Descoberta de Conhecimento em Processos de *Workflow* Modelados e Executados sobre Plataforma *Oracle*

Rafael S. Garcia, Duncan D. Ruiz

Pontifícia Universidade Católica do Rio Grande do Sul – Faculdade de Informática

Caixa Postal 90619-900 Porto Alegre – RS – Brasil

rafael_sgarcia@terra.com.br, duncan@inf.pucrs.br

Resumo.

O trabalho apresenta uma ferramenta web, plataforma Java-Tomcat, chamada Workflow Mining, que implementa uma arquitetura de execução de processos de descoberta de conhecimento, sobre base de dados de execuções de Workflow. Nessa arquitetura, algoritmos de mineração e métodos de visualização podem ser dinamicamente integrados, através da redefinição de classes e respectivos métodos. Pode-se, com isso, realizar análises do comportamento de execuções de processos, empregando diferentes configurações, de acordo com a conveniência e disponibilidade dos artefatos (algoritmos e métodos de visualização) disponíveis.

1. Introdução

Atualmente, com a necessidade cada vez maior da informatização das grandes organizações, a utilização de modelagem de processos de negócio através de alguma ferramenta de *workflow* vem se tornando uma prática bastante comum. Diversos processos presentes no dia-a-dia de uma grande empresa podem ser automatizados, desde a solicitação e aprovação de férias, viagens e compra de produtos, até a definição das atividades que um colaborador deve desenvolver dentro de um determinado projeto ou a identificação de um erro em um aplicativo (*bug*), e as etapas necessárias para a sua correção. Alguns desses processos podem ser de grande importância estratégica dentro das organizações e os gestores necessitam de ferramentas para poderem realizar determinadas análises sobre eles e responderem a perguntas como: "Qual o processo que mais atrasa?", "Porque este processo tem um índice muito grande de atrasos?" ou "Qual atividade que está levando mais tempo para ser executada do que foi planejado?".

Ferramentas fornecidas com os Sistemas de Gerência de *Workflow* (SGWf), como o *Oracle Workflow Monitor*, não respondem a estas perguntas. Nessas ferramentas o máximo de interação permitido é fazer um acompanhamento gráfico ou textual do processo, verificando em qual recurso o processo está atualmente, qual o resultado obtido em uma determinada atividade, qual o tempo total de execução do processo, entre outras. Nas organizações, o número de instâncias de processos pode crescer rapidamente. Nestes casos, se um gestor necessitar de uma análise sobre os dados de execuções de processos, terá que contá-los manualmente ou fazer consultas SQL complexas sobre os *logs* da aplicação, ambas as opções de difícil viabilização.

As tecnologias de Workflow [Ley00] e de Processo de Descoberta de Conhecimento (KDD) [Han01] vêm crescendo em visibilidade, projeção e destaque tanto nos meios acadêmicos quanto nos corporativos devido ao potencial em agilizar e qualificar a maneira como os gestores gerenciam seus processos de negócio [Bon01].

Este trabalho apresenta uma ferramenta que define uma arquitetura para a realização de processos de descoberta de conhecimentos sobre base de dados de *Workflow,* modelados e executados sobre plataforma *Oracle-Oracle Workflow* (OWF) [Cha02]. Como ponto positivo da ferramenta está a facilidade de inclusão de algoritmos de mineração de dados e de modos de visualização de resultados no sistema, bastando a implementação de uma classe especial, que utiliza a interface proposta. Este trabalho é resultado da pesquisa para a especificação e implementação completa deste ambiente computacional desenvolvido na PUCRS-FACIN ([Gar05, Gar05a]).

2. Workflow Mining – A ferramenta proposta

Para a especificação da ferramenta de apoio a análise analítica sobre dados de execução de processos de *Workflow,* iniciou-se pela identificação de todas as etapas necessárias para a realização do processo de descoberta de conhecimento, conforme proposto em [Han01]. O SGWf escolhido foi o implementado pela *Oracle*. A sua escolha baseia-se, entre outros fatores, na sua grande aceitação no mercado e na sua arquitetura desenvolvida com base nos padrões determinados pela *Workflow Management Coalition* (WfMC), conforme descrito em [Hol95]. A ferramenta proposta é composta por três módulos, conforme ilustrados na Figura 1: módulo de importação dos dados, módulo de análise dos dados, e módulo de visualização. Um quarto módulo é exibido e identificado como *Oracle Database 9i* e representa o SGBD *Oracle,* mais *Oracle Workflow 2.6.2* contendo todos os *logs* de execução de instâncias de processos.

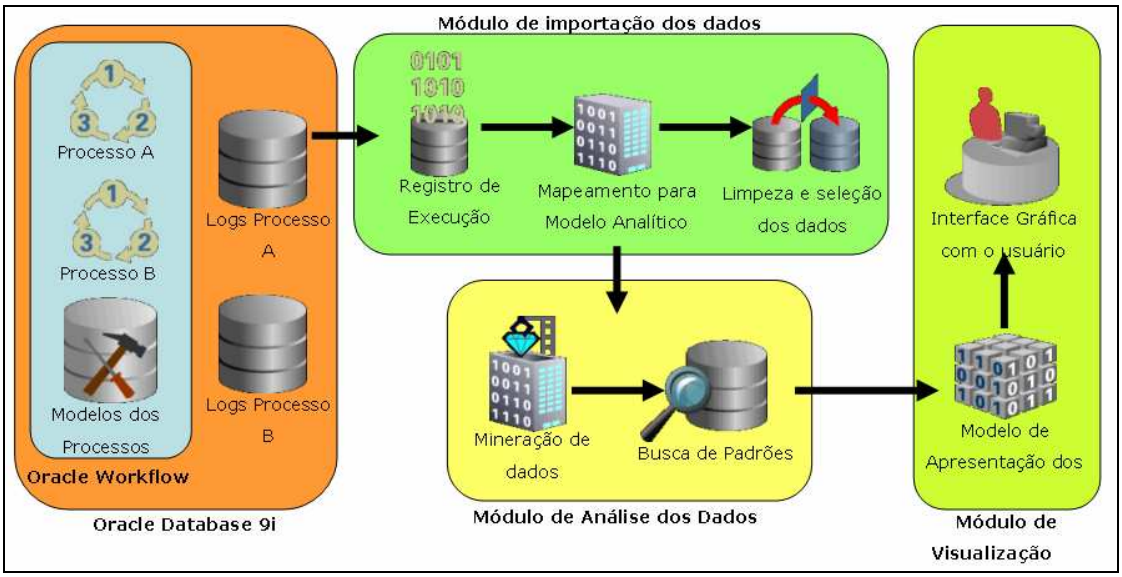

Figura 1 - Arquitetura da ferramenta Workflow Mining

2.1. Pré-Processamento de dados

O primeiro módulo da ferramenta é responsável pelo pré-processamento (etapa de extração, transformação) dos dados que representam a execução dos fluxos de *Workflow*. Segundo [Han01] esta é uma das fases mais importantes do processo de descoberta de conhecimento, consistindo em mais de 50% do tempo utilizado. Nesta etapa os dados que serão minerados são extraídos da base original e passam por processos de avaliação da necessidade de limpeza de ruídos, padronização, integração e transformação. Um exemplo de operações realizadas nesta etapa é a definição de estratégias para tratar atributos com valores nulos. Um trabalho realizado de forma

incorreta nesta fase pode resultar em padrões inconsistentes ou inválidos que só serão descobertos após a análise e interpretação dos resultados, em etapas finais do processo. A partir do modelo de implementação do *Oracle Workflow* são capturados os registros de execução de processos de negócio (ou Trilha de Auditoria) executados nela.

Para que esses dados possam ser analisados por processos de descoberta de conhecimento, é necessário organizá-los em um Modelo Analítico Multidimensional. Para tanto, foi adotado o modelo implementado pela HP [Gri04], e mostrado na Figura 2. A escolha deste modelo baseia-se no fato de já ter sido criado focado no armazenamento de modelos e dados de execução de processos de *Workflow*. Outro fator que influenciou na escolha deste modelo é o fato dele melhor se adequar ao modelo de implementação de *Workflow* do que o outro modelo analisado, descrito em [Ede02]. O ponto positivo em se utilizar um modelo analítico é permitir que a arquitetura possa ser livre quanto ao SGWf utilizado, desde que existam módulos de carga apropriados.

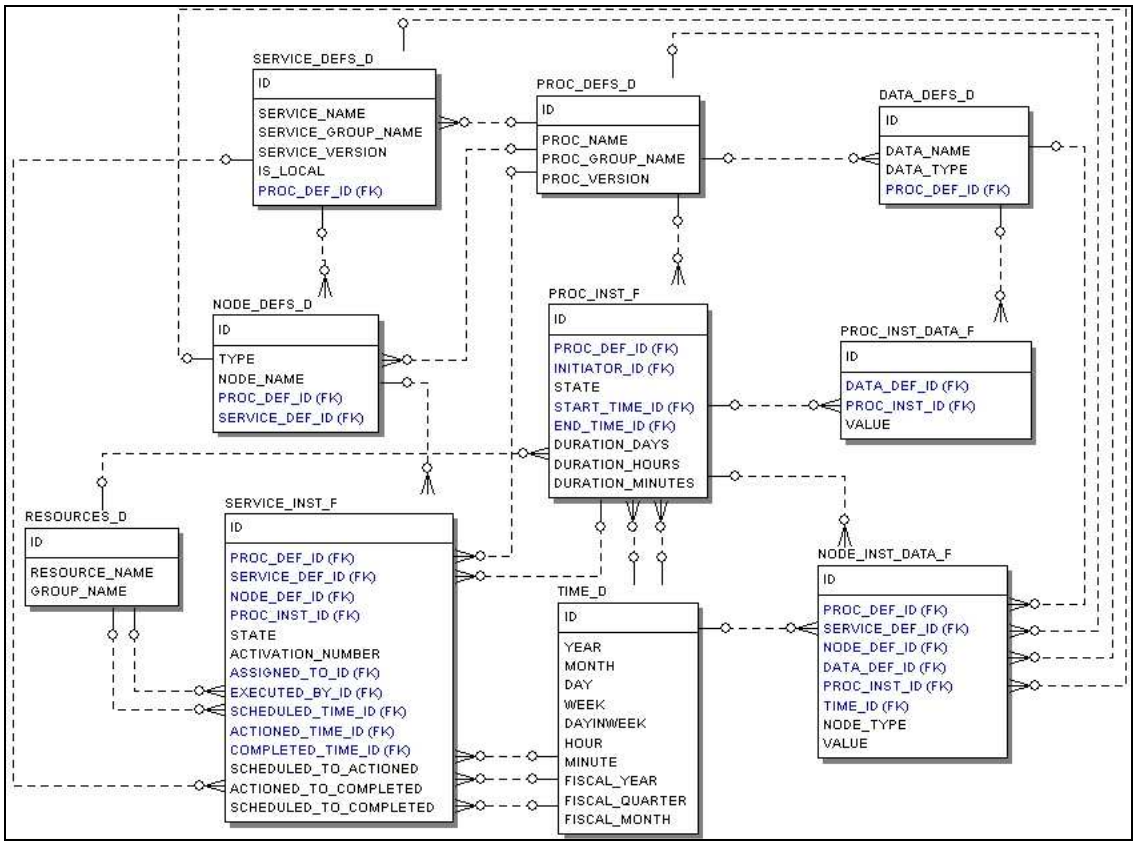

Figura 2 – Modelo Analítico adotado

Para cada tabela que compõe o modelo analítico foi adotada uma determinada estratégia de importação baseada na procedência dos dados recuperados a partir do modelo do OWF. A tabela 1 descreve as dimensões e fatos do modelo analítico adotado. A etapa de pré-processamento de dados foi denominada "importação de dados" por fazer analogia a importar dados de uma base de dados para outra.

2.2. Mineração de dados

O segundo módulo da ferramenta é o responsável pela execução dos algoritmos de mineração de dados e busca por padrões nos processos, ou seja, pela análise e

classificação dos resultados da execução destes algoritmos. Ele foi construído de forma a permitir uma boa infra-estrutura para a incorporação de ferramentas e algoritmos já existentes ou para o desenvolvimento de novos algoritmos de mineração de dados.

Tabela 1: Descrição das dimensões e fatos do modelo analítico adotado

Tipo	Nome	Descrição
Dimensão	Proc_Defs_D	Definição de modelos de processos. São importadas as definições de todos os processos definidos pelo OWF que possuem, pelo menos, uma instância executada. Para a versão de processo, o mesmo conceito foi mantido durante a importação do modelo do OWF, ou seja, foi usada a versão do processo principal.
Dimensão	Resources_D	Definição de usuários e grupos. São importados todos os usuários e papéis que foram definidos como inicializadores de instâncias de processos, ou responsáveis por notificações. Também é criado um usuário padrão que representa o Workflow Engine utilizado na execução de atividades que representam funções automatizadas.
Dimensão	Time_D	Definição de datas. Armazena todas as datas utilizadas na base do OWF, em diferentes modos (i.e. dia, mês, ano, hora e minuto). As datas importadas são de início e fim de execução de instâncias de processos e de início, limite e fim de execução de atividades.
Dimensão	Service_Defs_D	Definição das atividades do tipo função e notificação.
Dimensão	Node_Defs_D	Definição das mesmas atividades de Service_Defs_D, incluindo também os processos. No OWF, pode-se definir uma atividade em um *Item Type* (definição de modelos de processos) e utilizá-la em outro. Basicamente, a diferença entre as duas dimensões é o fato da dimensão dos serviços armazenar o grupo ao qual o serviço pertence, dado não definido no modelo da *Oracle*.
Dimensão	*Arc_Defs*	Definição de arcos entre os nodos, ou seja, as transições entre atividades. Armazena as informações extraídas da tabela correspondente no modelo do OWF.
Dimensão	*Data_Defs_D*	Definição de atributos que são definidos no OWF como três tipos distintos (processo, atividades e mensagem) e armazenados em tabelas específicas. Como o modelo da *Oracle* permite existirem atributos com mesmo nome, um tratamento especial foi necessário para que se pudesse fazer a distinção entre os atributos: o nome do atributo é a concatenação do identificador da origem do dado (*Process*, *Activity* ou *Notification*), do nome do processo, atividade ou notificação e do nome original do atributo.
Fato	*Proc_Inst_F*	Definição das instâncias dos processos. São importadas todas as definições das instâncias executadas e já finalizadas.
Fato	*Service_Inst_F*	Definição das instâncias de atividades. Registra informações de todas as atividades executadas, definidas em *Service_Defs_D*.
Fato	*Proc_Inst_Data_F*	Armazena todos os valores dos atributos definidos para os modelos de processos e para as instâncias já finalizadas, na forma textual.
Fato	*Node_Inst_Data_F*	Armazena os valores dos atributos definidos para as atividades e notificações, que foram executadas nas instâncias importadas. Os valores são armazenados, também, na forma textual.

A incorporação de novos algoritmos no sistema é feita através de um arquivo contendo a sua implementação na linguagem Java (arquivo *jar*). Para que essa incorporação possa ser o mais genérica possível, foi especificada uma interface padrão e cada implementação de algoritmo deverá ter uma classe que implemente essa interface. A partir dela, as operações poderão ocorrer de maneira automática na ferramenta, bastando apenas um cadastro do novo algoritmo e importação da sua implementação.

A interface proposta, chamada *AlgorithmInterface*, define um método denominado *executeAlgorithm*. Cada desenvolvedor, no momento em que for desenvolver a sua classe que implementa esta interface, deve se preocupar em como

buscar os dados no modelo analítico para passar para a sua implementação de algoritmo. Para a busca de dados na base analítica, uma consulta SQL deve ser definida em um arquivo XML e um nome único deve ser atribuído a ela. O parâmetro passado pela ferramenta para o método *executeAlgorithm* é uma instância de uma classe de serviço que possui a conexão com o banco de dados que se está utilizando (atualmente *Oracle*, porém, com possibilidades de alteração de SGBD). Com esta instância, o desenvolvedor pode executar a sua consulta na base de dados e preparar os dados retornados para a execução do seu algoritmo. A única restrição imposta é que o algoritmo só pode ser executado sobre os *logs* de execução de um único modelo de processo e versão previamente selecionados pelo usuário em uma tela específica do sistema. Estes dados também são informados como parâmetros do método executado na interface e também devem ser tratados pelo desenvolvedor do algoritmo.

2.3. Visualização de resultados

Após a identificação das regras de execução dos processos, os dados consolidados são apresentados ao usuário utilizando alguma maneira gráfica, constituindo, assim, o terceiro módulo. Dentre as alternativas, gráficos, fichas texto e recursos de visualização 3D podem ser utilizados. Para os modos de visualização, foi utilizada a mesma idéia de incorporação de implementações, utilizada para os algoritmos de mineração de dados.

Desta forma, a inclusão de novos algoritmos de visualização no sistema é feito através da criação de uma classe que implemente uma segunda interface proposta e que seja encapsulada juntamente com o arquivo de implementação (*jar*). Esta interface é denominada *VisualizationInterface* e define o método *generateVisualization*. Este método também recebe a instância da classe de serviço com o banco de dados (utilizada para buscar o resultado da execução do algoritmo de mineração) e o descritor da página (instância de *JspWriter*). Com o descritor da página, qualquer tipo de visualização pode ser gerada, deste acompanhamento textual até visualizações gráficas baseadas em *Java Applets* ou componentes do tipo *ActiveX*.

Os dados resultantes da execução do algoritmo são salvos em tabelas e este processo é realizado pela mesma instância de serviço de banco de dados passada por parâmetro do método *executeAlgorithm*. Para cada classe de algoritmo de mineração, é previamente definida no sistema uma tabela específica para armazenar os resultados da sua execução. Estas tabelas são definidas baseadas no tipo de algoritmo utilizado. Assim, tem-se uma tabela específica para armazenar resultados de algoritmos do tipo classificação, associação e assim por diante. A Figura 3 ilustra a tela de cadastro de um novo algoritmo no sistema.

3. Considerações finais e próximos passos

Neste trabalho foi apresentada uma ferramenta de análise de *logs* de execução de processos de *Workflow* denominada *Workflow Mining*. Esta ferramenta permite aos gestores das organizações, que possuem seus processos de negócios implementados na ferramenta de *Oracle Workflow,* possam realizar uma análise analítica sobre o comportamento de seus processos, baseada nas técnicas de descoberta de conhecimento.

Os resultados obtidos utilizando-se esta ferramenta auxiliaram no entendimento de como dois processos de negócio de uma empresa de desenvolvimento de *softwares* funcionam, identificando que determinadas atividades atrasam sempre que executadas pelo mesmo usuário. Foram utilizadas para os testes mais de 2.000 instâncias de processos.

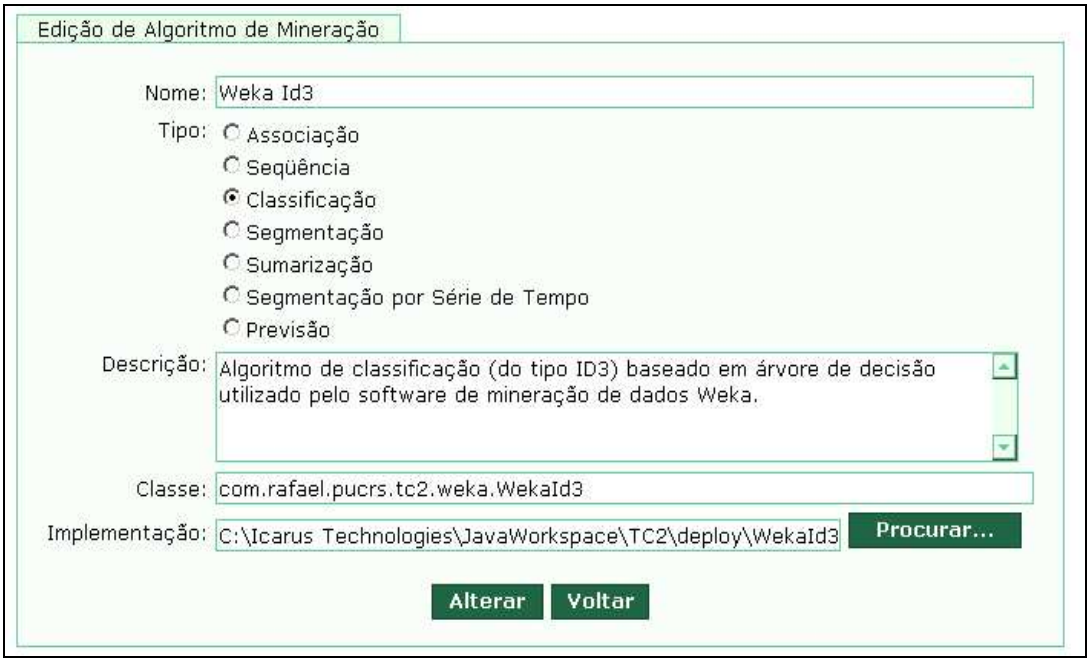

Figura 3 - Cadastro de um novo algoritmo no sistema

Os resultados foram considerados satisfatórios, visto que a ferramenta atingiu o seu objetivo principal, que era definir uma boa arquitetura para a realização de processos de descoberta de conhecimento. Para trabalhos futuros podemos citar: otimização dos scripts de pré-processamento de dados, criação de classes de serviços e arquivos de propriedades para a incorporação automática de outros SGWf para o processo, e um estudo e desenvolvimento de tabelas de armazenamento de resultados para outras classes de algoritmos.

Referências Bibliográficas

Bonifati, A.; Casati, F.; Dayal, U.; Shan, M. (2001). **Warehousing Workflow Data: Challenges and Opportunities.** Roma.

Chang, S.; Jaeckel, C. (2002) **Oracle Workflow Guide.** Release 2.6.2, Volume 1, Oracle Corp.

Eder, J.; Olivotto, G. E.; Gruber, W. (2002). **A Data Warehouse for Workflow Logs.** Austria.

Garcia, R. S.; Ruiz, D. D. (2005) **Pré-Processamento de Dados para Descoberta de Conhecimento em Processos de Workflow Modelados sobre Plataforma Oracle.** I Escola Regional de Banco de Dados – SBC. Porto Alegre.

Garcia, R. S. (2005a) **Descoberta de conhecimento em processos de *Workflow* modelados e executados sobre plataforma *Oracle.*** FACIN-PUCRS (Trabalho de Conclusão de Curso).

Grigori, D.; Casati, F.; Castellanos, M.; Dayal, U.; Sayal, M.; Shan, M. (2004) **Business Process Intelligence: Computer In Industry**. 53:321-243. Elsevier.

Han, J.; Kamber, M. (2001) **Data Mining: Concepts and Techniques.** Morgan Kaufmann.

Hollingsworth, D. (1995) **Workflow Management Coalition: The Workflow Reference Model**. http://www.wfmc.org/standards/docs/tc003v11.pdf

Leymann,F.; Roller,D. (2000) **Production Workflow: Concepts and Techniques.** Prentice-Hall.