [3] é o uso de técnicas de mineração de dados para descoberta automática e extração de informação de documentos web e serviços [15].

O uso da mineração web segue um fluxo semelhante a mineração de dados tradicional. Kosala e Blockeel [29] sugerem a decomposição da mineração web em quatro sub-tarefas:

- **Encontrar fonte** - Tarefa de recuperar os documentos web pretendidos;
- **Seleção e pré-processamento da informação** - Tarefa de selecionar e pré-processar os dados provenientes da web;
- **Generalização** - Descoberta dos padrões de um web site específico ou de vários sites;
- **Análise** - Validação e/ou interpretação dos padrões minerados.

Kosala e Hendrik [29] categorizaram a mineração web em três áreas de interesse baseado em qual parte da web minerar: mineração de conteúdo Web, mineração da estrutura Web e mineração do uso da Web.

- **Mineração de conteúdo Web**[4] Processo de descoberta de informações a partir de dados provenientes de conteúdos, dados e documentos Web. Conteúdo Web consiste de muitos tipos de dados como textos, imagens, vídeos, metadados e *hyperlinks*. Conteúdo Web consiste em dados não estruturados, como textos livres, ou semi-estruturados como paginas web.

- **Mineração de estruturas Web**[5] Tentam descobrir modelos subjacentes a estrutura de links da Web. O modelo é baseado na topologia dos hiperlinks com ou sem descrição desses links. Este modelo pode ser usado para categorizar páginas Web e é útil para gerar informação como a similaridade e o relacionamento entre diferentes sites Web.

- **Mineração de uso da Web**[6] tenta fazer sentido dos dados gerados pela navegação Web sessões e comportamentos.

[3] Do inglês *Web Mining*
[4] Do Inglês *Web Content Mining*
[5] Do Inglês *Web Structure Mining*
[6] Do Inglês *Web Usage Mining*

Enquanto a mineração de estrutura e de conteúdo utilizam os dados primários da web, a mineração de uso da web minera os dados secundários derivados das interações do usuários enquanto eles interagem com a Web.

Dados da mineração de uso da Web incluem os dados provenientes dos logs de acesso, logs do servidor de proxy, logs do navegador, perfil de usuário, dados de registro, sessões do usuário ou transações, cookies, clicks do mouse e alguns outros dados das interações do usuário com a Web.

2.1.3 Tarefas de Mineração

Após a definição da natureza dos dados, é necessário definir os objetivos que pretendem ser alcançados com a mineração de dados. Esta é a principal etapa no processo de descoberta de conhecimento. Etapa onde os dados serão analisados em busca de padrões. Nesta etapa, diferentes tipos de padrões podem ser minerados.

Desta forma, é conveniente categorizar a mineração de dados em tipos de tarefas, que correspondem aos diferentes objetivos para as pessoas que estão analisando os dados [22].

As tarefas de mineração podem ser classificadas em duas categorias [18]: a) Descritivas - encontrar padrões interpretáveis por humanos que descrevem os dados; b) Preditivas - o uso de algumas variáveis ou campos para predizer valores desconhecidos ou futuros de outras variáveis de interesse.

As principais tarefas de mineração são descritas abaixo [18, 20, 41, 21]:

Classificação e predição

Classificação é o processo de busca de um modelo que descreva e caracterize classes de dados ou conceitos, com a finalidade de ser capaz de utilizar o modelo para predizer a classe dos objetos cujo o rótulo da classe é desconhecido. O modelo derivado é derivado na análise do conjunto de dados de treinamento, dados cujo rótulo da classe é conhecido [21].

O modelo gerado pode ser representado de diferentes maneiras: Regras, árvore de decisão, entre outras. É apresentado na figura 2.1.3 um modelo gerado e representado através de uma árvore de decisão.

Um exemplo de uso da classificação é: identificar dentro de um grupo de clientes aqueles que são bons e maus pagadores baseado nos perfis dos clientes. Assim, pode-se classificar os novos clientes e definir limites e vantagens nas compras.

Enquanto a classificação prevê categorias (discretas, sem ordem) rótulos, os modelos de predição de funções de valores contínuos. Ou seja, é usada para prever valores numéricos indisponíveis ou que estão faltando ao invés de rótulos de classes [21].

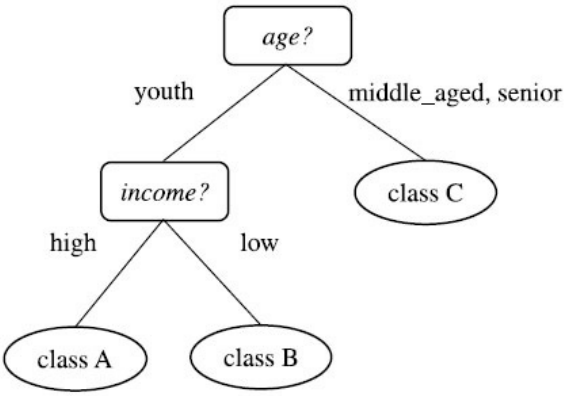

Figura 2.2: Árvore de decisão
Fonte: Extraída de [21]

Agrupamento

O processo de agrupar um conjunto de objetos físicos ou abstratos dentro de uma classe de objetos similares é chamado de agrupamento [21]. Difere do que ocorre com a classificação que analisa os dados baseado nos rótulos das classes, o agrupamento classifica os objetos sem analisar esses rótulos. Os objetos são agrupados baseado no principio da máxima similaridade intraclasse e minima similaridade interclasse [21]. Ou seja, o algoritmo de agrupamento busca reunir na mesma classe objetos que tenham alta similaridade entre eles e que tenha máxima diferença entre objetos das outras classes.

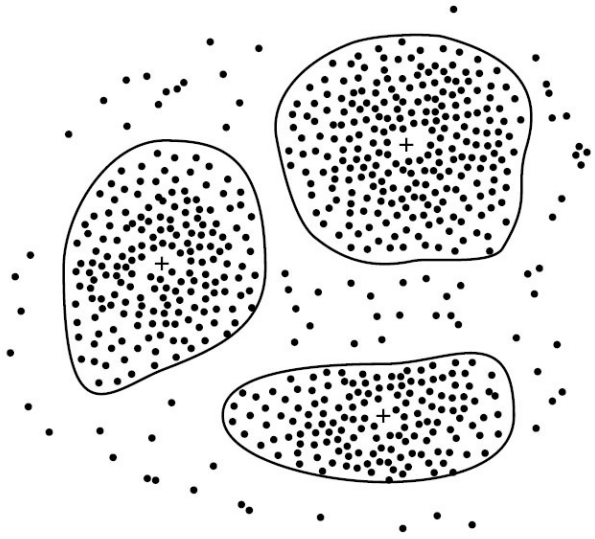

Figura 2.3: Grupos formados a partir dos dados fornecidos
Fonte: Extraída de [21]

Na figura 2.1.3 é apresentado possíveis grupos formados a partir dos dados fornecidos. Esses grupos podem representar um conjunto de clientes que possuem um comportamento de compras similares, por exemplo.

Análise de *Outlier*

Na análise de dados é comum encontrar objetos que não seguem o comportamento geral ou modelo de dados. Esses objetos que se manifestam diferentes ou de maneira inconsistente do restante do conjunto de dados, são chamados de *outliers* [21]. Objetos com esse comportamento, normalmente, são considerados ruídos ou exceções e são descartados. Entretanto, há momentos em que esses dados podem ser mais importantes que os objetos que possuem um comportamento regular. Um exemplo clássico para utilização deste tipo de análise é a detecção de fraude em cartões de crédito.

Os *outliers* podem ser observados na figura 2.1.3, os pontos que estão fora dos grupos podem ser *outliers*. Além dos métodos de agrupamento, a análise de *outliers* pode ser realizada por meio de testes estatísticos, medidas de distância ou métodos baseados em desvio.

Associações, Padrões frequentes

Dado um conjunto de dados, identificar relações entre atributos e itens com a presença de um padrão implica na presença de um outro padrão [20]. Existem alguns tipos de padrões frequentes, alguns deles são:

Mineração de padrões sequenciais.[7] é a mineração de eventos ordenados que ocorrem frequentemente ou subsequentes como padrões [21]. Um exemplo é: clientes que compram uma computador tendem a comprar uma impressora dentro de algumas semanas. Ou seja, são padrões que ocorrem com frequência e em sequencia, neste método a ordem em que esses eventos ocorrem é relevante.

Mineração de conjunto de itens frequentes[8] refere-se a um conjunto de itens que frequentemente aparecem juntos em um conjunto de dados [21]. Um exemplo clássico é: clientes que compram pão também compram leite. Esse padrão busca itens que ocorrem frequentemente, mas a sequencia em que os eventos ocorrem não é levada em consideração.

Normalmente, os algoritmos mineração de padrões frequentes utilizam regras para representar as associações entre os dados. Na figura 2.1.3 é apresentado o exemplo da representação de uma regra.

$$buys(X, \text{“}computer\text{”}) \Rightarrow buys(X, \text{“}software\text{”}) \ [support = 1\%, confidence = 50\%]$$

Figura 2.4: Regra de associação
Fonte: Extraída de [21]

[7] Do Inglês *Sequential pattern mining*
[8] Do Inglês *frequent itemset*

A regra representada na figura 2.1.3 indica que aquele cliente que compra um computador também compra um software. A regra representa também dois parâmetros: *confiança* e *suporte*. Esses parâmetros são utilizados pelo algoritmo para verificar a eficiência da regra e se essa regra representa o conjunto de dados. O *suporte* de 1% indica que a regra aplica-se a 1% dos casos estudados, e uma *confiança* de 50% indica que dos 1% em que regra se aplica, apenas 50% dos casos em que um cliente compra um computador também compra um *software*.

2.1.4 Mineração de Dados Educacional

Mineração de dados educacionais[9] é uma disciplina emergente, preocupada com o desenvolvimento de métodos para exploração de tipos de dados únicos provenientes de contextos educacionais, e usando esses métodos para melhor entender estudantes, e as definições que eles aprendem [8].

A mineração de dados aplicada com enfoque educacional tem algumas diferenças quando relacionada com a mineração de dados tradicional utilizada no comércio eletrônico [49]:

- **Domínio:** O objetivo do comércio eletrônico é orientar os clientes nas compras, enquanto o objetivo do *e-Learning* é orientar os estudantes no processo de aprendizagem [50]

- **Dados:** No comércio eletrônico, os dados utilizados são normalmente simples, como: log de acesso do servidor, mas em e-learning existem mais informações sobre a interação dos estudantes [13]

- **Objetivos:** O objetivo da mineração de dados no comércio eletrônico é aumentar o lucro, que é tangível e pode ser medido em termos de quantidades de dinheiro, o número de clientes e fidelização dos clientes. Já o objetivo da mineração de dados em *e-Learning* é a melhoria da aprendizagem. Esse objetivo é subjetivo e mais sutil medir

- **Técnicas:** Sistemas educacionais têm características especiais que requerem um tratamento diferente dos problemas de mineração tradicionais. Como consequência, algumas técnicas específicas de mineração de dados são necessárias para abordar, em particular, o processo de aprendizagem. [13] [23]

[9]Do Inglês *Educational data mining*

Métodos em Mineração de Dados Educacional

Os métodos mais utilizados na mineração de dados com enfoque educacionais foram caracterizados por Baker [1] e são apresentados abaixo:

- Predição
  - Classificação
  - Regressão
  - Density estimation
- Agrupamento
- Mineração de relações
  - Mineração de regras de associação
  - Mineração de correlações
  - Mineração de padrões sequenciais
  - Mineração de causas
- Destilação de dados para facilitar decisões humano
- Descoberta com modelos

Os Métodos de *predição, agrupamento, mineração de relações* são bem conhecidos e muito utilizados na mineração de dados tradicional. Desta forma, são apresentados abaixo apenas os métodos mais específicos a mineração de dados educacional.

Destilação de dados para facilitar decisões humanas

Uma área de interesse dentro da mineração de dados educacional é a destilação de dados para facilitar decisões humanas. A análise de dados por agentes humanos é realizada se esses dados forem apresentados de forma apropriada;

Os métodos utilizados nesta área da mineração de dados educacionais são métodos de visualização. No entanto, os métodos utilizados na mineração de dados educacional são diferentes dos métodos normalmente utilizados devido a sua estrutura especifica e o significado embutido nos dados educacionais.

A destilação dos dados para facilitar decisões humanas tem dois propósitos principais: a) identificação - os dados são apresentados de forma que humanos possam identificar os padrões mais facilmente, que são difíceis de expressar formalmente; b) Classificação - a destilação de dados pode ser usada também para apoiar a modelos de predição. Neste caso, parte dos dados são exibidos para serem rotulados por humanos. Esses rótulos são

utilizados como base para a construção desses modelos.

Descoberta com modelos

Na descoberta com modelos, o modelo de um fenômeno é desenvolvido por meio de *predição*, *agrupamento*, ou engenharia do conhecimento, em alguns casos (dentro da engenharia do conhecimento, o modelo é desenvolvido utilizando o raciocínio humano ao invés de métodos automatizados). Este modelo é então usado como um componente em outra análise, tais como, previsão de mineração ou de relacionamento.

Aplicação da mineração de dados educacional

É apresentado na figura como normalmente ocorre a mineração com enfoque educacional:

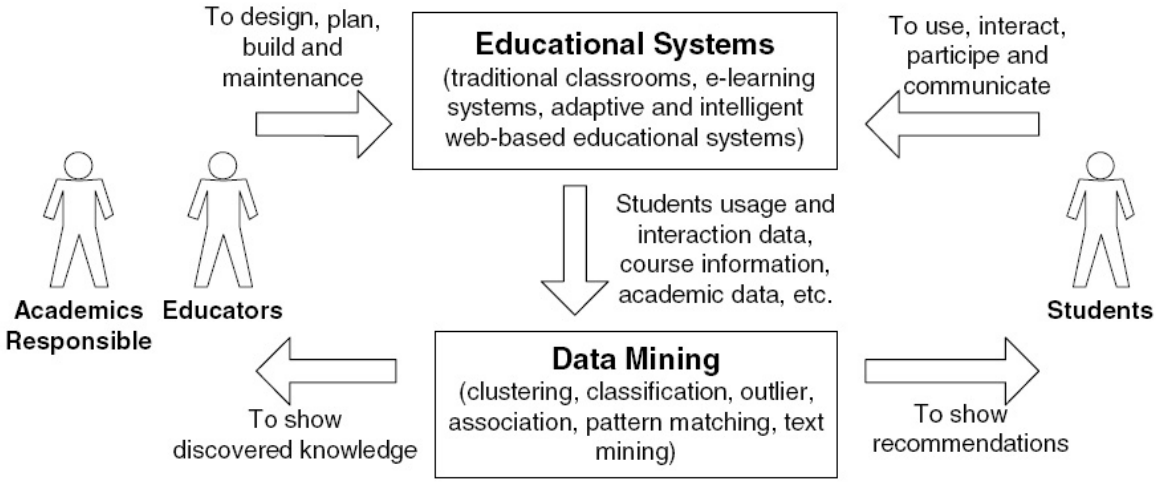

Figura 2.5: Ciclo de aplicação da mineração de dados educacional
Fonte: Extraída de [49]

Educadores e acadêmicos responsáveis projetam, planejam, constroem e mantem os sistemas educacionais. Os estudantes usam e interagem com esses sistemas, o que gera dados de uso e interações dos estudantes. A partir dos dados dos estudantes gerados e informações sobre os cursos disponíveis, diversas técnicas de mineração de dados são utilizadas para descobrir informações úteis. As informações obtidas podem ser utilizadas por estudantes e educadores.

A mineração de dados educacionais tem sido utilizada em: recomendação de conteúdo [42] [65] [58], recomendacao de cursos [51] [48], Análise de sentimento [28], aperfeiçoamento de ambientes *e-Learning* [33] [64] [57], modelgem do estudante [27] [59].

2.2 *Frameworks*

Pode-se encontrar diversas definições de *framework* na literatura, entretanto não há uma definição comum. Duas definições que são bastante referenciadas são reapresentadas por Johnson:

"Um *framework* é um conjunto de classes que captura um projeto abstrato para soluções de uma família de problemas relacionados."[24]

"Um *framework* é um conjunto de objetos que colaboram entre si para assumir um conjunto de responsabilidades de aplicações pertencentes a um domínio de problema específico."[25]

Outra definição bastante utilizada foi apresentada por Fayad:

"O *framework* é o esqueleto de uma aplicação que pode ser adaptado por um desenvolvedor de aplicações."[16]

A partir das definições acima podemos destacar alguns aspectos:

- Reusa o processo de análise e o projeto e não apenas código;
- São aplicações semiprontas, que se destinam a ser usado como base para o desenvolvimento de aplicações em um domínio especifico;

Dentro dos aspectos levantados, Markiewicz e Lucena [35] destacam que para um *framework* possa gerar aplicações dentro de um domínio, deve ter pontos de flexibilidades que possam ser personalizados para atender aos requisitos da aplicação.

Ainda segundo Markiewicz e Lucena [35], Esses pontos de flexibilidades dentro dos *frameworks* são chamados de *hot spots*. Outros pontos que não são facilmente multáveis e que compõem o núcleo do *framework*, também chamados de*frozen spots*.

As principais características presentes nos *framework* são detalhadas abaixo [16]:

- **Modularidade:** Obtida através do encapsulamento de pontos de implementação que são passiveis de mudanças. Este fato, minimiza os impactos gerados e melhora seu desempenho;
- **Reusabilidade:** é alcançada através do uso do projeto de *frameworks* para a construção de novas aplicações. O aumento da produtividade dos programadores, da qualidade e da confiabilidade dos programas gerados são ganhos obtidos através do reuso promovido pelos *frameworks*;

- **Extensibilidade:** o *framework* permite que desenvolvedores possam realizar adaptações: extensão, adição ou implementação de interfaces providas pelo *framework*. Isto permite a criação de uma variedade de aplicações a partir de um mesmo *framework*;

- **Inversão de controle:** Característica fundamental dos *frameworks*. Ele permite que métodos da aplicação construída devem ser invocados em resposta a um evento externo. Esta característica é baseada no principio de *Hollywood*, *"Don´t call us, we'll call* you

  . Ver exemplo na figura 2.2

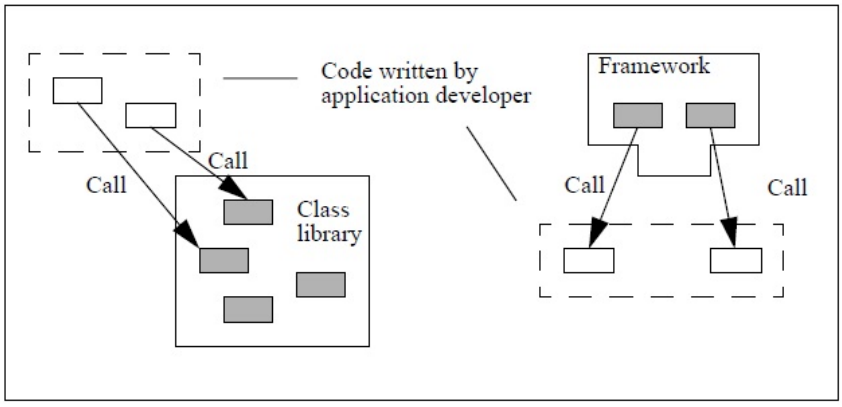

Figura 2.6: Diferença no fluxo de controle entre *frameworks* e bibliotecas de classes
Fonte: Extraída de [32]

Na figura 2.2 é apresentada a diferença de fluxo de controle entre *framework* e uma biblioteca de classes. Na letra (a) da figura acima, um desenvolvedor escreve seu código utilizando uma biblioteca de classes, escolhe quais as classe utilizar e como elas serão invocadas. Na letra (b), um desenvolvedor escreve seu código utilizando um *framework*. Ele estende ou implementa as interfaces necessárias, e o *framework* determina quais as classe e o modo que elas serão invocadas.

2.2.1 Classificação dos frameworks

Nesta seção são apresentadas as duas principais visões sobre a classificação dos *frameworks* encontrada na literatura. Os *frameworks* podem ser classificados quanto ao domínio e quanto a técnica de extensão.

Os *frameworks* classificados quanto ao domínio [55]:

- ***Framework* de Aplicação:** Estes *frameworks* abrangem funcionalidades que podem ser aplicadas em vários domínios. Um exemplo é: *frameworks* de construção de interface gráfica. São conhecidos também como *frameworks* horizontais.

- ***Framework* de Domínio:** Conhecidos também como *frameworks* verticais, encapsulam perícia em um domínio. Oferecem funcionalidades específicas do domínio especificado e são utilizados, normalmente, em diferentes domínios, como: telecomunicações, saúde, finanças, entre outros.

- ***Framework* de Suporte:** Oferecem serviços a nível de sistema, tais como acesso a arquivos, computação distribuída, entre outros. Normalmente, desenvolvedores utilizam estes *frameworks* de suporte diretamente ou modificações produzidas por fornecedores de sistemas. No entanto, mesmo os *frameworks* de suporte podem ser modificados ou adaptados, por exemplo, para o desenvolvimento de um novo produto.

Os *frameworks* também são classificados quanto a técnica de extensão [24] [16]:

- ***Whitebox:*** São conhecidos como *Whitebox frameworks* porque sua estrutura precisa ser entendida antes de ser usada. O comportamento da aplicação especifica é geralmente definida pela adição de métodos para a subclasse de uma ou mais classes. Estes *frameworks* são conhecidos também como *frameworks* dirigidos a arquitetura[55]. Utilizar *whitebox frameworks* têm algumas dificuldades, um problema é que um *whitebox framework* pode ser difícil de aprender a usar, vito que aprender a usá-lo é o mesmo que aprender como ele é construído.

- ***Blackbox:*** Também conhecidos como *frameworks* orientados a dados, os *blackbox frameworks* são conhecidos assim porque sua estrutura não precisa ser entendida para ser utilizada.
  Programadores podem adicionar um conjunto de componentes ao *framework* ao definir as interfaces desses componentes e integrar esses componentes ao *framework*. Normalmente, *Blackbox frameworks* são mais fáceis de serem entendidos e usados do que os *Whitebox framework*, mas são menos flexíveis.

- ***Graybox:*** são *frameworks* projetados para minimizar os problemas encontrados nos *whitebox frameworks* e *blackbox frameworks*. Permitem uma flexibilidade e extensibilidade sem expor informações desnecessárias.

2.2.2 Vantagens e Desvantagens

Antes da opção pelo uso de *framework*, é preciso uma avaliação entre os benefícios e os custos de usar um *framework*. Na figura 2.2.2 é apresentada essa relação entre os custos e os benefícios no uso dos *frameworks*.

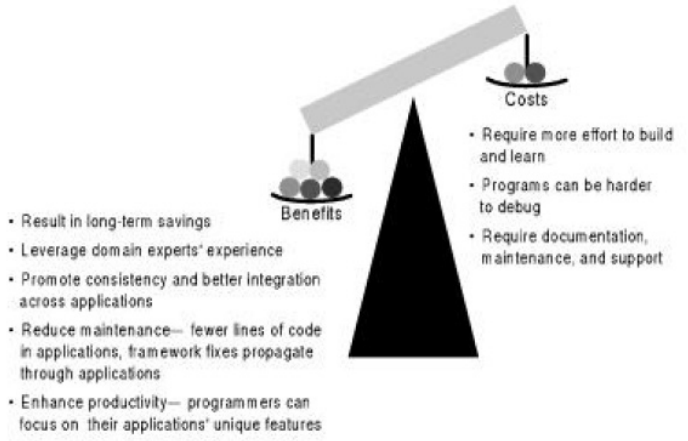

Figura 2.7: Benefícios x Custos
Fonte: Extraída de [55]

As principais vantagens na utilização dos *frameworks* são apresentadas abaixo [32] [9] [37]:

- ***Framework* incorporam conhecimento:** Isto significa que desenvolvedores não precisam se preocupar com os detalhes, e eles podem se concentrar exclusivamente no domínio do problema. [56]. (Foco na área de pericia)

- **Promove consistência e a melhor integração:** Os frameworks incorporam especialidades, os problemas são resolvidos uma vez e as regras de negocio e projeto são usados de maneira consistente. isto permite uma organização construir a partir de uma base que foi verificada para trabalhar no passado. Uma outra vantagem é que aplicações diferentes tem o mesmo "DNA"do sistema. Assim, as aplicações podem trabalhar juntas de maneira mais substancial [56].

- **Reduz a manutenção:** Quando a manutenção é realizada em no *framework*, as aplicações construidas com ele são atualizadas automaticamente.

- **Desenvolvedor tem menos código para escrever:** As abstrações comuns para as aplicações são capturadas pelo *framework*. Desta forma, menos código é necessário para a construção as aplicações que utilizam o *framework*, o que significa uma redução no tempo de desenvolvimento.

- **Confiabilidade:** *frameworks* costuma ter erros e *bugs* como os outros softwares, mas como os *frameworks* são reutilizados, eles tendem a ser mais estáveis e novos erros serão relatados mais raramente. Assim, a reutilização de *frameworks* estáveis aumenta a confiabilidade [32].

As desvantagens na utilização de *frameworks* são detalhadas abaixo [55] [37]:

- **Requer mais esforço para construir e aprender:** uma vez que o desenvolvimento de sistemas complexos é difícil, o desenvolvimento destes sistemas, de forma abstrata, tendo em mente reutilização, é mais difícil ainda [16];
- **Requer documentação, manutenção e suporte:** A documentação do *framework*, é crucial para o usuário do *framework*. Se os *frameworkS* não são suportados pela documentação de usuário necessária, eles provavelmente não serão utilizados. [37];
- **Integrabilidade** [16]: A maioria dos *frameworks* desenvolvidos tem o objetivo de possibilitar a extensão e não a integração entre outros *frameworks*.

2.3 Representação de Variabilidades

2.3.1 Modelo de Características

Um modelo de características[10] representa todos os produtos possíveis de uma linha de produto de software em um único modelo usando características [4]. Usada em diferentes fases do desenvolvimento, bem como na fase de definição da arquitetura.

O primeiro modelo de características foi proposto em 1990 por Kang et al [26]. A notação original foi chamada de FODA [11]. Depois disso, muitas extensões foram propostas, mas não exite um consenso sobre esses modelos.

Um exemplo básico do modelo de características é apresentado na Figura 2.3.1:

O modelo de características básico apresentado acima possui 4 (quatro) tipos de relações:

- **Obrigatório:** Quando uma característica filha aparece obrigatoriamente se a característica pai aparece. Na figura acima, um carro obrigatoriamente possui motor, transmissão e chassi.

[10] Do Inglês *feature model*
[11] Do Inglês *Feature-Oriented Domain Analysis*

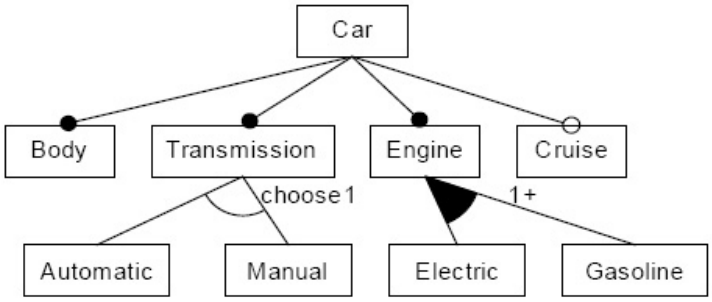

Figura 2.8: Modelo de características
Fonte: Extraída de [4]

- **Opcional:** Quando uma característica filha pode ou não aparecer se a característica pai aparece. O carro pode ser equipado com "cruise control"ou não.
- **Alternativa:** Quando dentre um conjunto de características, uma única característica filha deve aparecer quando a característica pai aparece. O carro pode ter transmissão automática ou manual.
- **Relação-Ou:** Ocorre quando dentre um conjunto de características filhas, uma ou mais características podem ser aparecer quando a característica pai aparece. O carro pode ser equipado com motor elétrico, a gasolina ou os dois ao mesmo tempo.

2.4 Web Semântica

A Web Semântica é uma web de informação derivada de dados através de uma teoria semântica para interpretar os símbolos. A teoria semântica fornece uma explicação do significado em que a conexão lógica de termos estabelece a interoperabilidade entre sistemas. [52]

A Web semântica não é uma Web separada, mas uma extensão da Web atual, em que é dado a informação um significado bem definido, permitindo que computadores e pessoas trabalhem em cooperação. É uma fonte para recuperar informação da Web (usando "redes"da Web a partir de arquivos *RDF*) e o acesso aos dados através de agentes semânticos ou Serviços Web Semânticos. [5].

Esta ideia, no entanto, não é nova, Tim Berners-Lee já articulava sobre a necessidade da semântica na primeira *World Wide Web Conference*, *WWW*, em 1994.

Grande parte do conteúdo disponível na web foi projetada para que humanos lerem, mas não para os computadores. Computadores conseguem processar algumas paginas Web pelo layout definido e/ou links que levam as outras paginas, mas não conseguem entender a semântica dos conteúdos disponíveis.

A Web Semântica trará estrutura ao conteúdo significativo de páginas Web, criando um ambiente onde agentes de software percorram de página em página podendo realizar

facilmente sofisticadas tarefas dos usuários [5].

Tim Berners-Lee apresenta um caso de utilização da semântica na Web:

Lucy precisava marcar uma consulta para sua mãe com um especialista e depois algumas sessões de fisioterapia. Para realizar esta tarefa, Lucy configurou seu agente semântico através de seu computador. O agente recuperou o tratamento prescrito para sua mãe com o agente do médico, checou os o serviços que estavam disponíveis ao plano da mãe de Lucy, e iniciou a busca por um local próximo a sua casa, classificando os serviços disponíveis. Em seguida, ele começou a verificar os horários disponíveis dos serviços e as agendas de Lucy e Peter, e propôs uma solução. A solução apresentada, no entanto, não agradou Peter, pois, teria que dirigir na hora do *rush*. Ele configurou seu próprio agente para refazer a pesquisa com outras preferencias, de tempo e localização. Quase que imediatamente o agente apresentou outro plano, que tinha algumas restrições, mas foi confirmado por Lucy.

Assim, Web Semântica é necessária para expressar informações de forma precisa, podendo tais informações serem interpretadas por máquinas e dessa forma permitirem que agentes de software possam processar, compartilhar e reusar, além de poder entender os termos que estão sendo descritos pelos dados [11]

Tim Berners-Lee apresentou em 2001 o modelo em camadas da Web semântica, conhecido como Bolo de Noiva. No ano de 2007, foi feita uma reavaliação das camadas propostas e apresentada uma nova perspectivas das camadas da Web semântica, que é apresentada na figura 2.9.

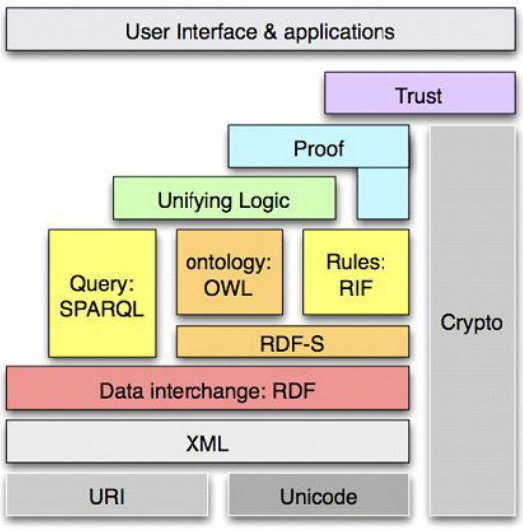

Figura 2.9: Camadas da Web Semântica
Fonte: Extraída de [61]

2.4.1 Ontologias

Para que a Web semântica funcione, computadores devem ter acesso a coleções estruturadas de informação e a conjuntos de regras de inferências que podem ser usados para conduzir um raciocínio automatizado. Representação de conhecimento, como é frequentemente chamada. [52]

As ontologias cumprem essa função. Fornecem o suporte necessário para a representação dessas informações de forma que computadores possam processá-las e assim realizar um raciocino automatizado. Desta forma, as ontologias possuem um papel fundamental na web semântica, pois possibilitam não só humanos possam compreender os conteúdos disponíveis na web, mas também permite que computadores possam acessar e processar tais informações.

O termo ontologia tem origem na filosofia. Uma ontologia é uma teoria sobre a natureza da existência, de que tipos de coisas existem. Em informática, uma ontologia pode ser definida como:

Uma ontologia é uma especificação formal e explícita de uma conceitualização compartilhada [54].

Abaixo segue um detalhamento sobre o conceito de ontologia [54]:

- **Conceitualização**: Um modelo abstrato de algum fenômeno do mundo.
- **Explicita**: Os conceitos usados, bem como as restrições sobre seu uso são, explicitamente, definidos.
- **Formal**: As ontologias podem ser processadas por máquinas.
- **Compartilhada**: As ontologias representam um conhecimento consensual.

Ontologias podem melhorar o funcionamento da Web em muitos aspectos [5]. Como melhorar a precisão dos sistemas de buscas da web, pois esses sistemas podem procurar por paginas que fazem referencias a conceitos preciso e não a palavras-chaves ambíguas. Outro beneficio que pode ser alcançado com uso da web semântica, é o uso de ontologias para anotação semântica de serviços Web[12]. Assim, o processo de descoberta e composição se torna mais preciso e pode ser realizado de forma automática.

2.4.2 Serviços Web Semânticos

De acordo com Payne e Lassila [44], serviços Web semânticos[13] (SWS) são definidos como sendo um aperfeiçoamento das descrições dos serviços Web através do uso de anotações

[12]Do Inglês *Web Services*
[13]Do Inglês *Semantic Web Services*

semânticas, o que possibilita um alto grau de automação na descoberta, composição, monitoramento e invocação. A introdução de semântica aos serviços Web faz com que tais serviços possam ser processados pela máquina. Desta forma, é possível criar de agentes de software capazes de descobrir, compor e invocar os serviços automaticamente. Além disso, a descoberta se torna mais precisa, visto que não é mais feita de maneira sintática, mas sim, semântica.

Os Serviços Web Semânticos são baseados na criação de descrições semânticas para serviços Web tradicionais por meio de ontologias com a finalidade de proporcionar um cenário onde agentes inteligentes tornam-se capazes de explorar as descrições semânticas desses serviços com a proposta de executar tarefas complexas sem a interferência direta de humanos [10].

No caso de Lucy apresentado por Tim Berners-Lee, pode-se verificar que os serviços Web semânticos possuem características dos serviços Web tradicionais, além de apresentarem a habilidade de automatizar processos da Web semântica. É apresentado na Figura 2.10 esta ideia, onde serviços Web semânticos são a união dessas duas tecnologias [7].

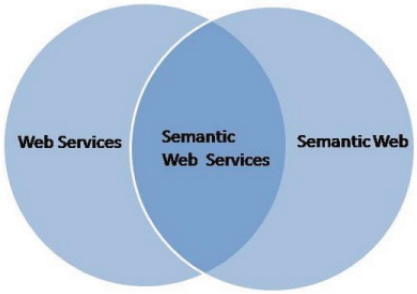

Figura 2.10: Serviços Web Semânticos

O processo de automação acontece durante a etapa de descoberta, composição e invocação dos serviços, isto possibilita que computadores possam realizar o processo sem a necessidade da interferência humana. Desta forma, o uso da Web Semântica permite a criação de sistemas dinâmicos, pois são baseados em serviços que podem ser descobertos e compostos automaticamente durante sua execução. Isto é possível, unicamente, porque o processo de descoberta e invocação dos serviços são automatizados. [38]

SWS funcionam baseados em uma descrição semântica, essas descrições oferecem o suporte para que esses serviços possam ser descobertos, selecionados e compostos de forma automática. Desta forma, as principais tecnologias para anotação semântica são: WSDL-S [63], SWSF [3], WSMO [46] and OWL-S[36].

OWL-S

OWL-S [36], anteriormente conhecido como DAML-S, é um conjunto de ontologias utilizadas para descrever serviços Web semânticos expressos em *Ontology Web Language (OWL)*. OWL-S foi submetida a W3C em 2004, e define uma ontologia superior para descrever semanticamente serviços Web ao longo de três principais aspectos [45]:

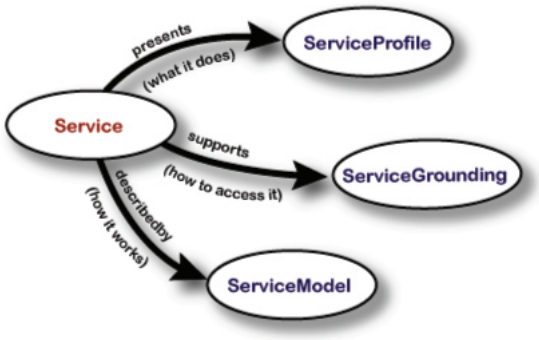

Figura 2.11: OWL-S Ontology
Fonte: Extraída de [6]

- O *Service Profile* descreve o que o serviço faz em termos de entradas, saídas, pré-condições e efeitos *(IOPEs)*;
- O *Service Model* descreve como o serviço funciona em termos de um modelo de processo, que pode descrever um comportamento complexo sobre a "underlying"do serviço
- O *Service Grounding* descreve como o serviço pode ser invocado, normalmente, pelo processamento de um arquivo *WSDL*

A figura 2.4.2 mostra as ontologias que compõem a OWL-S. Em resumo, a construção de descrições para o serviço são baseadas em OWL consistem na criação de instancias das ontologias propostas nessas abordagens.

Capítulo 3

Trabalhos Relacionados

3.1 *A Dynamic Web Service based Data Mining Process System*

Tsai e Tsai [60] propõem um sistema de mineração baseado em serviços Web, onde cada atividade de mineração é vista como um serviço Web. Assim, a partir dos requisitos do usuário, os serviço Web disponíveis são ligados usando o BPEL4WS [1] para construir o processo de mineração desejado. Por fim, o resultado da mineração descrito PMML [2] é retornado e o serviço de análise dos resultados invocado, retornando ao usuário os resultados obtidos através do processo de mineração.

A arquitetura do *framework* proposto neste trabalho inclui uma interface de usuário única, um construtor de processos de mineração e um executor de processos de mineração como é apresentado na Figura 3.1.

Single Integrated User Interface - o usuário acessa o sistema através de uma interface disponibilizada através de um *browser*. Esta interface oferece as funções de construção e execução de um processo de mineração. O usuário pode recuperar e modificar um modelo de processo de mineração construído anteriormente. Se o usuário não encontra um modelo semelhante, ele pode criar seu próprio modelo de processo de mineração através do *Data Mining Process Designer*.

***Data Mining Process Designer* -** Durante o processo de construção do modelo, o sistema oferece um guia para ajudar o usuário no processo de busca e seleção do serviço web apropriado dentre os serviços disponíveis. A execução da sequência dos serviços web é decidida e integrada usando o BPEL4WS. Após, a construção do modelo, a sequência construída pode ser executada depois.

***Data Mining Process executor* -** Executa o processo de mineração baseado nas descrições armazenadas nos arquivos BPEL4WS. O executor invoca os serviços sequen-

[1] Do inglês, *Business Process Execution Language for Web Service*
[2] Do inglês, *Predictive Model Markup Language*

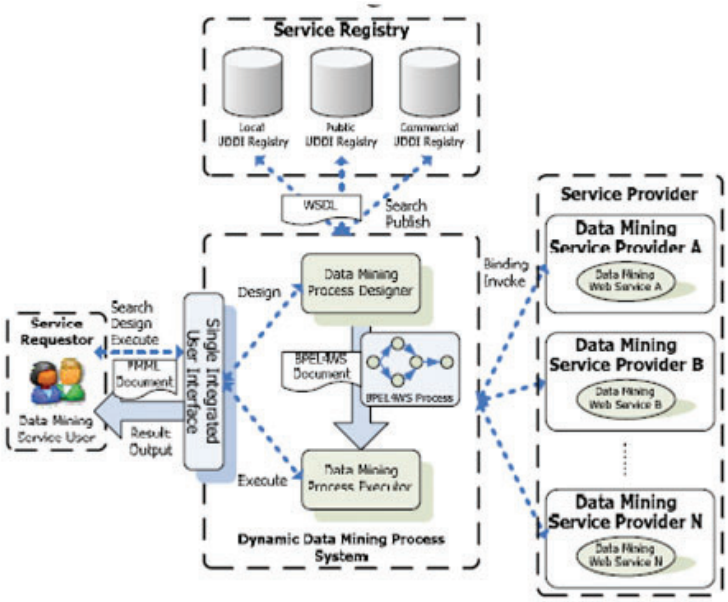

Figura 3.1: Arquitetura do *framework*

cialmente e combina o resultado em um documento PMML. Então, o resultado final é retornado a interface.

Apesar de oferecer uma interface ao usuário e basear sua arquitetura em serviços Web, o que facilita a integração com os ambientes *e-Learning*, este framework possui algumas limitações na sua aplicação dentro do contexto da mineração de dados educacionais, são elas: (i) este *framework* não possui foco educacional - Desta forma, para que possa ser utilizado no contexto educacional, é necessário um esforço maior de seus usuários, pois eles precisarão estender o *framework* levando em consideração os detalhes e particularidades inerentes a mineração de dados educacionais; (ii) Utiliza o BPEL4WS para descrição dos serviços - Para estender este *framework*, o usuário precisará, além de criar e adicionar o novo serviço, definir um novo fluxo de execução, incluindo o novo serviço que se deseja adicionar. Isso fará com que o usuário necessite conhecer todos os serviços disponíveis e ter conhecimento aprofundado em mineração para poder combiná-los na construção desse fluxo.

Diferentemente da proposta de Tsai e Tsai, a solução proposta neste trabalho tem o foco educacional e, por isso, tem seus serviços de pós-processamento pensados e direcionados a atender as necessidades e particularidades da mineração de dados educacional. Além disso, o esforço para a inclusão de um novo serviço Web é reduzido, pois com as descrições semânticas não é necessário predefinir um fluxo específico de execução desses serviços. Os serviços são combinados automaticamente baseado nessas descrições semânticas por meio dos parâmetros informados pelo usuário.

3.2 *A Semantic Web Service Oriented Framework for Adaptive Learning Environments*

Dietze, Gugliotta e Gugliotta [12] propõem um *framework* baseado em serviços Web semânticos com a finalidade de oferecer conteúdo personalizado aos estudantes baseado em objetivos de aprendizagem definidos. Para isto, objetos de aprendizagem foram encapsulados em serviços Web e descritos semanticamente utilizando *Web Service Modelling Ontology* - WSMO, possibilitando que o processo de composição e invocação fosse realizado de forma automática e dinâmica utilizando o IRS-III, *Internet Reasoning Service*. Assim, baseado nos objetivos que se deseja alcançar, a funcionalidade mais adequada é selecionada e invocada, permitindo uma adaptação altamente dinâmica a diferentes necessidades dos alunos.

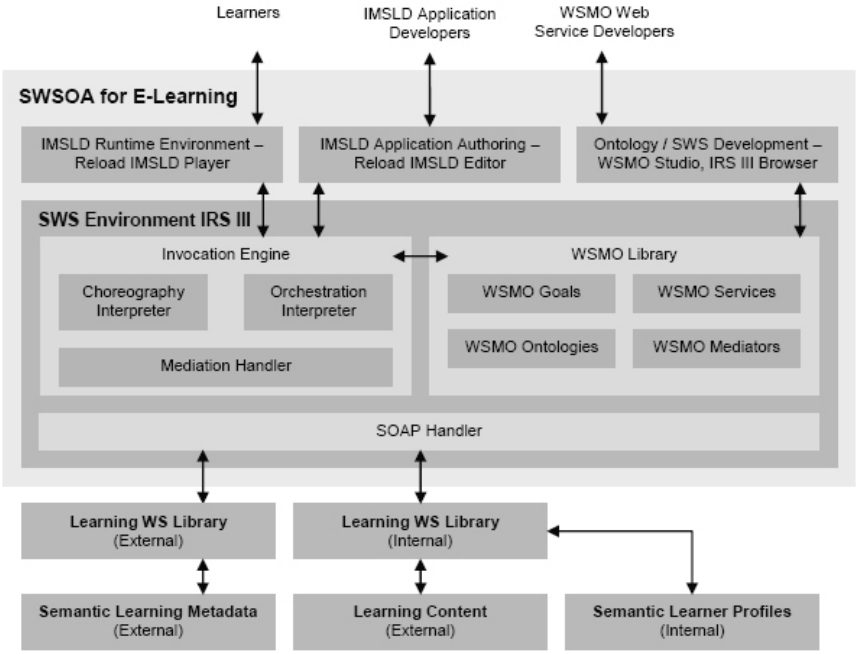

Figura 3.2: Arquitetura do *framework* proposto

A arquitetura representada na Figura 3.2 é, fundamentalmente, baseada nas camadas semânticas descritas. Na figura, o agente SWS é baseado no *WSMO framework* serve como fundação para a seleção, composição e invocação dinâmica dos serviços web. Os serviços são distribuídos através de diferentes repositórios externos e provêem funcionalidades baseados em dados existentes de aprendizagem e repositórios de metadados. Além disso, muitas interfaces de usuários para desenvolvimento e apresentação de aplicações de aprendizagem, bem como para o desenvolvimento das descrições formais semânticas dos serviços web são utilizadas. A implementação atual faz uso de ambientes em tempo de execução e implementa uma arquitetura orientada a serviços Web semânticos baseada na infraestrutura destes componentes. Para o processamento de WSMO em tempo de

execução, assim como o ambiente de desenvolvimento para WSMO foi usado o IRS III, enquanto o processador e editor de IMS LD é suportado pelo *Reload Learning Design Editor and Player*.

3.3 *TADA-Ed: Tool for Advanced Data Analysis in Education*

O *TADA-Ed* [39], *Tool for Advanced Data Analysis in Education* é uma plataforma de mineração de dados direcionada a professores, permitindo-lhes visualizar e minerar os exercícios *on-line* dos estudantes com o intuito de descobrir padrões pedagógicos relevantes. Esta ferramenta inclui alguns algoritmos de classificação, agrupamento e regras de associação adaptados da biblioteca disponibilizada pelo Weka e contém também facilidades de filtragem e pré-processamento de dados.

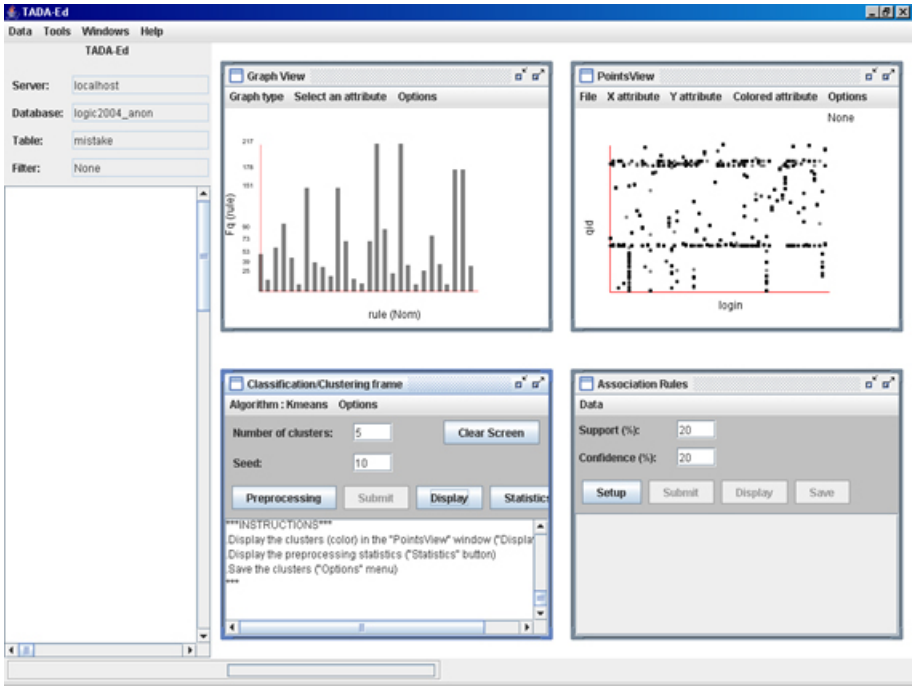

Figura 3.3: Tela do sistema proposto

TADA-Ed possui quatro telas principais:

- No canto superior esquerdo mostra um histograma e linhas;
- No canto superior direito mostra um gráfico em duas dimensões;
- No quadro inferior esquerdo possui os algoritmos de classificação e agrupamento, como *K-means*;
- No canto inferior direito contém os algoritmos de regras de associação, como *Apriori*.

As janelas são interligadas e ao selecionar um ponto na tela de visualização são exibidos os seus principais atributos. Além disso, ao selecionar uma das barras da tela de visualização de gráficos os pontos relacionados são destacados na tela de visualização de pontos.

O TADA-Ed possui ainda facilidades relacionadas a administração, como: os filtros (Seleção de um subconjunto de dados) e pré-processamento do banco de dados para o módulo de classificação / agrupamento e de regras de associação módulo. Porém, nessa abordagem o próprio educador deve analisar sua base de dados no TADA-Ed para poder obter informações relevantes que auxiliem o processo pedagógico e possam ser aplicados ao ambiente. Desta forma, o custo necessário para realizar a mineração será elevado, dada a necessidade de intervenção direta do professor tanto no processo da mineração dos dados quanto na análise dos resultados obtidos com ela.

A vantagem do *framework* proposto neste trabalho é a fácil integração entre o ambiente *e-Learning* e a solução proposta. Assim, os resultados podem ser aplicados diretamente nos ambientes e como possui um foco educacional, os detalhes inerentes a mineração de dados educacionais são atendidos.

Capítulo 4

Framework Proposto

Neste capítulo é apresentado *framework* proposto, bem como as etapas seguidas para sua construção. Inicialmente, são apresentados os requisitos da solução, os principais atores [1] envolvidos e suas interações com a solução proposta. Posteriormente, é apresentada as decisões de projeto tomadas para a definição da arquitetura do *framework* proposto, que é apresentada em seguida.

4.1 Requisitos da solução

Esta seção descreve os requisitos da solução, tanto funcionais quanto não-funcionais, baseados nos desafios envolvidos no contexto da mineração de dados educacionais levantados anteriormente. A identificação dos requisitos se baseou na estratégia baseada em atores, isto é, para cada um dos atores é feito um questionamento sobre as funcionalidades que podem ser requisitadas ao sistema, baseado no seu perfil de uso. Sendo assim, antes da apresentação dos requisitos da solução (Seção 4.1.2), a Seção 4.1.1 apresenta os atores envolvidos nos diagramas de casos de uso.

4.1.1 Atores

Os principais usuários da solução proposta são professores e desenvolvedores de ambientes educacionais, que podem assumir dois tipos de papéis relacionados ao *framework*: desenvolvedor e usuário. Assim, é apresentada a seguir as descrições desses papéis, que são considerados os principais atores do sistema.

Desenvolvedor: inicialmente, é preciso verificar se o sistema possui todos os serviços necessários sobre mineração de dados educacionais para atingir o objetivo desejado. Caso os serviços disponíveis não sejam suficientes, o desenvolvedor irá realizar adaptações e/ou

[1] do Inglês *stakeholders*

extensões necessárias no *framework* para realizar a mineração de dados em seu ambiente *e-Learning*, de forma a atingir tais objetivos. Essa adaptação pode consistir, por exemplo, na inclusão de um novo algoritmo de pré-processamento, de mineração, de um serviço para recuperar dados de uma base específica ou até mesmo um serviço educacional não previsto anteriormente.

Usuário: em outro momento, quando o sistema contém todos os serviços necessários para atingir os objetivos desejados pelo usuário, é desejável instanciar o *framework* por meio das escolhas relacionadas a cada uma das etapas da mineração de dados: pré-processamento, mineração e pós-processamento, de forma a realizar a mineração de dados em um ambiente *e-Learning* específico. Assim, basta o usuário definir os objetivos e realizar o processo de invocação do *framework* proposto passando os parâmetros de entrada e saída necessários.

4.1.2 Requisitos Funcionais

Após a definição dos atores, foram identificados os requisitos desejados para o sistema:

RF1. O sistema deve oferecer serviços de descoberta de conhecimento: pré-processamento, mineração de dados O sistema deve fornecer os mecanismos para viabilizar as etapas da mineração tais como: pré-processamento, transformação e mineração para que os ambientes educacionais possam realizar tal processo.

RF2. O sistema deve oferecer serviços de pós-processamento O sistema deve possuir funcionalidades de pós-processamento, serviços educacionais, que traduzam os resultados obtidos pelo processo de descoberta de conhecimento aos ambientes educacionais.

RF3. O sistema proposto deve oferecer um mecanismo que possibilite a composição e a execução dos serviços disponibilizados pelo sistema O sistema deve oferecer um mecanismo que permita ao usuário selecionar os serviços e executá-los.

RF4. O sistema deve possibilitar ao desenvolvedor a adição de novos serviços. O sistema deve possibilitar a adição de novas serviços a fim de permitir a adição de funcionalidades não previstas. Esses serviços poderão ser serviços de pré-processamento, mineração e/ou pós-processamento.

Os requisitos funcionais do sistema foram representados graficamente através de um diagrama de Casos de Uso descrito em UML (Figura 4.1), que representa de forma explícita o relacionamento entre atores e as funcionalidades do sistema. A seguir, é apresentada a

descrição de cada um dos casos de uso apresentados.

Figura 4.1: Principais Casos de Uso do *Framework* Proposto

Para realizar a instanciação de um produto do *framework*, um conjunto de serviços compostos, é necessário a invocação do gerenciador de serviços. Na figura 4.2 são apresentados os casos de uso do gerenciador de serviços, que foi construído a partir do *middleware grinv*. O *middleware grinv* [2], proposto por Heitor Barros, é uma solução adaptável que provê o processo de descoberta, composição e invocação automática de serviços Web semânticos. Desta forma, os casos de uso apresentados são funcionalidades que foram reutilizadas do *middleware*.

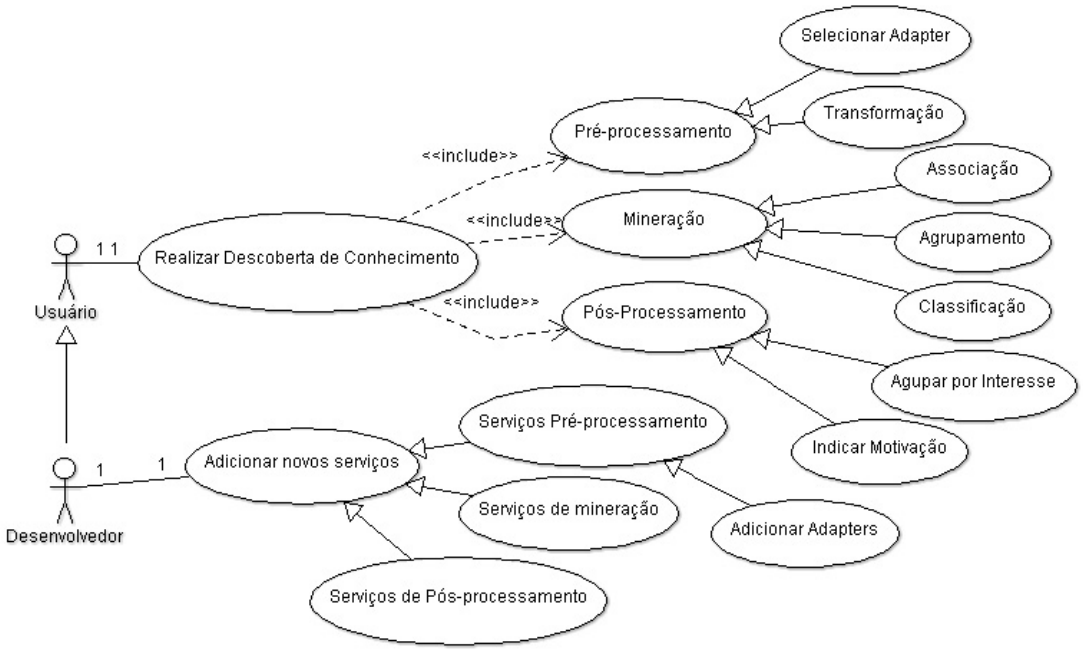

Figura 4.2: Principais Casos de Uso do *middleware grinv* reutilizados

Nome: Realizar descoberta de conhecimento
Usuário: Usuário do *framework*
Descrição: O usuário poderá realizar a mineração através de um mecanismo que realiza a composição e execução dos serviços disponíveis. Os serviços que podem ser utilizados

na composição são: serviços de pré-processamento, mineração e pós-processamento.

Nome: Adicionar novos serviços
Usuário: Desenvolvedor
Descrição: O desenvolvedor poderá adicionar novos serviços. Esses serviços podem ser serviços de pré-processamento, mineração e de pós-processamento.

4.1.3 Variabilidades desejadas

Para explicitar a diversidade de produtos que podem ser gerados com o *framework*, as variabilidades especificadas são apresentadas na Figura 4.3, através de um modelo de características[2] que contêm as possíveis decisões que envolvem a configuração de produtos.

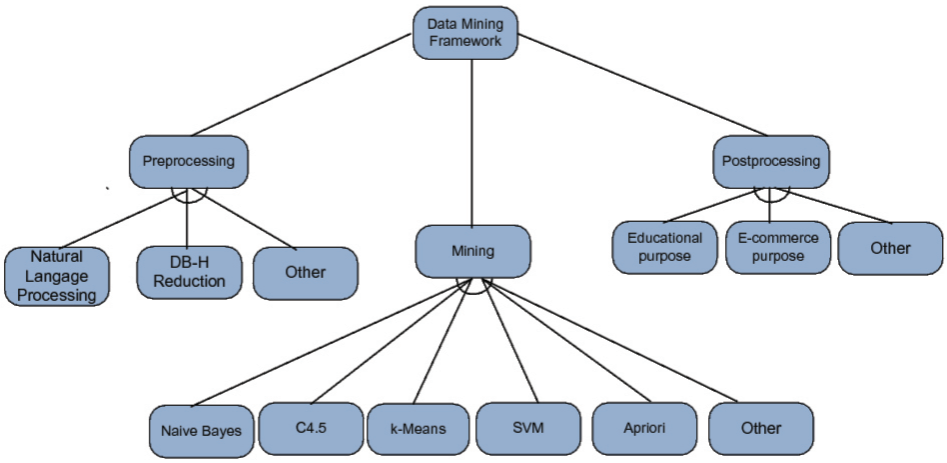

Figura 4.3: modelo de características do *framework*

É apresentada na Figura 4.3 o modelo de características do *framework*. O modelo é composto por 3 (três) características opcionais. Assim, no processo de instanciação do *framework*, uma ou mais características podem ser escolhidas, podendo ser: *preprocessing*, *mining* e/ou *postprocessing*. As características opcionais citadas são compostas por características alternativas, desta forma, após a definição das características opcionais que vão integrar a instância, uma entre as características alternativas de cada característica opcional deve ser escolhida.

Por exemplo, na instanciação do *framework* foram escolhidas as características *mining* e *postprocessing*, após esta etapa as características alternativas foram escolhidas: *Apriori* da característica *mining* e *educational propose* da característica *postprocessing*.

[2]Do Inglês *feature model*.

4.1.4 Requisitos Não-funcionais

RNF1. [Portabilidade] Independência de plataforma. O *framework* deve ser portável para qualquer plataforma.

RNF2. [Interoperabilidade] O *framework* proposto deve poder ser utilizado por qualquer ambiente *e-learning*. O *framework* deve poder ser integrado a qualquer ambiente educacional disponível, independentemente da linguagem com a qual o ambiente educacional foi desenvolvido. (O *framework* deve ter mecanismo de integração que permita que quaisquer ambientes educacionais possam ser facilmente adaptável para o uso da mineração de dados educacional)

RNF3. [Adaptabilidade] Permitir a seleção de algoritmos/ferramentas de mineração de dados, de acordo com a necessidade dos ambientes educacionais. O *framework* deve possibilitar aos seus usuários a personalização do funcionamento, através da seleção de algoritmos apropriados às suas necessidades de mineração.

RNF4. [Dinamismo] O *framework* deve permitir a Localização/composição dinâmica dos serviços disponíveis. O *framework* proposto deve possibilitar a descoberta e composição dos serviços disponíveis de forma dinâmica e automática.

RNF5. [Manutenibilidade] O *framework* deve facilitar a identificação e correção de falhas, além da evolução dos seus requisitos. O *framework* proposto deve informar erros com mensagens informativas e possibilitar a identificação e correção das falhas. Além do mais, o *framework* deve permitir, de forma facilitada, que novas funcionalidades e/ou novas ferramentas sejam adicionadas a ele.

RNF5. [Reuso] O *framework* deve promover o reuso dos serviços disponíveis. O *framework* proposto deve promover o reuso dos serviços de pré-processamento, mineração e pós-processamento disponíveis no sistema afim de possibilitar a redução de custos de desenvolvimento.

4.2 Projeto arquitetural

A arquitetura de software possui um papel decisivo para a satisfação dos requisitos de qualidade de um sistema de software [53], entre eles a flexibilidade e adaptabilidade, que são essenciais no contexto de um *framework*. Por essa razão, o projeto da arquitetura foi realizado de forma sistemática, envolvendo importantes decisões de projeto. Para a

condução do projeto arquitetural do *framework* foi realizada inicialmente a decomposição de seus módulos funcionais. Essa decomposição teve o intuito de aumentar a coesão dos componentes arquiteturais e consequentemente facilitar a compreensão e manutenção do sistema. Em seguida, a arquitetura foi refinada de modo a atender os requisitos não-funcionais apresentados na Seção 4.1.4. No tocante às decisões de projeto adotadas no desenvolvimento da solução proposta, podemos destacar três delas: (i) a opção por um framework; (ii) a opção pelo uso de serviços; e (iii) a opção por serviços Web semânticos baseados em ontologias. A seguir, essas decisões são melhores justificadas.

A Opção por um *Framework*. Ao avaliar os requisitos da solução, optou-se pela construção de um *framework*, visto que a solução proposta possui características inerentes a essa categoria de sistemas, tais como reusabilidade e adaptabilidade, que permite sua extensão e adaptação as necessidades de cada usuário dentro do domínio de aplicação definido.

Desta forma, a solução proposta promove o reuso de algoritmos/ferramentas de pré-processamento, mineração de dados e pós-processamento. E com isso, permite o reuso de soluções desenvolvidas para a mineração de dados educacionais, e que agora não serão úteis apenas a seus desenvolvedores, como destacado em [49].

Outra característica importante da solução proposta é a adaptabilidade, uma vez que pode ser adaptada às necessidades de cada usuário. Desta forma, adaptabilidade permite que o usuário adicione/modifique funcionalidades da solução proposta para alcançar os objetivos desejados com a mineração de dados educacionais. Por exemplo, adicionar uma nova funcionalidade educacional não prevista anteriormente.

Além das características citadas anteriormente, a solução herda outras vantagens que são inerentes aos *frameworks*, como menos código para projetar e escrever, redução do custo de desenvolvimento, entre outros.

Opção pelo uso de Serviços. *Frameworks* orientados a objetos tradicionais proveem reuso e adaptação, mas são restritos as linguagens de programação em que foram desenvolvidos. Assim, a decisão de construir um *framework* orientado a objetos tradicional não é suficiente para atender os requisitos de interoperabilidade e, em alguns casos, portabilidade.

Para promover os requisitos de interoperabilidade, portabilidade e permitir uma fácil de integração, necessários no contexto do *framework* proposto, foi adotada uma abordagem orientada a serviços [14]. Os serviços foram escolhidos, pois a interoperabilidade e a portabilidade são características inerentes a eles. Um serviço Web pode ser implementado e utilizado por quaisquer aplicação mesmo que tenham sidos implementados em diferentes linguagens de programação. Desta forma, os serviços foram usados para encapsular os algoritmos/ferramentas de pré-processamento, mineração de dados e pós-processamento. A facilidade de integração foi alcançada pelo fato da invocação de serviços Web ser uma tarefa simples.

Opção pelo uso de Serviços Web Semânticos. Apesar das grandes vantagens conseguidas com a utilização dos serviços Web, a proposta de um *framework* baseado em serviços não atende aos requisitos especificados. Pois, apesar do processo de mineração de dados ter um fluxo definido - seleção, pré-processamento, transformação, mineração e pós-processamento - a definição dos serviços quais os serviços de cada etapa deve ocorrer de forma estática, isto é, durante a definição do processo de mineração é necessário que o usuário indique um serviço Web deseja utilizar em cada etapa. Como no contexto do *framework* há a necessidade de se adicionar novos serviços dinamicamente, o uso da tecnologia tradicional não permite que os serviços sejam descobertos, compostos e invocados de forma dinâmica e automática. Em abordagens como o BPEL do inglês, Business Process Execution Language, é necessário predefinir quais serviços serão utilizados em cada etapa, restringindo, assim, os serviços a fluxos predefinidos.

Sendo assim, é necessário complementar a definição dos serviços com informações semânticas adicionais, possibilitando que a sua descoberta, composição e execução sejam realizadas de forma dinâmica e automatizada. Dessa forma, foi proposto o uso de serviços Web baseados em ontologias, serviços Web semânticos. Logo, o fluxo de execução dos serviços não precisa ser definido a priori para que seja executada a mineração. Além disso, o *framework* pode compor os serviços disponíveis e prover outras funcionalidades, que não haviam sido previstas inicialmente.

4.3 Visão Lógica da Arquitetura

Baseada nas ponderações supracitadas, foi proposto um *framework* baseado em serviços Web semânticos para prover a mineração de dados educacionais em ambientes *e-Learning*. O *framework* segue uma abordagem orientada a serviços para facilitar a extensão e integração com diferentes aplicações na Web.

Desta forma, tal ferramenta provê a) técnicas de mineração: possui técnicas de mineração implementadas, com isso os ambientes não necessitarão mais implementá-las para realizar mineração de dados (e.g. Regras de associação); b) serviços educacionais: disponibilizará serviços educacionais aos ambientes, provendo a esses ambientes funcionalidades da mineração de dados com enfoque educacional (e.g. Serviço de recomendação de materiais); c) automatização dos processos: irá prover a composição, descoberta e invocação de serviços de forma dinâmica e automática.

A Figura 4.4 apresenta a visão lógica da arquitetura do *framework* proposto, incluindo seus principais componentes arquiteturais. Como pode ser visto, foi adotada uma arquitetura em camadas [53], que preserva uma separação explícita entre o controle da aplicação (Web Services) e os dados armazenados (Database e Log). Vale ressaltar que a camada

de controle interna baseia-se em uma arquitetura orientada a serviços (SOA[3]), que define explicitamente a existência de um localizador de serviços (Service Manager). Porém, diferentemente de uma arquitetura SOA tradicional, o elemento Service Manager possui a habilidade de localizar serviços, dinamicamente, a partir das suas descrições semânticas descritas em uma ontologia. Além disso, os serviços identificados podem ser, dinamicamente, compostos e executados. Um outro destaque que deve ser ressaltado se refere à definição explícita de três categorias de serviços de acordo com as etapas da mineração de dados: Educational Services, que encapsula algoritmos de pós-processamento no contexto de sistemas educacionais; Mining Services, que encapsula algoritmos de mineração de dados; e Preprocessing Services, que encapsula algoritmos de pré-processamento. A seguir, cada um dos elementos da arquitetura é apresentado em maiores detalhes.

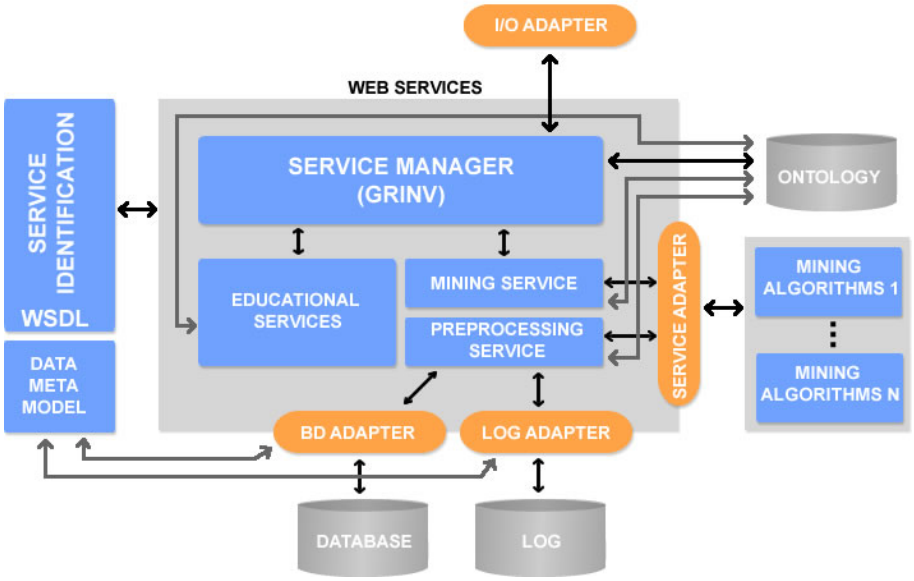

Figura 4.4: Visão lógica

4.3.1 *Mining Algorithms*

Esta camada é composta por vários algoritmos de mineração que serão utilizados para atingir os objetivos da mineração educacional. Estes algoritmos são encapsulados em serviços Web de forma que suas funcionalidades possam ser utilizadas por outras camadas.

Inicialmente, foi utilizada a ferramenta WEKA [62] para servir como prova de conceito. No entanto, é importante notar que a utilização de serviços Web proporciona a utilização de várias ferramentas ao mesmo tempo e que a arquitetura se torna, facilmente, expansível para a adição de novas ferramentas (e.g. DBMiner, Clementine). O WEKA foi escolhido por ser amplamente utilizado na comunidade científica, possuir uma documentação bastante abrangente e de fácil utilização. Além disso, o WEKA provê uma API

[3]Sigla do Inglês *Service-Oriented Architecture*

que permite utilizar os seus algoritmos e ferramentas diretamente em código Java, o que possibilita a criação de aplicações que o utilizem.

4.3.2 *Adapter*

Abaixo são descritos os adaptadores presentes na arquitetura da solução proposta.

I/O Adapter

O *I/O adapter* é o adaptador que define os padrões para comunicação com os ambientes *e-Learning*. O formato de entrada definido para a comunicação com o *framework* é o par (chave, valor), que representam a URI da ontologia e seu valor. O padrão de saída é um XML definido pelo serviços de pós-processador, sem um *Schema* definido. O *framework* possibilita ainda o recebimento de dados que podem ser utilizados na mineração através do *I/O adapter*. Para isto, os dados deve seguir um padrão, que é um XML com *Schema* definido para o formato de tabela.

Service Adapter

É a abstração do encapsulamento de algoritmos mineração disponibilizados pelas ferramentas de mineração. Assim, cada serviço encapsula um algoritmo específico de mineração, o que permitir que outras camadas possam descobrir, compor e invocar os serviços de forma dinâmica, combinando algoritmos de acordo com as necessidades dos usuários.

DB Adapter

O *Data Base Adapter* é um adaptador que possibilita desenvolvedores de ambientes *e-Learning* utilizarem o *framework* proposto independentemente de quais são ou como suas fontes de dados foram modeladas. Para isso, o usuário do *framework* deve construir um adaptador implementando a interface definida pelo **Data Meta Model**, que define o formato dos dados entendido pelo *framework*.

Para utilizar o adaptador construído, o usuário do *framework* deve transformá-lo em um serviço Web semântico. Depois de construído o serviço, o usuário passa ao *framework* a URI do serviço construído como parâmetro de entrada. Assim, o *framework* poderá recuperar os dados dos ambientes.

Log Adapter

O *Log Adapter* tem uma função semelhante ao **DB Adapter**, que é permitir a desenvolvedores de ambientes educacionais utilizarem o *framework* proposto para minerar os logs gerados por seus ambientes em busca de informações educacionais relevantes. O processo de construção e utilização do **Log Adapter** é o mesmo do **DB Adapter**. Ou

seja, implementar uma interface definida, transformá-la em serviço e passar a URI do serviço construído como parâmetro de entrada ao *framework*, que utilizará o serviço para recuperar os dados armazenados nos logs.

Estes adaptadores foram apresentados de forma separada na arquitetura da solução mesmo com funções e utilização tão semelhantes, pois buscou-se destacar a importância dessas duas fontes de dados dentro da mineração de dados.

4.3.3 *Web Services*

Esta camada é composta pelos serviços Web que encapsulam as funcionalidades disponibilizados pelo *framework* proposto, como o gerenciador de serviços, algoritmos de pré-processamento, algoritmos de mineração e pós-processamento. Estas funcionalidades são sub-divididas em quatro camadas, respectivamente: i) *Service Manager*; ii) *Mining Services*; iii) *preprocessing services*; iv) *Educational Services*. Porém, apenas o *Service Manager* é um serviço Web tradicional. As outras camadas são compostas por serviços Web semânticos, que são serviços Web tradicionais com uma anotação semântica. Isto permite que o gerenciador de serviços possa realizar o processo de descoberta, composição e invocação automática e dinâmica desses serviços.

4.3.4 *Preprocessing Services*

Esta camada fornece serviços Web que encapsulam algoritmos de pré-processamento disponibilizados pela camada *mining algorithms*. Estes serviços fornecem funcionalidades para preparação e tratamento dos dados, essenciais para a mineração no processo de descoberta de conhecimento.

Alguns dos serviços Web disponíveis nesta camada são:

ListToARFF

Descrição: Este serviço realiza a transformação de uma lista com o histórico de acesso de todos os usuários em um formato que possa ser processado pelos algoritmos de mineração, um arquivo ARFF[4]. Desta forma, o serviço retornará como saída uma URI que representa o arquivo ARFF gerado.

Parâmetros de entrada: Lista com o histórico de todos os usuários

Parâmetros de Saída: URI do ARFF

StringToRules Service

Descrição: Serviço de transformação responsável por traduzir as saídas geradas pelos serviços de regras de associações. As saídas geradas por estes serviços é uma única *string*

[4]Arquivo aceito pela ferramenta de mineração WEKA

com todas as informações, incluindo as regras de associações. Por isso, este serviço é necessário para extrair essas as regras dessas saídas confusas e transformá-las em uma lista de regras bem definidas permitindo que outros serviços utilizem estas regras para, por exemplo, realizar recomendação de conteúdo.
Parâmetros de entrada: *String* (saída dos serviços de associação contendo todas as informações inclusive as regras)
Parâmetros de Saída: Regras de associação

Mining Services

Esta camada fornece uma abstração dos requisitos implementados nas ferramentas de mineração em serviços Web. Através desta camada, os ambientes educacionais não necessitam implementar os algoritmos de mineração, bastando apenas utilizar os serviços disponíveis, diminuindo a complexidade e o tempo de desenvolvimento desse tipo de sistema. Além disso, diminui-se o acoplamento entre esses ambientes/ferramentas das aplicações, pois fazem uso dos algoritmos de mineração, independentemente da ferramenta que os implementam.

Alguns dos serviços Web disponíveis nesta camada são:

Apriori Service

Descrição: Este serviço prover o algoritmo de mineração Apriori. Esta técnica faz parte da tarefa de regras de associação, cujo objetivo é obter da base de dados em questão a associação entre os dados, a fim de descobrir padrões. Assim, o algoritmo apriori procura na base de dados essas associações gerando regras de associação, onde baseado em duas medidas, confiança e suporte, seleciona as melhores regras de associação descobertas.
Parâmetros de entrada: Confiança, suporte e a base de treinamento (URI do ARFF)
Parâmetros de Saída: *String* com todas as informações incluindo as regras

Simple k-Means Service

Descrição: Este serviço disponibiliza o algoritmo de mineração *Simple k-means*. Este algoritmo consiste em agrupar os dados fornecidos como entrada gerando um modelo. Desta forma, o modelo gerado é armazenado em um arquivo e o serviço retorna como saída a URI desse arquivo.
Parâmetros de entrada: Base de treinamento (URI do ARFF)
Parâmetros de Saída: URI do Modelo

Naive Bayes Service

Descrição: Este serviço disponibiliza o classificador *Naive Bayes*, pertencente a tarefa de

mineração de Classificação. Esta técnica realiza classificação baseada em probabilidades obtidas a priori, recebe o nome de Bayes - por utilizar a regra de Bayes para classificar; e *Naive* - ingênuo - por considerar independência entre as características usadas para classificar. Assim como, *Simple K-Means* este serviço gera um modelo de classificação e armazena em um arquivo passando como parâmetro de saída a URI deste arquivo.
Parâmetros de entrada: Atributo (qual dos atributos será usado para classificar), Base de treinamento (URI do ARFF)
Parâmetros de Saída: URI do Modelo

Bayes Net Service
Descrição: Este serviço disponibiliza um classificador Bayes Net. Esta técnica realiza classificação baseado em probabilidades.
Parâmetros de entrada: Atributo (qual dos atributos será usado para classificar), Base de treinamento (URI do ARFF)
Parâmetros de Saída: URI do Modelo

4.3.5 *Educational Services*

Esta camada é composta por serviços Web semânticos educacionais, que proveem as funcionalidades educacionais utilizadas pelo *framework* proposto no pós-processamento para atingir os objetivos da mineração de dados educacional, como a recomendação de conteúdo. Para tal, os serviços educacionais processam os resultados obtidos dos serviços de mineração que compõe a camada *mining services*.

Classificação do usuário (baseado na área de interesse usando *K-Means*)
Descrição: Este serviço visa classificar usuários de acordo com suas áreas de interesse. Este serviço realiza esta classificação baseado no modelo gerado pelo algoritmo K-Means, que agrupou os usuários baseado nos links acessados em áreas de interesse. Com isso, usando o histórico do usuário que se quer classificar, este serviço irá realizar a classificação e retornar a qual classe esse usuário pertence.
Parâmetros de entrada: URI do Modelo gerado pelo serviço *K-Means*, lista com histórico do usuário
Parâmetros de Saída: a classe a qual o usuário pertence

Recomendação de links Web baseado em regras
Descrição: Este serviço utiliza regras de associação para realizar recomendação de conteúdo, links, entre outros.
Parâmetros de entrada: Regras de associação, Lista contendo histórico de acesso da

sessão atual do usuário que se quer recomendar
Parâmetros de Saída: Recomendação

4.3.6 *Service Manager*

Esta camada é composta por um único serviço Web, o *Service Manager* ou Gerenciador de serviços. Este serviço tem o papel de realizar a ponte entre a aplicação cliente e os serviços disponibilizadas pelo *framework*. Ou seja, prover o acesso a quaisquer funcionalidades disponibilizadas pelo *framework*. Para isto, basta que o usuário faça a invocação do gerenciador de serviços, que é um serviço Web tradicional, passando os parâmetros de entrada e e saída desejados. Baseado nesses parâmetros, o gerenciador de serviços realiza o processo de descoberta, composição e invocação automática e dinâmica dos serviços Web semânticos disponibilizados pelas camadas inferiores, executando todo o processo de mineração de dados educacional através de uma única interface.

Para a construção do Gerenciador de Serviços foi utilizado o *middleware grinv*. Este *middleware* disponibiliza algoritmos para realização do processo de descoberta e composição de serviços, além de permitir realizar o processo de invocação. No entanto, os algoritmos disponíveis não satisfaziam as necessidades apresentadas na definição deste *framework*. Desta forma, foi necessário realizar a extensão do *middleware* para adicionar um novo algoritmo - que realizasse busca em uma estratégia de encadeamento para trás, para que o processo de descoberta e composição pudesse atender as especificações definidas. O algoritmo compara os parâmetros passados pelo usuário com os parâmetros de entrada e saída dos serviços Web semânticos disponíveis para realizar o processo de descoberta e composição.

Os parâmetros utilizados pelo gerenciador de serviços são: i) parâmetros de entrada - que são os dados que serão minerados e os parâmetros de configuração dos algoritmos de mineração, como as medidas de confiança e o suporte as regras do algoritmo *Apriori*, por exemplo; ii) Parâmetros de saída - são as saídas pretendidas pelo usuário, ou seja, objetivos que o usuário deseja alcançar com a mineração de dados educacional, como a recomendação de estudante.

Esses parâmetros devem ser passados respeitando um formato predefinido. Os parâmetros são passados por pares, a primeira parte é a URI da ontologia e a segunda parte o seu valor. Isto ocorre porque todos os serviços disponíveis no *framework* são semânticos, ou seja, eles possuem uma ontologia que os descrevem. Assim, para que o *framework* saiba exatamente o que está sendo passado, é necessário apontar na ontologia e informar o valor. Por isso, o par [URI, valor].

Assim, o usuário pode utilizar quaisquer funcionalidades educacionais disponibilizadas pelo *framework* proposto apenas alterando os parâmetros de entrada e saída. Pois, o

gerenciador de serviços realiza o processo de descoberta, composição e invocação de forma automática e dinâmica.

4.3.7 Ontologia

De acordo com [44], serviços Web semânticos são geralmente definidos como uma melhoria aos serviços Web através de uma anotação semântica, permitindo uma maior automação do processo de descoberta, composição e invocação dos serviços Web. Além disso, o uso de ontologias fornece uma maior dinâmica a este processo.

Desta forma, foi proposta uma ontologia que descreve tarefas e técnicas de mineração de dados, especificando parâmetros de entrada dos algoritmos de mineração e as saídas desses algoritmos. Posteriormente, esta ontologia foi estendida para incluir parâmetros de entrada e saída de serviços disponibilizados pelo *framework* proposto, mas que não foram provenientes de ferramentas de mineração, por exemplo: serviços de transformação, serviços educacionais, entre outros.

A ontologia que descreve tarefas e técnicas de mineração de dados, além de especificar os parâmetros de entrada e saída de algoritmos de aprendizagem é apresentada na figura 4.5:

Abaixo, encontra-se a descrição de algumas classes da ontologia apresentada na Figura 4.5:

- *DataMining*: é uma classe genérica que representa a mineração de dados (sub-área de Inteligência Artificial) na ontologia.
- *DataMiningTasks*: representa as tarefas de mineração de dados, como: associação, agrupamento, classificação e outras técnicas que também podem ser modeladas.
- *DataMiningTechniques*: classe mãe de todas as técnicas separadas por tipo de tarefa.
  - *AssociationTechniques*: As técnicas que realizam associação devem ser subclasses dessa classe. Exemplo: Algoritmo *Apriori*.
  - *ClusterTechniques*: As técnicas que realizam clusterização devem ser subclasses desta classe. Exemplo: *K-means*.
  - *ClassifyTechniques*: As técnicas que realizam classificação devem ser subclasses desta classe. Exemplo: *C4.5*.
- *Parameters*: representa os parâmetros de entrada e saída dos algoritmos de aprendizagem que podem vir a ser representados na ontologia.

Para criar as ontologias foi utilizado o método 101 [43]. Para descrever as ontologias foi utilizada *OWL* e *OWL-S* para descrever os serviços Web. *OWL-S* foi escolhida pela sua

Figura 4.5: Ontologia utilizada pelo *framework*

compatibilidade com o *middleware* utilizado na construção do Gerenciador de Serviços e ser largamente utilizada pela comunidade cientifica.

4.3.8 Arquitetura de implementação

A arquitetura apresentada segue um estilo arquitetural heterogêneo que combina um estilo de arquitetura de *pipes-and-filters* e estilo de arquitetura em camadas. O estilo de arquitetura de *pipes-and-filters* foi usado para definir o processo de mineração, que é composto por três atividades básicas: pré-processamento, mineração de dados e pós-processamento. Estas atividades são implementadas como serviços Web e estruturadas como apresentado na Figura 4.6: i) O componente Mining Preprocessing contém serviços que preparam os dados antes de serem minerados. ii) O componente Mining Services contém serviços que encapsulam algoritmos e ferramentas de mineração em serviços Web. iii) O componente de Mining Postprocessing contém serviços que implementam algoritmos de pós-processamento de acordo com aplicações específicas.

A orquestração entre serviços de pré-processamento, mineração de dados e pós-processamento é realizada pelo componente Service Manager, que é um serviço Web responsável por realizar a descoberta, composição e invocação automática e dinâmica dos serviços Web semânticos disponíveis.

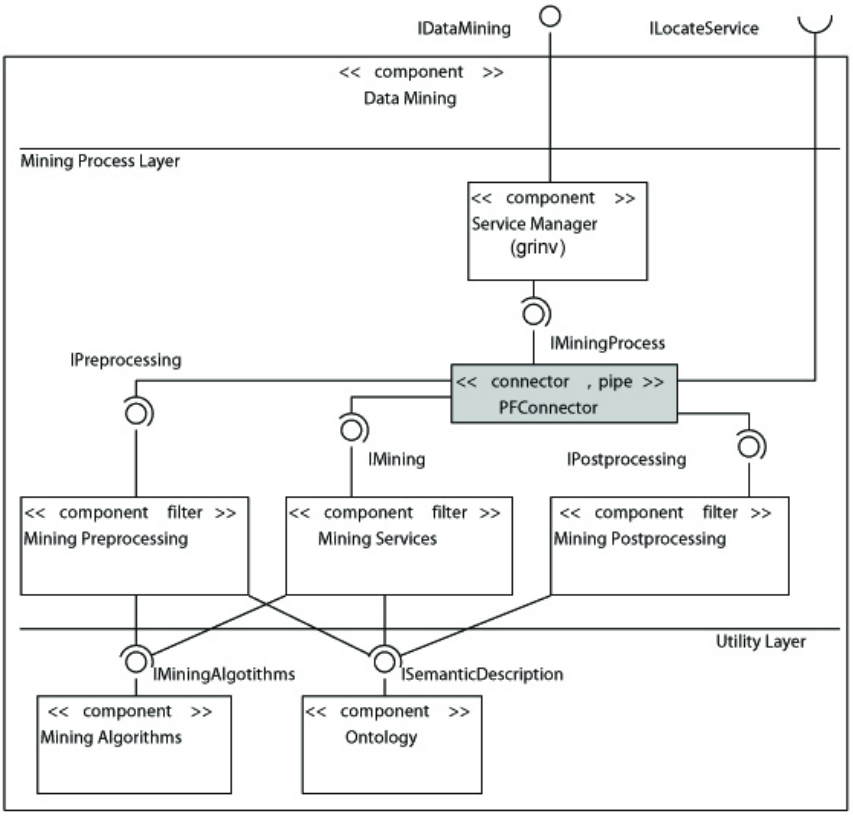

Figura 4.6: Arquitetura Detalhada Mineração de Dados

Além de suportar a execução do processo de mineração de dados, dois componentes de utilidade também foram definidos: i) Mining Algorithms componente que contem os algoritmos que podem ser usados pelo pré-processamento (usado pelo componente Mining Preprocessing) e mineração de dados (usado pelo componente Mining Services); ii) Componente Ontology que detalha as descrições semânticas dos serviços Web permitindo a identificação dinâmica e orquestração dos serviços.

É apresentado na Figura 4.7 o comportamento do componente Data Mining, depois de receber uma requisição da interface IDataMining. Depois de receber a requisição dos serviços e os respectivos dados para analisar, o componente Service Manager requisita a execução do processo de mineração de dados através da interface IMiningProcess do conector PFConnector. Este conector segue o protocolo de pipes-and-filters, ele executa os três componentes seguindo a sequência pela orquestração de seus serviços. Primeiro, um serviço de Mining Preprocessing é localizado (através da requisição a interface ILocateService) e executa (IPreprocessing). Os serviços de pré-processamento podem usar algoritmos providos pelo componente Mining Algorithms através da interface (IMiningAlgorithms). Depois do

pré-processamento, um serviço de mineração é localizado e executado passando os dados pré-processados como argumento para a interface (IMining), que podem também usar os algoritmos de mineração providos (IMiningAlgorithms interface). Finalmente, os resultados da mineração devem ser pós-processados de acordo com a proposta especifica. Para isto, o conector PFConnector localiza o serviço de pós-processamento e executa-o passando os resultados da mineração como argumento (IPostprocessing interface). Todos os serviços de mineração são especificados semanticamente atraves de ontologias (ISemanticDescription interface).

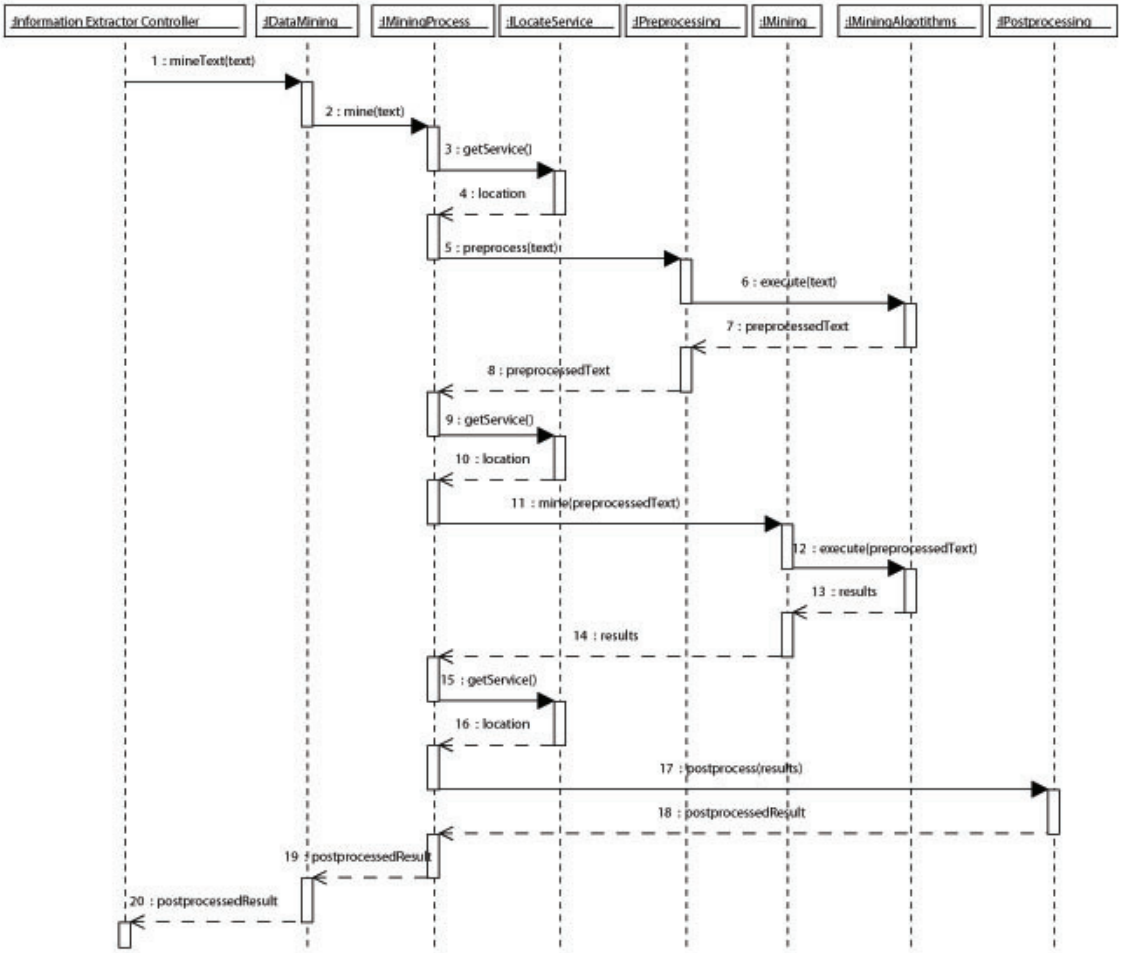

Figura 4.7: Diagrama de Sequencia Data mining

4.4 *Wizard*

A solução proposta possui um *wizard*, que tem a finalidade de auxiliar os usuários da solução no processo de instanciação e uso do *framework*. Este *wizard* é uma ambiente Web, que é disponibilizada junto com o *framework*, e oferece funcionalidades como: Adição de novos serviços de pré-processamento, mineração e/ou pós-processamento; e a realização do processo de instanciação, passando pelas fases de escolha do objetivo da mineração,

definição dos dados de entrada, definição dos parâmetros da mineração, avaliação do modelo criado até a apresentação dos parâmetros necessários para execução da instância criada. Esse processo será detalhando no estudo de caso do presente trabalho.

Capítulo 5

Estudos de Caso

Neste capitulo é apresentado o estudo de caso realizado com a finalidade de validar o *framework* proposto. O presente estudo de caso consiste em apresentar, passo-a-passo, o processo necessário para a instanciação e utilização da solução proposta em uma aplicação para realizar mineração de dados educacionais. Desta forma, é possível verificar, detalhadamente, as adaptações necessárias para que a solução proposta possa ser utilizada em ambientes *e-Learning*, bem como suas vantagens e desvantagens.

Para realização deste estudo de caso, foi desenvolvida uma aplicação que aborda um problema relevante no contexto da educação a distância: a evasão escolar. Para tal, foram utilizados dados colhidos de um ambiente *e-Learning* real.

5.1 Descrição do problema

A evasão escolar é um problema que ocorre nos diversos níveis e modalidades de ensino. Na modalidade à distância, este problema é, particularmente, mais complexo, já que a interação estudante-professor acontece por meio de um ambiente educacional e essa ausência de contato direto aliada ao grande número de estudantes matriculados por curso, dificulta ainda mais a percepção sobre estudantes que estão pouco motivados e, assim, tendem a desistir do curso mais facilmente.

A fim de avaliar o problema definido, foi realizada uma análise do histórico de utilização dos usuários gerado pelo ambiente *e-Learning*. Para restringir o universo amostral, foram considerados apenas estudantes matriculados em uma disciplina de um curso de graduação. A disciplina escolhida teve a duração de 7 (sete) semanas e teve a participação de aproximadamente 240 (duzentos e quarenta) estudantes. O objetivo foi avaliar a motivação dos estudantes que participam da referida disciplina.

Para isto, foram definidas três categorias de motivação: (1) não motivado, (2) pouco motivado; (3) motivado. Assume-se que qualquer que seja o aluno, ele deve se enquadrar em uma dessas três categorias. A identificação da categoria motivacional dos alunos

possibilitará, por exemplo, que os professores, tutores e coordenadores realizarem um planejamento para evitar que os alunos que não estejam motivados ou estejam pouco motivados não desistam do curso ou o conduzam sem compromisso.

Após a definição do problema, é descrito o processo de instanciação do *framework* proposto para a realização da mineração de dados educacional objetivando a identificação dos estudantes que tem maior probabilidade de evasão.

5.2 Processo de instanciação

As etapas necessárias para realizar o processo de instanciação são apresentadas na figura 5.1. São previstas três etapas para a instanciação de aplicações a partir da solução proposta: (1) Selecionar pós-processador, (2)Selecionar minerador e (3) Selecionar pré-processador. Ainda no contexto deste trabalho, o *wizard* desenvolvido para auxiliar usuários no processo de instanciação segue as etapas definidas.

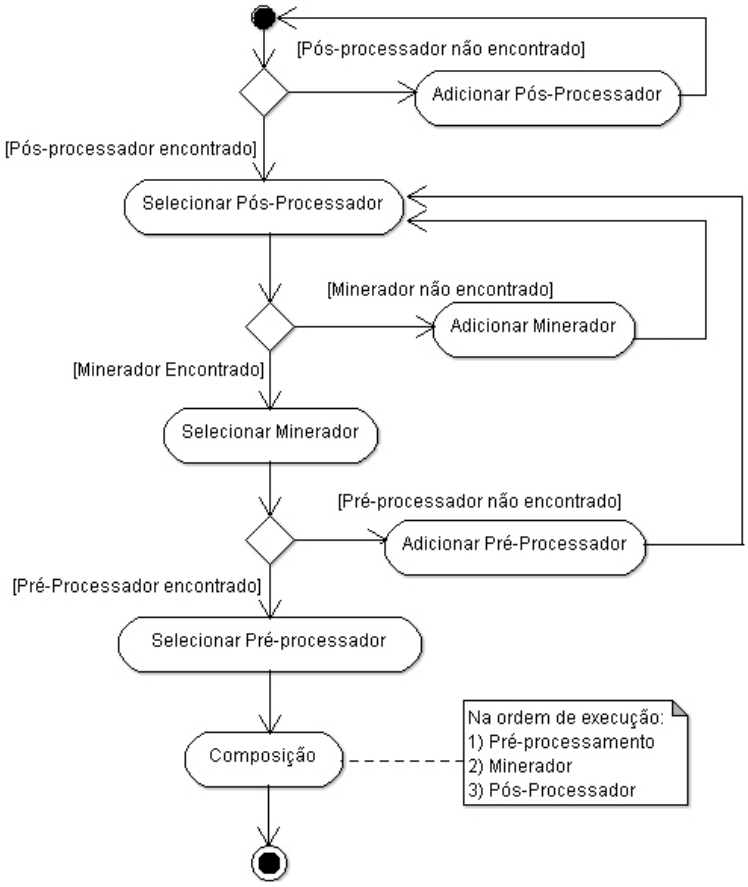

Figura 5.1: Diagrama de atividades

No diagrama de atividades apresentado acima, o primeiro passo é a escolha dos objetivos que se pretende alcançar com mineração de dados educacional, o *wizard* realiza consulta no repositório de serviços de pós-processamento e exibe esses serviços ao usuário.

Caso o serviço com objetivo desejado não esteja disponível, o usuário pode adicionar o serviço de pós-processamento desejado e reiniciar o processo. Após a escolha do objetivo, é a escolha do serviço de mineração. Como no passo anterior, se o serviço de mineração desejado não estiver disponível, o usuário pode adicioná-lo e reiniciar o processo. A última opção é apresentada ao usuário, a possibilidade de escolha entre os *adapters* cadastrados, que possuem a função de recuperar os dados que serão utilizados na mineração. Mais uma vez, se o usuário não encontrar o *adapter* com os dados que deseja utilizar, ele tem a possibilidade de adicionar o *adapter* desejado e reiniciar o processo. Por fim, Com a definição de quais dados serão utilizados e objetivo que se deseja alcançar, o *wizard* consulta o *framework* que retorna a composição dos serviços que satisfazem as opções selecionadas.

As seções seguintes apresentam a execução de cada uma das três etapas propostas para geração da aplicação proposta, ilustrando inclusive a utilização do *wizard* desenvolvido.

5.3 Pré-Processamento

Primeiramente, é necessário definir quais dados serão utilizados para extrair a informação desejada e atingir o objetivo definido, que é identificar se um determinado aluno está motivado ou não. Para isto, foram definidas meta-informações a partir dos dados de utilização dos usuários do ambiente *e-Learning*. As meta-informações foram definidas baseadas na experiência do autor acumulada no exercício das funções de professor-tutor à distância e professor pesquisador em um período de 2 (dois) anos e são detalhadas a seguir.

5.3.1 Meta-informações

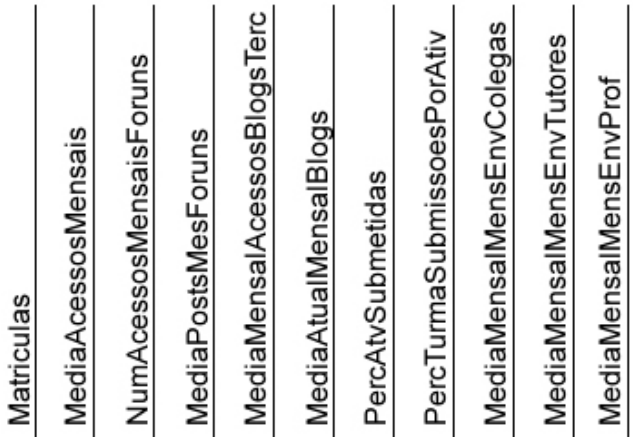

Figura 5.2: Meta-informação definidas para o processo de mineração

São apresentadas na figura 5.2 as meta-informações definidas. Em seguida, são apresentadas suas respectivas descrições em detalhes.

- **Matricula** - O número de matrícula do aluno para que possamos identificar os alunos que serão classificados;
- **MediaAcessosMensais** - Representa a média de acessos mensais do usuário ao ambiente *e-Learning*;
- **NumAcessosMensaisForuns** - Representa o número de acessos mensais ao fórum;
- **MediaPostsMesForuns** - Representa o número médio de *posts* por mês nos fóruns das disciplinas;
- **MediaMensalAcessosBlogsTerc** - Representa a média mensal de acessos aos Blogs de terceiros; colegas, tutores e/ou professores;
- **MediaAtualMensalBlogs** - Representa a média mensal de atualização dos Blogs do aluno em questão;
- **PercAtvSubmetidas** - Representa a porcentagem de atividades submetidas entre todas as atividades disponíveis;
- **PercTurmaSubmissoesPorAtiv** - Representa a porcentagem de atividades submetidas pela turma da qual ele pertence;
- **MediaMensalMensEnvColegas** - Representa a média mensal de mensagens enviadas aos colegas;
- **MediaMensalMensEnvTutores** - Representa a média mensal de mensagens enviadas aos tutores;
- **MediaMensalMensEnvProf** - Representa a média mensal de mensagens enviadas aos Professores.

Como parte do pré-processamento, foi necessário construir rotinas para recuperar os dados necessários para alimentar a tabela de meta-informações. Desta forma, foram construídas consultas em *SQL* e uma rotina em java que executou essas consultas, construiu a referida tabela e armazenou os meta-dados em um arquivo.

É apresentado na listagem 5.1 uma consulta especificada para obter uma meta-informação e sua descrição:

Code 5.1: Consulta para obter a meta-informação `PercAtvSubmetidas`

```
1 SELECT (FEITAS.TOTAL*100/ATVDISP.TOTAL) AS PERC
2 FROM
3
4  (SELECT COUNT(a.course) AS TOTAL
5 FROM assignment a
```

```
6  WHERE a.course IN
7
8          (SELECT c.id
9          FROM course c, course_display disp,
10           user u, role_assignments a
11         WHERE u.id = 13474 AND c.id = disp.course AND
12          u.id = disp.userid AND u.id = a.userid AND a.roleid = 5
13         GROUP BY c.id
14         ORDER BY c.id ) AND a.course != 351 ) ATVDISP,
15
16  (SELECT COUNT(*) AS TOTAL
17 FROM assignment_submissions a
18 WHERE userid = 13474 ) FEITAS
```

A consulta apresentada na listagem 5.1 recupera a média de submissões de atividades da turma utilizada neste estudo de caso. Ela pode ser subdividida nas seguintes partes:

- A primeira parte da consulta, que vai da linha 4 (quatro) até a linha 6 (seis), busca o número total de atividades disponíveis dentro da disciplina especificada;
- A disciplina especificada é resultado da segunda parte consulta, que vai da linha 8 (oito) até a linha 14 (quatorze). Essa sub-consulta lista os *id* dos cursos que o aluno *13474* está matriculado.
- A terceira parte da consulta, que vai da linha 16 (dezesseis) até a linha 18 (dezoito), verifica o número de atividades submetidas pelo aluno *13474*

5.3.2 Construção do *adapter*

A etapa final do pré-processamento é a construção do *adapter*. A construção do *adapter* é necessária, pois os algoritmos de mineração só aceitam os dados em um formato específico. Assim, a função deste serviço é recuperar os dados que serão utilizados na mineração, independente do formato em que foram armazenados, e os converter em um formato aceito pelos algoritmos de mineração que serão utilizados.

Normalmente, o usuário precisará implementar o *adapter* para cada objetivo que deseja alcançar com a mineração. Pois, para atingir o objetivo pretendido, os algoritmos de mineração precisam receber um subconjunto específico dos dados, e em um formato específico.

Desta forma, foi construído um *adapter* que lê o arquivo com os dados armazenados pelo pré-processamento executado e os transforma no formato que é compreendido pelo *framework*. Após a construção, o *adapter*, que é um serviço Web semântico, é adicionado ao *framework* para que possam ser utilizados nas etapas posteriores.

Na Figura 5.3 é apresentada a tela do *wizard* que possibilita a adição de novos *adapters* ao *framework*. Depois de adicionados, os *adapters* ficam disponíveis e podem ser reutilizados.

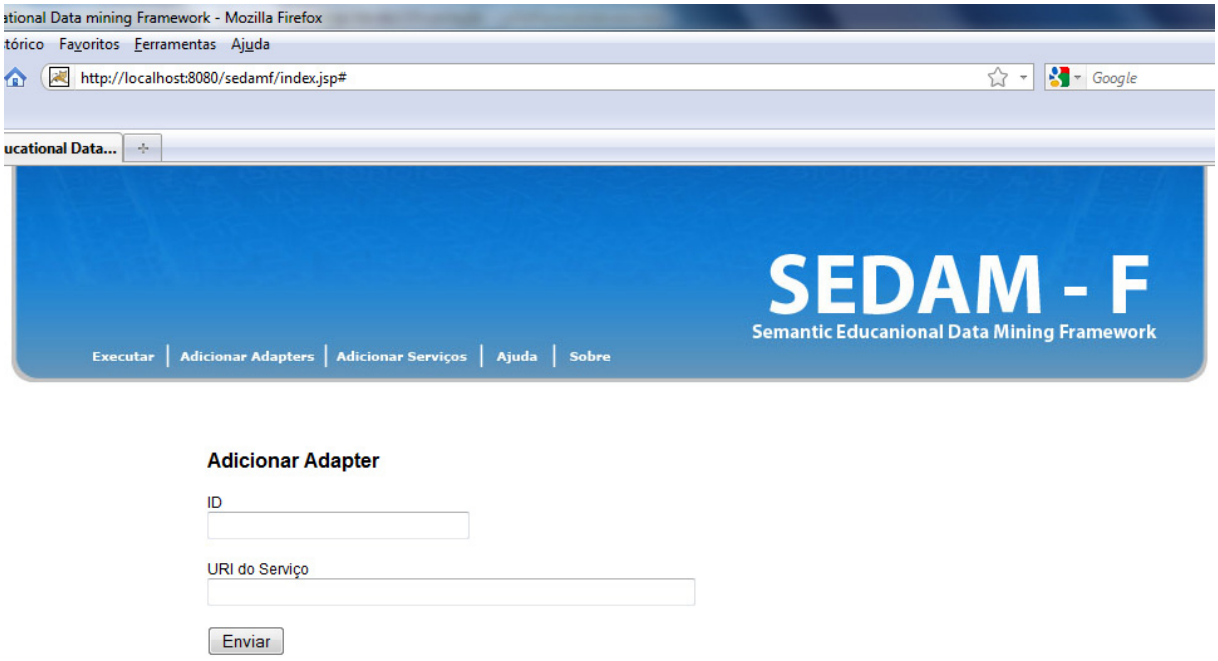

Figura 5.3: Adicionar *Adapter*

5.4 Mineração

Após o pré-processamento, foi necessário definir qual a técnica e algoritmo de mineração a serem adotados, tendo em vista o objetivo a ser alcançado com a mineração, que neste estudo de caso é classificar os alunos em: não motivado, pouco motivado ou motivado. Para realizar a mineração foi escolhido o algoritmo de classificação J48 [47].

A etapa seguinte é verificar se os serviços disponibilizados pelo *framework* atendem as necessidades do usuário. No contexto deste estudo de caso, foi utilizado o algoritmo de classificação J48 e um serviço de pós-processamento que retorne os alunos e a classe aos quais eles pertencem. Esses serviços já estavam disponíveis no *framework*. Assim, não é necessário a construção destes serviços, pois eles serão reutilizados.

O usuário pode identificar os serviços disponíveis através da interface fornecida pelo *framework*. Isto possibilitará que usuários com pouca experiência na mineração de dados possam utilizar a mineração de maneira mais simples. Na Figura 5.4 é apresentada a interface do *framework* que possibilita a escolha do objetivo desejado.

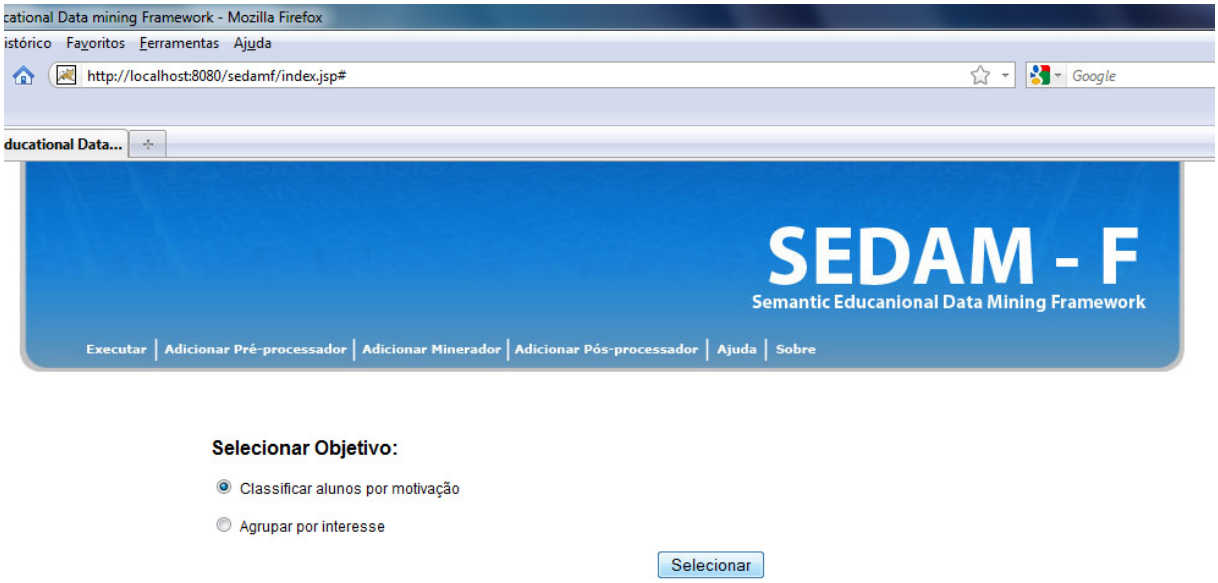

Figura 5.4: Selecionar o Objetivo da mineração

Caso este objetivo não esteja disponível, o usuário precisará implementar e adicionar um novo serviço, que realize a tarefa pretendida, ao *framework*.

O *framework* já oferece vários serviços que encapsulam algoritmos de pré-processamento e de mineração, assim os usuários do *framework* só precisarão construir um novo serviço para obter os dados que serão minerados e o serviço de pós-processamento, se necessário. No entanto, em algum momento, o usuário pode necessitar de um algoritmo que não esteja disponível. Graças as descrições semânticas dos serviços, o *framework* é flexível e dinâmico o suficiente para que esse serviço seja incluído e utilizado logo em seguida. Ou seja, após sua inclusão, o serviço já estará disponível na interface e para o gerenciador de serviços - que é o responsável pelo processo de descoberta, composição e invocação dos serviços.

Nesta etapa, verificou-se que o serviço de pós-processamento necessário também já estava disponível. Assim, não houve a necessidade de construí-lo.

5.5 Execução da Instanciação

Depois de realizar as adaptações necessárias para a execução do *framework*, é possível executar a aplicação de duas formas complementares: (1) Invocar o gerenciador de serviços; e (2) Executar a instância via *wizard*.

5.5.1 Invocar o Gerenciador de serviços

O processo de execução da instancia do *framework* pode ocorrer via código. Este processo ocorre através da invocação do **gerenciador de serviços**.

O gerenciado de serviços é um serviço Web tradicional, por isso, sua invocação é simples e fácil de ser executada. Para se realizar a invocação do gerenciador de serviços é necessário informar alguns parâmetros: parâmetros de entrada e parâmetros de saída. Os parâmetros de entrada são, basicamente, os dados que serão minerados e os parâmetros dos algoritmos de mineração utilizados. Os parâmetros de saída são os objetivos que se pretende alcançar com a mineração.

No entanto, o método de passagem desses parâmetros têm uma forma especifica. Assim, como foi supracitado, os parâmetros são passados por pares, a primeira parte é a URI da ontologia e a segunda parte o seu valor. Isto ocorre porque todos os serviços disponíveis no *framework* são semânticos, ou seja, eles possuem uma ontologia que os descrevem. Assim, para que o *framework* saiba exatamente o que está sendo passado, é necessário apontar na ontologia e informar o valor. Por isso o par [URI, valor].

5.5.2 Executar a instância via *wizard*

A instância do *framework* pode ser invocada através da interface disponibilizada. A interface permite que o usuário escolha o objetivo que se deseja alcançar com a mineração, o *adapter* que será utilizado, a definição dos parâmetros dos algoritmos de mineração, visualizar o resultado da mineração e a etapa final - que é a definição de todos os parâmetros e seus valores. A interface conhece os parâmetros que precisam ser solicitados aos usuários, pois ela acessa as ontologias disponíveis e informa ao usuário quais os valores que precisam ser definidos antes que a execução aconteça. Desta forma, não é necessário que o usuário acesse as ontologias para saber as URIs dos serviços que ele deseja utilizar e, assim, saber quais parâmetros precisam ser definidos, como acontece no maneira de invocação anterior. Além disso, a interface permite ainda a interação entre essas etapas com a finalidade de possibilitar o ajuste dos parâmetros dos algoritmos de mineração.

O processo de invocação acontece da em 6 passos:

O Passo 1 é apresentado na Figura 5.5. Neste passo o usuário já selecionou a opção executar no menu superior e esta tela é apresentada. Nesta tela o usuário deve escolher qual o objetivo deseja alcançar com a mineração de dados educacional. Neste caso, selecionamos a classificação de alunos pela motivação, e avançamos ao próximo passo.

A tela seguinte é apresentada na Figura 5.6. Neste passo o usuário deve escolher o *adapter* que deseja usar. Este *adapter* já foi adicionado no processo de instanciação do *framework*. Neste passo, escolhemos o *adapter* ClassificacaoMotivacao que adicionamos anteriormente e vai recuperar os dados que precisamos para realizar a mineração, e avançamos para o próximo passo.

Neste Passo 3, representado na Figura 5.7, iremos definir os parâmetros do algoritmo de mineração, que é o algoritmo de classificação J48 como supracitado. Os parâmetros do

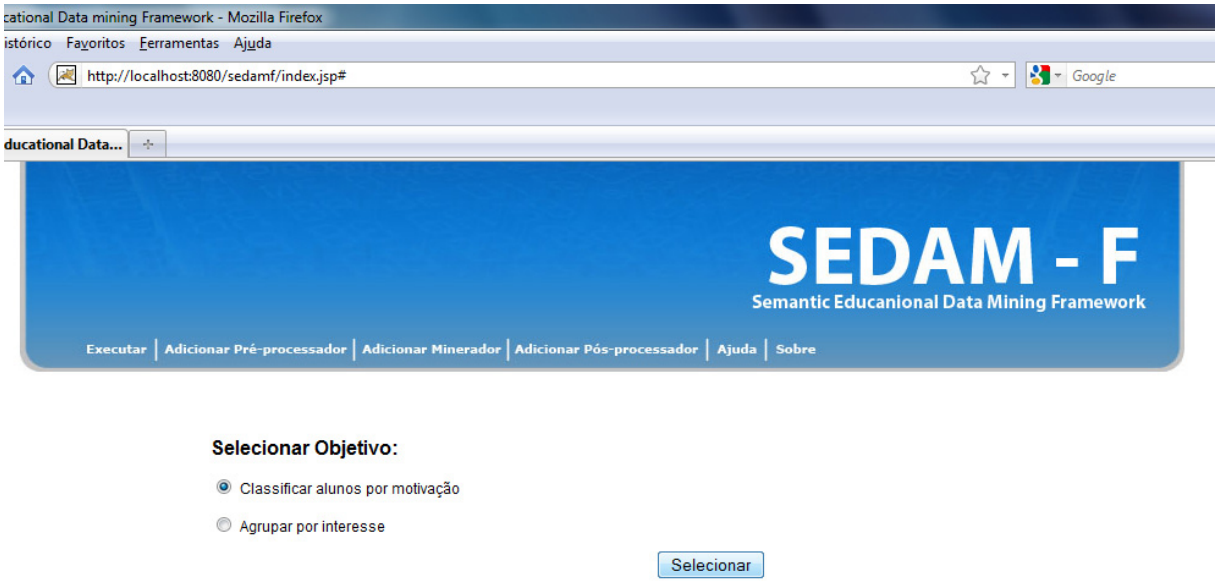

Figura 5.5: Passo 1: Escolha do objetivo da mineração

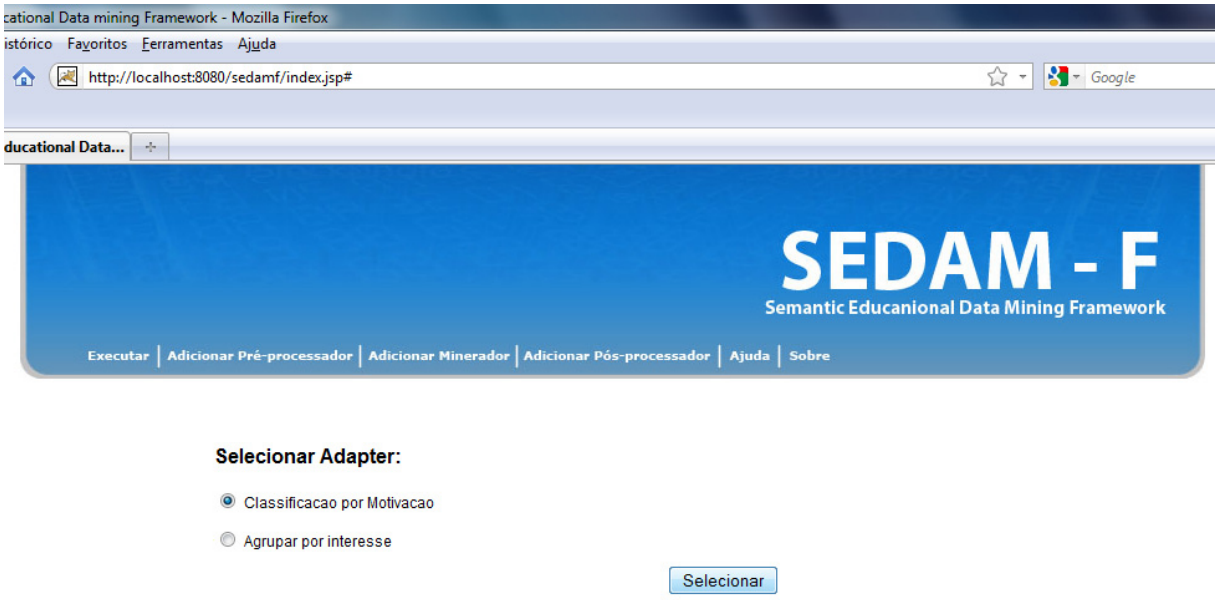

Figura 5.6: Passo 2: Escolha do adapter

algoritmo de classificação são as bases de treinamento e testes necessárias, que devem ser fornecidas pelo usuário, para a construção do modelo que será utilizado nos passos seguintes para classificar os alunos em: não motivados, pouco motivados e motivados.

Como pode ser observado na Figura 5.8, no Passo 4, iremos visualizar o resultado da execução do algoritmo de classificação sobre a base de treinamento e a base de testes. O resultado exibido neste passo apresenta valores importantes que usaremos para avaliar se o modelo construído é eficiente. Se sim, passaremos ao passo seguinte. Caso contrario, podemos voltar ao passo anterior para alterar os parâmetros e ajustar o modelo construído.

No Passo 5, apresentado nas Figuras 5.9 e 5.10, pode-se visualizar o resultado da apli-

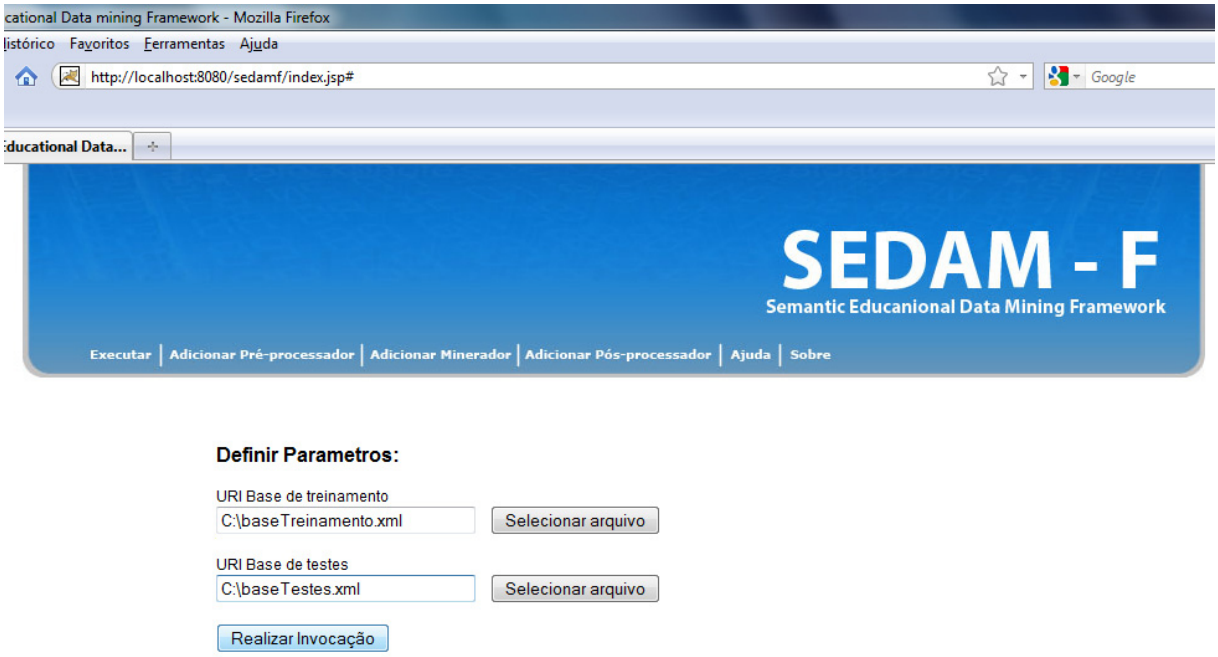

Figura 5.7: Passo 3: Definição dos parametros de mineração

cação do modelo construído nos dados passados inicialmente através do *adapter*. O resultado apresentado, mostra o conteúdo do XML gerado como saída. Este arquivo contém o número de matricula e o status do aluno; Desmotivado, Pouco motivado e Motivado. Nesse passo podemos verificar se o resultado foi satisfatório e encerrar o processo ou retornar a definição dos parâmetros para tentar ajustar o modelo.

No Passo 6, o último passo, é apresentada a saída final do *framework*. Esta saída contém o código que serão usados nos ambientes *e-Learning*, para que estes ambientes possam invocar o *framework* e utilizar a mineração de dados de maneira simples e eficiente. Este código que será usado pelos ambientes é composto por: parâmetros dos algoritmos que precisam ser definidos e seus valores. Ou seja, quais as URIs das ontologias que serão utilizadas, para informar as entradas e saídas desejadas, e seus valores.

5.6 Adaptação do Ambiente

Após a execução da mineração através da *wizard* fornecido pelo *framework*, é necessário a adaptação do ambiente para que este possa utilizar a mineração para prover funcionalidades e vantagens a educadores, professores e alunos. Esta etapa deve ser realizada nos ambientes dependendo de como seus desenvolvedores pretendem utilizar tal informação. Neste estudo de caso, foi realizada a classificação dos alunos com a finalidade de identificar se os alunos estão motivados ou não.

Desta forma, é necessário identificar a quem interessa e como a informação gerada será utilizada. Dado o contexto e as informações geradas, pode-se concluir que essas informações

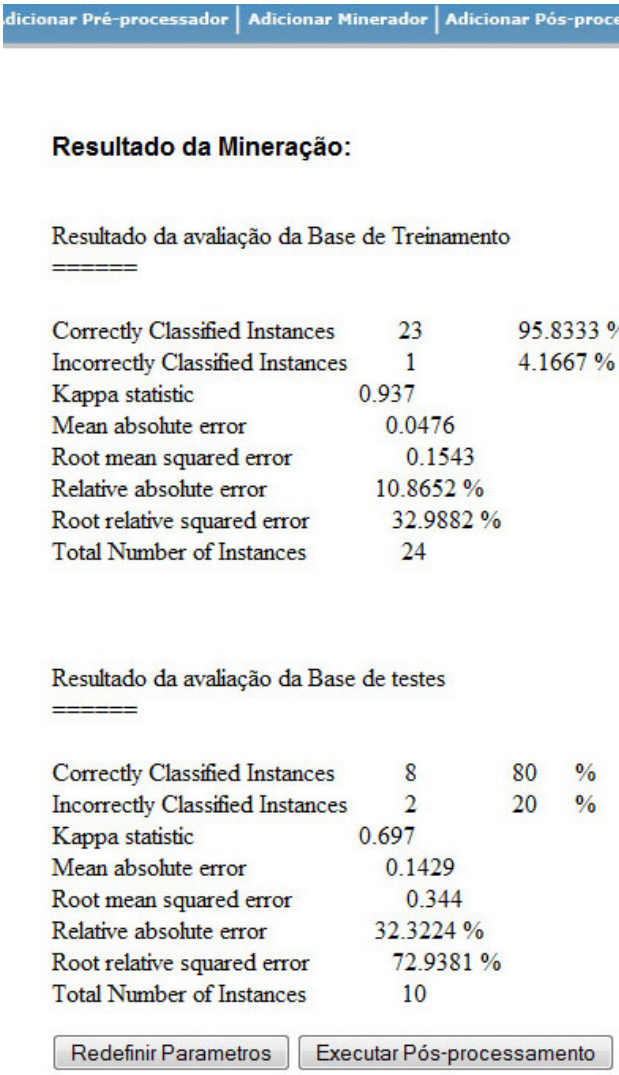

Figura 5.8: Passo 4: Resultado do algoritmo de mineração; Voltar ao passo 3 ou avançar ao passo 5

poderiam estar disponíveis a professores e outros educadores envolvidos. Assim, concluiu-se também que cada professor gostaria de identificar quais alunos são motivados ou não na sua disciplina, que pode estar iniciando ou em andamento.

Após chegar na definição do que é necessário alterar e onde isso será feito, optou-se por acrescentar um link denominado de: "Verificar motivação da turma"em cada disciplina. Assim, o professor pode selecionar este link e será apresentada a ele uma página com os alunos e seus status: Não motivado, Pouco motivado ou Motivado. Para isso, é necessário invocar o *framework* a cada vez que o professor clicar neste link. Além disso, é necessário também construir uma pagina que terá a função de receber o resultado da mineração e exibir ao professor. Vale ressaltar que o *framework* exporta os resultados das minerações executadas no formato XML, e que a utilização desse arquivo é de responsabilidade do autor da aplicação, ou seja, do cliente do *framework*.

Figura 5.9: Passo 5: Resultado do pós-processamento; Voltar ao Passo 4 ou Avançar ao passo 6

Assim, a adaptação deste ambiente consistirá na adição de um link na referida disciplina e a criação de uma pagina para receber os resultados. O código que vai efetuar a execução do *framework* precisa ser construído, assim como a página. Mas como já foi disponibilizado pelo *wizard* os parâmetros e os valores necessários para sua invocação, o processo consistirá na invocação de um serviço Web tradicional, cujo o processo é simples de ser realizado.

5.7 Análise dos Resultados e Discussão

No estudo de caso, o *framework* foi instanciado para realizar a classificação de alunos de um curso de educação a distância quanto a sua motivação. Para isto, foram definidas algumas meta-informações. Dentre essas meta-informações, algumas não se mostraram relevantes para a classificação realizadas, como: Mensagens enviadas para tutores, professores e colegas; e a média mensal de atualização dos blogs e visita de blogs de terceiros. Pois, essas ferramentas são raramente utilizadas pelos alunos. As meta-informações que se mostraram mais relevantes foram: média de acesso mensal, número de acessos ao fórum, média de participação nos fóruns e Percentual de atividades submetidas.

Ainda relacionado aos resultados obtidos, é apresentado na Figura 5.8 os valores alcançados na construção dos modelos. O modelo gerado a partir da base de treinamento classificou 95%

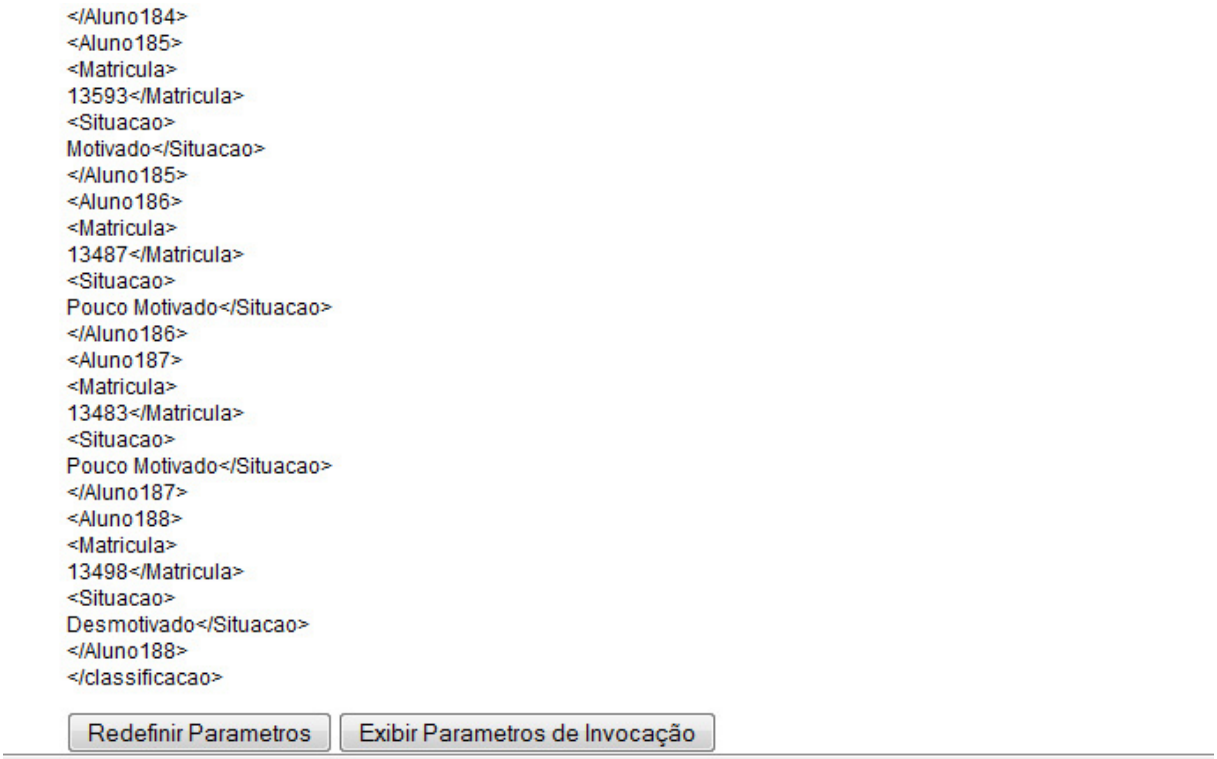

Figura 5.10: Passo 5: Resultado do pós-processamento; Voltar ao Passo 4 ou Avançar ao passo 6

das instâncias corretamente, enquanto o modelo gerado a partir da base de testes classificou 80% das instancias corretamente. Vale ressaltar que as instâncias não classificadas corretamente referem-se a casos polêmicos, onde os alunos participaram bem de uma atividade, mas não tão bem de outras atividades de uma mesma disciplina. A conclusão alcançada após uma avaliação manual cuidadosa foi que em casos como esses, até mesmo um agente humano teria dificuldade de classificá-lo como motivado, pouco motivado ou não motivado. Uma solução para esse impasse poderia ser considerar o tempo como mais uma meta-informação no processo de mineração. Assim, seria possível perceber se a(s) atividade(s) faltantes são as primeiras ou as últimas, o que poderia induzir a interpretações diferentes dos dados.

Além disso, a base de dados utilizada não é bem estruturada; um exemplo desse problema é a ausência de chaves estrangeiras em tabelas que fazem relacionamentos com outras tabelas. Essa total falta de estrutura dificultou o processo de seleção dos dados e a construção do serviço de pré-processamento, responsável por obter as meta-informações do banco de dados e armazená-las no formato ARFF com o intuito de serem mineradas.

Em termos de tempo de execução, a execução total do estudo de caso realizada em aproximadamente 07 horas, sendo 06 horas para desenvolver o pré-processador, 40 Minutos para desenvolver o pós-processador, 3 minutos para instanciar o produto utilizando o *wizard*, incluindo o reuso do algoritmo de mineração para classificação dos dados, e 20 segundos para executá-lo. Note que partindo de uma implementação pronta do pré-processador e pós-

processador, o reuso e flexibilidade do *framework*, aliado à facilidade do *wizard* proporcionam um rápido desenvolvimento e instanciação do produto.

O objetivo do presente estudo de caso foi avaliar o *framework* proposto de acordo com as questões levantadas por Romero e Ventura [49] e apresentadas como motivação do trabalho.

A primeira questão é relacionada com a falta de ferramentas e técnicas gerais que podem ser aplicadas a vários ambiente educacionais. Com a opção pela construção de um *framework*, a solução proposta consistiu na criação de uma ferramenta flexível que permite tanto a adaptação e reuso de ferramentas e técnicas, quanto a extensão do *framework* com a adição de novas ferramentas e técnicas. Além disso, com a flexibilidade do *framework*, os seus usuários podem adaptá-lo de acordo com suas necessidades, mesmo que não tenham sido idealizadas inicialmente.

Outra questão de Romero e Ventura está relacionada à integração das ferramentas de mineração de dados com ambientes educacionais para que os resultados de tarefas de mineração possam ser requisitados e aplicados diretamente no ambiente. O *framework* proposto também atendeu bem a essa questão. Uma questão tecnológica que facilitou consideravelmente essa integração foi a opção por serviços Web, através da construção de um *framework* orientado a serviços. Esta facilidade de integração se dá ao fato que serviços Web tem como principal vantagem a interoperabilidade, que é obtida pelo uso de padrões difundidos na Web. Seu uso independe da linguagem em que foram desenvolvidos, o que garante uma integração simples, independentemente que haja ou não diferenças entre as linguagens de programação do *framework* e do ambiente. Essa qualidade não é garantida no caso dos *frameworks* orientados a objetos tradicionais. No entanto, a adoção de serviços Web tradicionais não satisfaz totalmente a questão levantada. A execução da mineração como uma atividade única seria complexa, pois, apesar da facilidade no uso dos serviços web, o usuário ainda teria que realizar a composição dos serviços web manualmente, o que traria um custo adicional para utilização do *framework* proposto. Desta forma, o *framework* proposto foi desenvolvido utilizando serviços web semânticos e através dessa descrição semântica foi possível a utilização de um gerenciador de serviços que realiza automaticamente toda a parte de descoberta, composição e invocação através de uma interface única. Além de abstrair essas tarefas do usuário, o gerenciador de serviços semânticos possibilita que instâncias do *framework* sejam acessadas através de uma única atividade, responsável pela execução do serviço semântico e por todos todos os passos anteriores necessários (localização e composição). Além da praticidade, a existência de um gerenciador central também proporciona um maior dinamismo ao processo de execução, permitindo inclusive a execução de diferentes serviços apenas com a alterações de parâmetros de uso e configuração. Outra vantagem dos uso de serviços web semânticos consiste na manutenção do sistema. Devido à definição dinâmica das composições de serviços, novos serviços podem ser reconhecidos em tempo de execução, sem que haja a interrupção do sistema.

Um outro ponto está relacionado a complexidade no uso de ferramentas por educadores e

usuário não especialistas em mineração de dados. Essa questão, que também é levantada por Romero e Ventura [49], aponta a necessidade de ferramentas que apresentem uma interface intuitiva e de simples utilização, sem a necessidade da configuração de parâmetros para simplificar a configuração e execução. Neste ponto, o *framework* proposto atende parcialmente a questão levantada. A solução proposta abordou essa questão através um *wizard* gráfico intuitivo e de fácil utilização, que possibilita ao usuário estender o *framework*, além de realizar todo o processo de instanciação e execução do produto final Porém, apesar da abstração de programação proporcionado pelo *wizard*, continua sendo exigido, em alguns momentos da configuração do *framework*, que o usuário tenha um mínimo de conhecimento técnico sobre a mineração de dados educacionais. Pois, no dia-a-dia, o usuário precisará adicionar novos serviços e definir parâmetros dos algoritmos de mineração, bem como avaliar os modelos construídos para verificar a necessidade de alteração desses parâmetros. Entretanto, mesmo com a exigência de conhecimentos em mineração de dados, o *wizard* possibilita uma redução de tempo no processo de instanciação e facilita o processo de adaptação do ambiente educacional, visto que todo o processo de instanciação e execução dessa instância pode ser realizado automaticamente pelo *wizard*, que inclusive apresenta os parâmetros e valores que devem ser passados pelo ambiente educacional no momento da execução do produto instanciado.

Para possibilitar o funcionamento do *wizard* gráfico foi necessário construir um algoritmo para realizar o processo de descoberta e composição dos serviços disponíveis. Pois, os algoritmos disponibilizados não satisfaziam as particularidades necessárias para o funcionamento do *wizard* gráfico da maneira que foi definida. O algoritmo desenvolvido foi utilizado para estender o *grinv* e utilizá-lo como parte da solução proposta. Vale ressaltar que apesar do *grinv* não ser contribuição direta desse trabalho, o algoritmo de composição que o estendeu é considerado uma contribuição secundária deste trabalho.

As principais vantagens que podem ser percebidas no presente estudo de caso, especialmente na utilização do *framework* proposto, são: i) Reuso de ferramentas e técnicas; ii) Fácil integração com ambientes *e-Learning*; iii) Interface simples e intuitiva; iv) Adição de serviços dinamicamente; v) Abstração de estruturas e linguagens de programação, através do *wizard*.

Em termos de desvantagens, uma das principais desvantagens do *framework* é relacionada ao esforço para desenvolver novos serviços. Essa dificuldade pode ser justificada pelo fato do *framework* ser baseado em serviços Web semânticos e a construção desses serviços ainda não ser uma tarefa simples. A principal dificuldade está relacionada ao mapeamento que precisa ser realizado entre o serviço web tradicional e as ontologias, que fornecem a descrição semântica dos serviços, para transformá-lo em serviço web semântico.

Capítulo 6

Considerações Finais

Este trabalho apresentou um *framwork* para mineração de dados educacional orientado a serviços Web semânticos. Esta ferramenta proporciona aos seus usuários facilidades no uso da mineração de dados para fins educacionais, disponibilizando serviços que encapsulam algoritmos de pré-processamento, mineração e pós-processamento, que compostos fornecem funcionalidades educacionais e podem ser utilizados pelos ambientes *e-Learning* por meio da invocação do gerenciador de serviços. O *framework* disponibiliza ainda um *wizard*, que permite realizar desde o processo de extensão até o processo de instanciação do *framework* de maneira simples.

Foi apresentado um estudo de caso com o objetivo de realizar uma avaliação do *framework* proposto. Neste estudo de caso foi utilizada a base de dados de um ambiente *e-Learning* real para realizar a classificação de alunos de acordo com a motivação. Para tal, foram definidas e extraídas meta-informações de cada aluno da turma escolhida. E apesar de não ser o foco principal deste trabalho, o modelo gerado no processo de mineração foi satisfatório, atingindo um taxa 95% de acerto na classificação das instâncias da base de treinamento e 80% de acerto na classificação das instâncias da base de testes, visto que algumas instâncias eram ambíguas e difíceis de serem classificadas até por um agente humano. Ainda no estudo de caso, foram avaliados os resultados obtidos com o uso *framework* diante dos objetivos levantados. Neste ponto, o *framework* atendeu bem a dois dos objetivos levantados: integração com ambientes e reuso de técnicas e/ou ferramentas. Ao outro objetivo apresentado, necessidade de uma interface simples e intuitiva para usuários não especialistas, o *framework* atendeu parcialmente. Pois, apesar de oferecer um *wizard* bem simples e que reduz o tempo dos processos de extensão e instanciação do *framework*, o usuário ainda precisa conhecer o processo de extração de conhecimento e algoritmos de mineração para definir os parâmetros desses algoritmos. Pode-se avaliar também as vantagens e desvantagens na utilização da solução proposta para a realização da mineração de dados educacional.

Como trabalhos futuros pretende-se: i) concluir o desenvolvimento do *framework* visto que a mesma encontra-se em fase de protótipo; ii) Realizar mais avaliações da solução proposta

em ambientes *e-Learning* diferentes; iii) Desenvolver novos estudos em domínios diferentes; iv) Adicionar novos serviços de pré-processamento, mineração e pós-processamento.

Referências

[1] Baker, R. S. (*in press*), Data mining for education, *in* 'International Encyclopedia of Education (3rd edition)', Elsevier, Oxford, UK.

[2] Barros, H. (2011), Um middleware adaptável para descoberta, composição e invocação automática de serviços web semânticos, Master's thesis, Universidade Federal de Alagoas.

[3] Battle, S., Bernstein, A., Boley, H., Grosof, B. & Gruninger, M. (2005), 'Semantic web services framework (swsf)', http://www.w3.org/Submission/2005/07/.

[4] Benavides, D., Ruiz-Cortés, A., Trinidad, P. & Segura., S. (2006), A survey on the automated analyses of feture models, *in* 'Jornadas de Ingeniería del Software y Bases de Datos (JISBD)'.

[5] Berners-Lee, T., Hendler, J. & Lassila, O. (2001), The semantic web, A.

[6] Burstein, M., Hobbs, J., Lassila, O., Mcdermott, D., Mcilraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N. & Sycara, K. (2004), 'OWL-S: Semantic Markup for Web Services', Website. URL `http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/`.

[7] C.Daconta, M., Obrst, L. J. & Smith, K. T. (2003), *The Semantic Web: A Guide to the Future of XML, Web Services and Knowledge Manangement*, Joe Wilkert.

[8] Community, E. D. M. (2010), Disponível em http://educationaldatamining.org/. Acessado em 10 de Abril de 2010.

[9] Cotter, S. & Potel, M. (1995), *Inside Taligent Technology*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[10] Devedzic, V. (2004), 'Education and the semantic web', *International Journal of Artificial Intelligence in Education* **14**, 39–65.

[11] Devedzic, V. (2006), *Semantic Web and Education (Integrated Series in Information Systems)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[12] Dietze, S., Gugliotta, A. & Domingue, J. (2007), A semantic web service oriented framework for adaptive learning environments, *in* 'ESWC '07: Proceedings of the 4th European conference on The Semantic Web', Springer-Verlag, Berlin, Heidelberg, pp. 701–715.

[13] Donnellan, D. & Pahl, C. (2002), Data mining technology for the evaluation of web-based teaching and learning systems, *in* M. Driscoll & T. C. Reeves, eds, 'Proceedings of World Conference on E-Learning in Corporate, Government, Healthcare, and Higher

Education 2002', AACE, Montreal, Canada, pp. 747–752. URL http://www.editlib.org/p/15296.

[14] Erl, T. (2005), *Service-Oriented Architecture: Concepts, Technology, and Design*, Prentice Hall PTR, Upper Saddle River, NJ, USA.

[15] Etzioni, O. (1996), 'The world-wide web: quagmire or gold mine?', *Commun. ACM* **39**(11), 65–68.

[16] Fayad, M. E., Schmidt, D. C. & Johnson, R. E. (1999), *Building Application Frameworks: Object-Oriented Foundations of Framework Design*, John Wiley & Sons, Inc., New York, NY, USA.

[17] Fayyad, U. M. (1996), 'Data mining and knowledge discovery: Making sense out of data', *IEEE Expert: Intelligent Systems and Their Applications* **11**(5), 20–25.

[18] Fayyad, U., Piatetsky-shapiro, G. & Smyth, P. (1996*a*), 'From data mining to knowledge discovery in databases', *AI Magazine* **17**, 37–54.

[19] Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P. (1996*b*), 'The kdd process for extracting useful knowledge from volumes of data', *Commun. ACM* **39**(11), 27–34.

[20] Goebel, M. & Gruenwald, L. (1999), 'A survey of data mining and knowledge discovery software tools', *SIGKDD Explor. Newsl.* **1**, 20–33. URL http://doi.acm.org/10.1145/846170.846172.

[21] Han, J. (2005), *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[22] Hand, D. J., Smyth, P. & Mannila, H. (2001), *Principles of data mining*, MIT Press, Cambridge, MA, USA.

[23] Jialiandosmarr, L. & Zaiane, O. R. (2004), Combining usage, content, and structure data to improve web site recommendation, *in* 'In 5th International Conference on Electronic Commerce and Web Technologies (EC-Web)', pp. 305–315.

[24] Johnson, R. E. & Foote, B. (1988), 'Designing reusable classes', *Journal of Object-Oriented Programming* **1**(2), 22–35.

[25] Johnson, R. E. & Russo, V. F. (1991), Reusing object-oriented designs, Technical report, A.

[26] Kang, K. C., Cohen, S. G., Hess, J. A., Novak, W. E. & Peterson, A. S. (1990), Feature-oriented domain analysis (foda) feasibility study, Technical report, Carnegie-Mellon University Software Engineering Institute.

[27] Khodeir, N., Wanas, N. M., Darwish, N. M. & Hegazy, N. (2010), Inferring the differential student model in a probabilistic domain using abduction inference in bayesian networks, *in* 'In Educational Data Mining', pp. 299–300.

[28] Kim, S. M. & Calvo, R. A. (2010), Sentiment analysis in student experiences of learning, *in* 'In Educational Data Mining', pp. 111–120.

[29] Kosala, R. & Blockeel, H. (2000), 'Web mining research: a survey', *SIGKDD Explor. Newsl.* **2**(1), 1–15.

[30] Kurgan, L. A. & Musilek, P. (2006*a*), 'A survey of knowledge discovery and data mining process models', *Knowl. Eng. Rev.* **21**(1), 1–24.

[31] Kurgan, L. A. & Musilek, P. (2006*b*), 'A survey of knowledge discovery and data mining process models', *Knowl. Eng. Rev.* **21**, 1–24. URL `http://portal.acm.org/citation.cfm?id=1166026.1166027`.

[32] Landin, N. & Niklasson, A. (1995), Development of object-oriented frameworks, Technical report, A.

[33] Macfadyen, L. & Sorenson, P. (2010), Using lims (the learner interaction monitoring system) to track online learner engagement and evaluate course design, *in* 'In Educational Data Mining', pp. 301–302.

[34] Markellou, P., Mousourouli, I., Spiros, S. & Tsakalidis, A. (2005), 'Using semantic web mining technologies for personalized e-learning experiences', *Web-Based Education* **Z**, 461–826.

[35] Markiewicz, M. E. & Lucena, C. J. (2001), 'Object oriented framework development', http://www.acm.org/crossroads/xrds7-4/frameworks.html. Último acesso em Outubro de 2010.

[36] Martin, D., Burstein, M., Hobbs, E., Lassila, O., Mcdermott, D., Mcilraith, S., Narayanan, S., Parsia, B., Payne, T., Sirin, E., Srinivasan, N. & Sycara, K. (2004), Owl-s: Semantic markup for web services, Technical report, A. URL `http://www.w3.org/Submission/OWL-S/`.

[37] Mattsson, M., Mattsson, M. & Mattsson, M. (1996), 'Object-oriented frameworks - a survey of methodological issues'.

[38] McIlraith, S. A., Son, T. C. & Zeng, H. (2001), 'Semantic web services', *IEEE Intelligent Systems* **16**(2), 46–53.

[39] Merceron, Agathe; Yacef, K. (2005), 'Tada-ed for educational data mining', *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning* **7**, A.

[40] Mühlenbrock, M. (2005), Automatic action analysis in an interactive learning environment, *in* 'Proceedings of the workshop on Usage Analysis in Learning Systems at the 12th International Conference on Artificial Intelligence in Education AIED-2005', Amsterdam The Netherlands, pp. 73–80.

[41] Mitra, S., Pal, S. & Mitra, P. (2002), 'Data mining in soft computing framework: a survey', *Neural Networks, IEEE Transactions on* **13**(1), 3 –14.

[42] Nagata, R., Takeda, K., Suda, K., Kakegawa, J. & Morihiro, K. (2009), Edu-mining for book recommendation for pupils, *in* 'In Educational Data Mining', pp. 91–100.

[43] Noy, N. F. & mcguinness, D. L. (2001), 'Ontology development 101: A guide to creating your first ontology', Online. URL `http://www.ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-noy-mcguinness.html`.

[44] Payne, T. R. & Lassila, O. (2004), 'Guest editors' introduction: Semantic web services', *IEEE Intelligent Systems* **19**(4), 14–15.

[45] Pedrinaci, C., Domingue, J. & Sheth, A. (2010), 'Semantic web services'.

[46] Polleres, A., Bussler, C., Travers, G., Domingue, J. B. & Burdett, D. (2005), 'Web service modeling ontology (wsmo) submission', http://www.w3.org/Submission/2005/06/.

[47] Quinlan, J. R. (1993), *C4.5: programs for machine learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[48] Ranka, A., Anwar, F. & Chae, H. S. (2010), Pundit: Intelligent recommender of courses, *in* 'In Educational Data Mining', pp. 339–340.

[49] Romero, C. & Ventura, S. (2007), 'Educational data mining: A survey from 1995 to 2005', *Expert Systems with Applications* **33**(1), 135 – 146. URL `http://www.sciencedirect.com/science/article/B6V03-4JW10WR-2/2/7df5551b2738c1402738eece307e6c61`.

[50] Romero, C., Ventura, S. & Bra, P. D. (2005), 'Knowledge discovery with genetic programming for providing feedback to courseware authors', *User Modeling and User-Adapted Interaction* **14**, 425–464. URL `http://portal.acm.org/citation.cfm?id=1058047.1058086`.

[51] Sacín, C. V., Agapito, J. B., Shafti, L. & Ortigosa, A. (2009), Recommendation in higher education using data mining techniques, *in* 'In Educational Data Mining', pp. 191–199.

[52] Shadbolt, N., Berners-Lee, T. & Hall, W. (2006), 'The semantic web revisited', *IEEE Intelligent Systems* **21**, 96–101. URL `http://dx.doi.org/10.1109/MIS.2006.62`.

[53] Shaw, M. & Garlan, D. (1996), *Software Architecture: Perspectives on an Emerging Discipline*, 1st ed., Prentice Hall.

[54] Studer, R., Benjamins, V. R. & Fensel, D. (1998), 'Knowledge engineering: Principles and methods'.

[55] Taligent (1994*a*), Building object-oriented frameworks, Technical report, Taligent Inc.

[56] Taligent (1994*b*), Leveraging object-oriented frameworks, Technical report, Taligent Inc.

[57] Tang, C., Yin, H., Li, T., Lau, R. W. H., Li, Q. & Kilis, D. (2000), Personalized courseware construction based on web data mining, *in* 'WISE '00: Proceedings of the First International Conference on Web Information Systems Engineering (WISE'00)-Volume 2', IEEE Computer Society, Washington, DC, USA, p. 2204.

[58] Tang, T. & McCalla, G. (2003), 'Smart recommendation for an evolving e-learning system', *Workshop on Technologies for Electronic Documents for Supporting Learning, International Conference on Artificial Intelligence in Education* **A**, 699–710.

[59] Tang, T. Y. & McCalla, G. (2002), Student modeling for a web-based learning environment: a data mining approach, *in* 'Eighteenth national conference on Artificial intelligence', American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 967–968.

[60] Tsai, C.-Y. & Tsai, M.-H. (2005), A dynamic web service based data mining process system, *in* 'CIT '05: Proceedings of the The Fifth International Conference on Computer and Information Technology', IEEE Computer Society, Washington, DC, USA, pp. 1033–1039.

[61] W3C (n.d.), 'W3c semantic web activity'. URL `http://www.w3.org/2007/03/layerCake.png`, Last access in April 2011.

[62] Witten, I. H. & Frank, E. (2005), *Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)*, Morgan Kaufmann. URL `http://www.cs.waikato.ac.nz/~ml/weka/book.html`.
WSDL-S Submission Request to W3C

[63] WSDL-S Submission Request to W3C *(n.d.), http://www.w3.org/Submission/2005/10/. Last access in June 2010.*

[64] Zaane, E. O. & Zaïane, O. R. (2001), 'Web usage mining for a better web-based learning', A **A***, 60–64.*

[65] Zaiane, O. R. (2002), 'Building a recommender agent for e-learning systems', A **A***, 55.*