Framework para Modelagem de Processos usando Redes de Petri de Alto-Nível e Regras Ativas: um Enfoque sobre a Eliminação de Dados

Wallace A. Pinheiro[1,3], Geraldo Xexéo[1,2], Jano M. de Souza[1,2], Ricardo Barros[1]

[1]Instituto Alberto Luiz Coimbra de Pós-graduação e Pesquisa de Engenharia
Universidade Federal do Rio de Janeiro (UFRJ)

[2]Departamento de Ciência da Computação, Instituto de Matemática
Universidade Federal do Rio de Janeiro (UFRJ)
Cidade Universitária, Centro de Tecnologia, Bloco H, Sala 319, Caixa Postal 68.511 – CEP: 21945-970 – Rio de Janeiro, RJ – Brasil

[3]Seção de Engenharia de Computação – Instituto Militar de Engenharia (IME)
Pr. General Tibúrcio, 80 - CEP: 22290-270 - Rio de Janeiro, RJ – Brasil

{awallace,xexeo,jano, barros}@cos.ufrj.br

Abstract. *The data explosion scenario has been predicted since 1970 by Alvin Toffler in "Future Shock". The information overload considers a polluted informational space that means too much information and low quality or relevance to the required context. The current situation seems to be critical and the uncontrolled increase of information quantity will make this situation worse. To minimize this problem, we propose a framework to model processes based on Petri nets and active rules. The processes, modeled like patterns, exemplify the use of the framework and they aim to eliminate not relevant and unnecessary data. The patterns integrate data flow and data control perspectives in one unique description model.*

Resumo. *O cenário da explosão de dados já era previsto desde 1970 por Alvin Toffler em "Future Shock". A sobrecarga de informação considera um espaço informacional poluído, isto é, contém muita informação de baixa qualidade ou baixa relevância quanto à tarefa que nos propomos. Se a situação atual já nos parece crítica, o aumento crescente e sem controle da quantidade de informação projeta que os problemas desse tipo aumentarão já em curto prazo. Para auxiliar no tratamento desses problemas, este trabalho propõe um framework de modelagem de processos baseados em redes de Petri de alto-nível e regras ativas. Os processos aqui modelados na forma de padrões exemplificam o uso desse framework e são voltados para a eliminação de dados não relevantes e desnecessários. Os padrões propostos integram as perspectivas de fluxos de controle e de dados em um modelo único de descrição.*

1. Introdução

Um recente estudo feito pela SBC [de Carvalho, 2006] revela a preocupação com o crescimento explosivo de dados nos diversos setores empresariais, governamentais e não governamentais. Essas instituições enfrentam uma dicotomia, pois estão sempre buscando informações e, ao mesmo tempo não conseguem analisar a enorme quantidade

de dados disponíveis. Portanto cresce a necessidade de tratar a sobrecarga e a poluição de dados através da redução (abstração e sumarização) das massas de dados por meio de modelagem computacional, simulações e outros.

Dados não podem ser considerados recursos perenes e intocáveis, sendo necessária a sua freqüente reavaliação em busca da manutenção ou melhoria da qualidade. Para isto, é necessário que os mesmos sejam gerenciados durante todo o seu ciclo de vida dentro das organizações, ou seja, desde a sua criação ou obtenção até a sua destruição. Infelizmente, esta não é uma tarefa fácil, pois normalmente existem diversas fontes de dados que fornecem informações variadas, similares e até mesmo conflitantes. Este trabalho propõe o uso de um fluxo combinado de controle e de dados na descrição de padrões de eliminação de dados. Esse fluxo, modelado através de Redes de Petri de Alto-Nível gera uma visão integrada da solução, propiciando um modelo único de descrição.

Este artigo está organizado da seguinte forma: a próxima seção discute os assuntos relacionados ao tema. A seção 3 apresenta as duas perspectivas do problema que são usadas na descrição dos padrões. A seção 4 descreve os padrões de eliminação de dados, enquanto a seção 5 propõe uma arquitetura de suporte à estratégia adotada. Finalmente, a seção 6 apresenta a conclusão e as perspectivas dos trabalhos futuros.

2. Assuntos Relacionados

Nos últimos séculos mudanças radicais aconteceram no contexto das organizações. O conhecimento acumulado de forma cada vez mais acelerada levou a uma nova revolução, que estamos vivendo hoje em dia, a revolução da informação [Cavalcanti, 1995]. A gestão do conhecimento vem ganhando enfoque, pois auxilia a tratar os efeitos destas mudanças. Essa gestão envolve diversas etapas que vão desde a identificação do que é importante a organização saber até o descarte do que é ou se tornou não relevante.

Figura 1 - Classificação da Informação segundo a finalidade para uma Organização (fonte: Moresi, 2001).

Moresi [Moresi, 2001] classifica a informação[1] segundo o papel que ela desempenha nas atividades de uma organização de acordo com a Figura 1. Nesta figura, pode-se destacar a importância da eliminação da informação sem interesse. Apesar do descarte de informação não relevante ser uma importante fase no processo de gestão de informações de uma organização, de modo geral, isto tem sido negligenciado.

Os padrões de eliminação de dados seguem a idéia dos padrões de processos formalizados por diversas iniciativas [Russell, 2006; Russell, 2004; Van der Aalst, 2003]. Entretanto, uma das diferenças é que estas pesquisas descrevem fluxos de controle e de dados de maneira independente, além de não fornecerem uma boa base formal em termos sintáticos e semânticos em função do grande número de paradigmas e conceitos diferentes existentes na área de *workflows* [Van der Aalst, 2005; Van Der Straeten, 2007]. Para superar essas dificuldades, Mulyar [Mulyar, 2005] acredita que a combinação das perspectivas de controle e dados pode ser feita utilizando Redes de Petri Coloridas (RdPC´s), pois, além de oferecerem uma sólida base conceitual, as RdPC´s permitem tratar esses fluxos de forma homogênea, como acontece no mundo real.

Propõe-se neste artigo a utilização de um fluxo combinado de controle e dados na descrição de padrões de eliminação de dados. Entretanto, diferentemente dos padrões propostos por Mulyar [Mulyar, 2005], este trabalho utiliza Redes de Petri de Alto-Nível (RdPAN´s) modelando duas perspectivas distintas: modelagem do fluxo de controle através dos conceitos utilizadas na área de regras ativas e modelagem do fluxo de dados através da simulação dos repositórios, suas interligações e das estruturas de dados.

Regras ativas permitem descrever fluxos de ações e eventos, onde as ações podem ser executadas em função da ocorrência dos eventos. Isto pode ser visto como uma forma de fluxo de controle orientado a eventos. Os processos são, normalmente, iniciados por eventos, e as regras ativas permitem descrever o fluxo de controle associados a esses processos. Nesta área Li [Li, 2004] propôs a *Conditional Colored Petri Net* (CCPN) uma RdPC estendida para a representação e execução de regras ECA (evento-condição-ação). Um dos problemas desse trabalho é que as regras normalmente são criadas em arquivos textos ou ferramentas separadas que não utilizam a notação das RdPC´s para, só então, serem reescritas em RdPC. Isto dificulta o projeto, análise e depuração de regras. As RdPAN´s [Trompedeller, 1995] utilizadas neste trabalho englobam as características das redes de Petri Coloridas, das redes de Petri Temporais e das redes de Petri Hierárquicas.

3. As Duas Perspectivas do Problema

Dependendo da perspectiva, fluxo de controle ou de dados, os lugares, transições e marcas das RdPAN´s podem assumir diferentes significados, como visto a seguir:

1) Modelagem do Fluxo de Dados - associam-se os seguintes significados aos elementos da rede de Petri:

[1] Neste trabalho, os termos dado e informação são usados indistintamente, embora alguns autores considerem dado como a representação de aspectos, fatos ou fenômenos, enquanto informação é o dado com significado.

- Lugar – corresponde a um local de armazenamento temporário ou persistente dos dados. Cada lugar suporta uma estrutura de dados no formato da linguagem CPN ML. O tipo de estrutura de dados da marca deve ser escolhido de acordo com o dado a ser mapeado. Por exemplo, o tipo *DATA* e variações (ex.: *DATA_TIME*);
- Transição – representa a uma regra ou conjunto de regras;
- Marca – corresponde a uma instância do dado definidos nos lugares. Cada marca tem o seu tipo e estrutura definido no lugar que a contém.

2) Modelagem de Fluxo de Eventos - nesta perspectiva lugares, transições e marcas modelam regras do tipo ECA. São associados os seguintes significados aos elementos da rede de Petri:

- Lugar – corresponde a um padrão de evento ou ação. Cada evento ou ação corresponde a um tipo especificado na linguagem CPN ML. Em regras ECA, um evento pode gerar uma ação. Neste caso, a ação de uma regra pode ser correspondente a um evento em outra regra. Os lugares permitem os tipos de eventos *PRIMITIVE* e *COMPOSITE* correspondendo respectivamente a eventos primitivos sinalizados por bases de dados e aplicações, e eventos compostos formados pela composição dos primitivos.
- Transição – representa a condição para a ocorrência de uma ação ou evento dependente;
- Marca – representa a instância (ocorrência) de um evento ou de uma ação de um padrão definido no lugar que a contém.

4. Padrões de Eliminação de Dados

Neste trabalho, os padrões são modelados por RdPAN's, sendo que a estrutura documental dos padrões usa uma adaptação simplificada do formato GOF, proposto por Gamma [1994]. A RdPAN´s permitem modelar formalmente sistemas e fluxos de informações entre seus diversos componentes, sendo apropriadas para descrição desses padrões. Na descrição dos padrões em RdPAN´s são usadas variáveis e funções em CPN ML, que é uma implementação do padrão ML [Harper, 2005]. São exemplos as funções *read*, get*, put**, entre outras.

A maioria dos padrões apresenta duas fases: a primeira simplesmente detecta a existência ou não do dado de interesse, enquanto a segunda efetivamente destrói o dado. A idéia é que na primeira fase, todos os dados podem ser acessados por qualquer aplicação, enquanto na segunda fase pode existir um bloqueio dos dados a serem apagados, não sendo mais possível acessá-los. Neste processo, a intervenção humana não é necessária, sendo que os padrões ou monitoram os dados continuamente ou são acionados de tempos em tempos, de forma automática, realizando a checagem dos dados.

Os padrões foram classificados de acordo com o critério finalidade. A finalidade reflete o que o padrão faz e se divide em duas categorias: prevenção e correção. A categoria prevenção descreve padrões com o objetivo de prevenir a entrada de dados não desejados, eliminando-os antes de serem incorporados ao repositório interno da

aplicação ou da organização. A categoria correção descreve os padrões que têm como objetivo eliminar dados não desejados em um repositório interno. Até o momento foram classificados segundo este critério oito padrões apresentados na Tabela 1. Por restrições de espaço somente dois padrões serão completamente descritos nas seções seguintes.

Tabela 1: Classificação dos padrões Data Killing pelo critério de finalidade

Finalidade	**Prevenção**	Allowed Data	Considera o conteúdo dos dados para permitir ou não a entrada de dados externos nos repositórios internos das organizações.
		Blocking Prohibited Data	Previne a entrada de dados considerados proibidos (não desejados) nos repositórios internos de organizações, levando em consideração o conteúdo dos dados.
		Blocking Similar Data	Previne a entrada de dados iguais ou similares aos já existentes nos repositórios internos de organizações.
		Sampled Data	Amostra dados externos de acordo com períodos de tempo pré-definidos. Os dados não amostrados são considerados descartes.
	Correção	Discarding Similar Data	Deixa apenas uma versão de um dado, eliminando dados iguais ou similares nos repositórios internos de organizações.
		Discarding Obsolete Data	Elimina os dados que se tornaram obsoletos, em virtude da passagem do tempo, dos repositórios internos de organizações.
		Discarding Prohibited Data	Elimina os dados proibidos (não desejados) dos repositórios internos de organizações, levando em consideração o conteúdo dos dados.
		Discarding Useless Data	Baseando-se no critério de taxa de uso do dado, elimina os dados que apresentam baixa taxa de utilização dos repositórios de organizações.

É importante considerar que a perda de dados relevantes não é aceitável em um ambiente organizacional. Desta forma, todos os padrões propostos permitem a recuperação dos dados, deste que os dados eliminados sejam mantidos num repositório secundário (lixeira) antes de sua exclusão definitiva. A exclusão definitiva dos dados depende das regras de cada organização. Por exemplo, no Brasil os registros fiscais devem ser mantidos por 5 anos de acordo com lei. Estes registros, no entanto, não precisam sobrecarregar os sistemas transacionais dessas empresas, podendo ser transferidos para repositórios secundários com a ajuda dos padrões propostos.

4.1 Discarding Similar Data

Descrição (Exemplos e Motivação): os sistemas de uma organização podem possuir fontes de dados diferentes. Quando isto ocorre, se não for feita uma verificação prévia, muitos dados similares ou duplicados são armazenados nos repositórios internos dessas organizações. Como visto anteriormente, um dos maiores problemas é identificar os dados que representem o mesmo objeto no mundo real, apesar de apresentar pequenas variações. Algumas vezes estes dados não podem ser descartados, por causa da arquitetura dos dados ou de regras de negócio da empresa. Nos casos em que o descarte pode ocorrer, sua motivação está associada a diversos fatores, entre eles: facilitar a manutenção da consistência da base, evitar redundâncias, não permitir o aumento da base com dados duplicados ou diminuir o tempo de processamento de consultas.

Aplicabilidade: quando se deseja a eliminação de dados similares ou duplicados em bases de dados. Esses dados podem ser eliminados ou armazenados em um repositório de descarte. A eliminação das duplicações facilita a manutenção da consistência da

base, pode aumentar a velocidade de processamento dos dados e diminuir o tempo de processamento de consultas.

Solução Abstrata: a rede de Petri da Figura 2 descreve o *internal repository* onde existem dados similares ou duplicados, e o repositório para descarte de dados (*trash*). Há duas marcas no repositório interno que versam sobre o mesmo tópico: Petri *Nets* ou PN com o mesmo ISBN. A Figura 3 detalha o processo de seleção dos dados. A primeira regra ECA (*R1*) verifica se um novo dado inserido representa um dado já existente na base. Se existir algum dado similar (detectados pela expressão de guarda de *R1* com a função *detectedSimilarData(...))*, então a segunda regra (*R2*) pode ser disparada. Nesta regra, o dado inserido é eliminado para a lixeira através do arco que contém a função *putSimilarData(...)*.

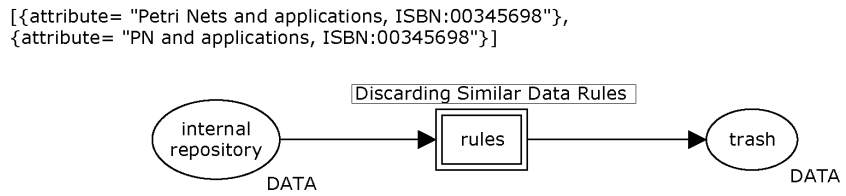

Figura 2 - Modelagem do fluxo de dados

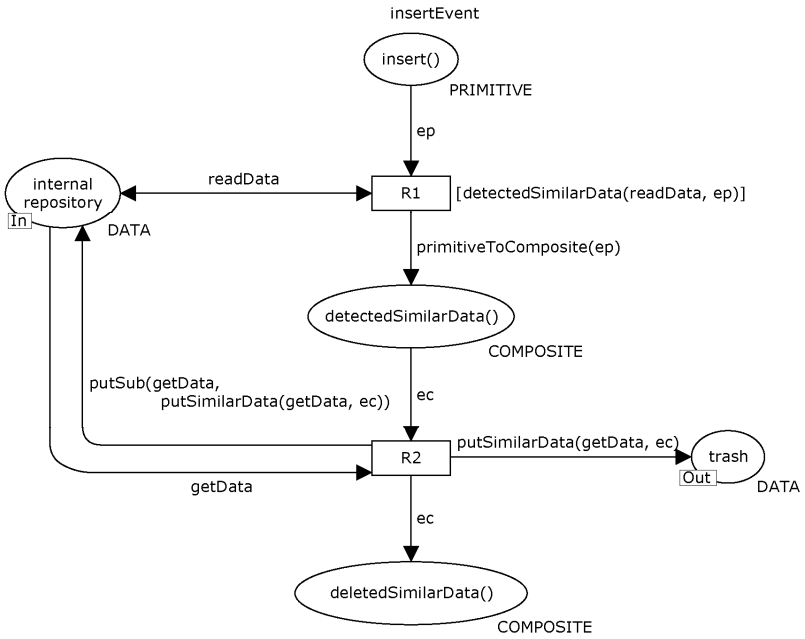

Figura 3 - Modelagem das Regras

4.2 Discarding Obsolete Data

Descrição (Exemplos e Motivação): alguns dados perdem a sua importância com o passar do tempo e das regras de negócio da empresa. Estes dados sobrecarregam sistemas se não forem eliminados dos repositórios principais. Esses dados podem ser armazenados em repositórios secundários, tais como os repositórios de dados históricos de *data warehouses*, ou ser simplesmente eliminados. São exemplos desses dados os referentes a registros financeiros para fins legais e de auditoria.

Aplicabilidade: dados que não tenham mais valor em função das regras de negócio de uma organização em relação ao tempo podem ser considerados obsoletos. Os dados obsoletos podem ser descartados ou armazenados em um repositório de armazenamento temporário. Podem ser usadas regras adicionais que estabeleçam os critérios de tempo para identificação dos dados obsoletos por determinada área ou assunto.

Solução Abstrata: a rede de Petri da Figura 4 descreve o *internal repository* onde podem existir dados obsoletos e o repositório para descarte de dados (*trash*). Há quatro marcas no repositório interno versando sobre assuntos diversos e com quatro diferentes datas de criação. A Figura 5 detalha o processo de seleção dos dados. Duas regras são utilizadas neste processo. A primeira regra (*R1*) verifica existência de dados obsoletos considerando o tempo de disparo do evento periódico e o período de validade do dado. Este padrão considera a existência de eventos periódicos que disparam a primeira regra, ou seja, de tempos em tempos os dados de entrada serão lidos em busca de dados obsoletos. Se existirem dados obsoletos (detectados pela expressão de guarda de *R1* com a função *detectedOldData(...))*, então a segunda regra (*R2*) pode ser disparar. Nesta regra, os dados obsoletos são enviados para a lixeira através do arco que contém a função *putOldData(...)*.

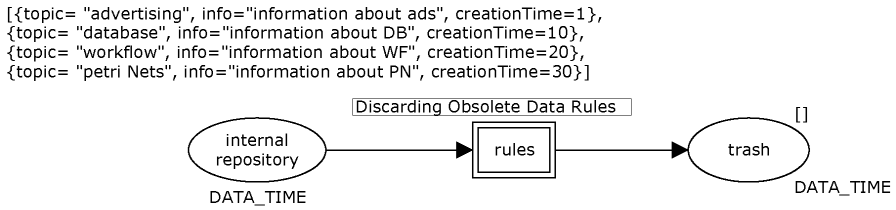

Figura 4 - Modelagem do fluxo de dados

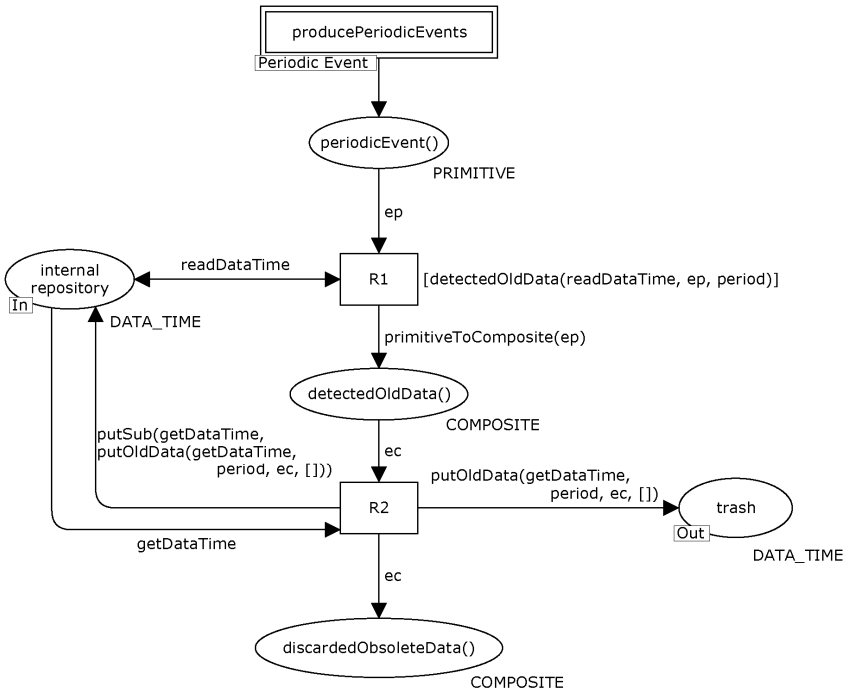

Figura 5 - Modelagem das Regras

5. Arquitetura Proposta

Nesta seção é apresentada a arquitetura proposta para suportar a modelagem e execução dos processos de eliminação de dados. A figura 6 apresenta essa arquitetura. Na camada superior da arquitetura tem-se a ferramenta CPN Tools [Ratzer, 2003] com suas duas interfaces, a CPN Tools Interface e a Britney Suite, além do simulador de RdPAN. Esta ferramenta permite que regras ativas e fluxo de dados utilizando RdPAN sejam especificados. Estas regras são armazenadas em repositórios de regras e podem ser utilizadas sempre que necessário.

A camada intermediária da arquitetura inclui os dispositivos de comunicação. CPN ML permite a comunicação com outras linguagens, dessa forma, os elementos dessa camada são implementados utilizando Java. Este camada inclui três elementos:

- Leitor de Eventos – responsável por receber os eventos advindos das bases de dados e aplicações, e depositar as marcas nos lugares dos eventos detectados;
- Leitor de Dados – responsável por ler os dados relevantes para as regras e alocá-los nos respectivos lugares;
- Gerador de Ações – responsável por traduzir as ações tomadas pelas regras no simulador para ações das bases de dados e aplicações. Estas ações incluem alteração nos dados e sinalização de novos eventos.

A camada inferior possui as interfaces e *plugins* para bases de dados e aplicações externas, incluindo *Web services*. A arquitetura prevê a utilização de servidores *Web* fornecendo serviços baseado em padrões de regras definidos nas RdPAN´s. Isto permite que os serviços de eliminação de dados sejam oferecidos via *Web* para aplicações que tenham interesse. Além disso, a arquitetura é expansível, prevendo que outros *plugins* possam ser criados futuramente.

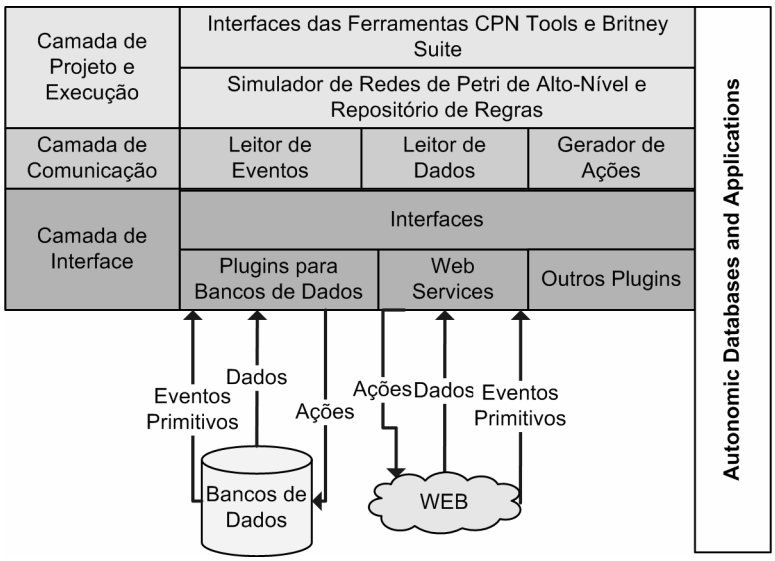

Figura 6 - Arquitetura da solução proposta.

6. Experimento: Aplicação do padrão Discarding Similar Data

Para ilustrar a aplicação do padrão *Discarding Similar Data*, consideramos um experimento sobre um repositório de dados sobre hotéis obtidos de dois sites de pesquisa: *hoteltravel.com* e *travel.hotels-and-discounts.com*. Normalmente existe a ocorrência de coincidências no hotéis retornados por esses sites. Para fins de simplificação, o repositório corresponde a um banco de dados relacional contendo somente uma tabela que agrega todos os resultados obtidos dos sites. As Tabelas 2 e 3 apresentam os 10 primeiros resultados retornados por cada um desses sites segundo uma busca pelos hotéis de melhor qualidade da cidade do Rio de Janeiro.

Tabela 2: Lista de hotéis retornados pelo site hoteltravel.com

Ordem	Hotel	Diária (USD)
1	**Caesar Park**	**347**
2	Copacabana Palace	408
3	InterContinental Rio	140
4	Othon Palace	152
5	**Sheraton Barra**	**197**
6	**Sheraton Rio**	**185**
7	Sofitel Rio de Janeiro	268
8	Augustos Copacabana	93
9	California Othon	97
10	Copacabana Mar	102

Tabela 3: Lista de hotéis retornados pelo site travel.hotels-and-discounts.com

Ordem	Hotel	Diária (USD)
1	Ipanema Plaza	240
2	**Sheraton Rio**	**161**
3	Rio Othon	170
4	Luxor Regente	123
5	Windsor Palace	108
6	Orla Copacabana	119
7	**Caesar Park Ipanema**	**360**
8	Porto Bay Rio Internacional	167
9	**Sheraton Barra**	**230**
10	Lancaster Othon Travel	91

As linhas 1, 5, 6 da Tabela 2 e as linhas 2, 7 e 9 da Tabela 3 apresentam dados de hotéis idênticos retornados pelos sites. A verificação de similaridade neste caso é baseada no nome do hotel com um nível de similaridade mínimo de 65%. A política de

eliminação de dados utilizada neste caso elimina os dados similares com maior valor de diária. Isto ocorre, pois os sites oferecem descontos diferenciados, apesar de frequentemente endereçarem os mesmos hotéis. Primeiramente no repositório são inseridos os dados da Tabela 2 e depois os dados da Tabela 3. O padrão *Discarding Similar Data* irá eliminar os dados considerados similares a cada inclusão de uma tupla de dados no repositório. Para checar a similaridade entre os nomes dos hotéis são propostas as funções de similaridade apresentadas na tabela 4.

Tabela 4: Funções de Similaridade

Função *simName*	Função *simFullName*
$simName(x,y) = \begin{cases} 1, & if\ x_t = y_t,\ \forall t \in [0, \max(m,n)] \\ 0, & in\ other\ case. \end{cases}$	$simFullName(nameA, nameB, simLevel) = \begin{cases} true, & if\ \dfrac{\sum_{q=1}^{\max(s,t)} simName(nameA_q, nameB_q)}{\max(s,t)} > simLevel \\ false, & in\ other\ case. \end{cases}$

A função *simName* recebe como parâmetro um par de palavras *x*, *y*, onde x_i é a i-ésima letra da palavra *x*, y_j é a j-ésima letra da palavra *y*, *m* é o tamanho da palavra *x*, *n* é o tamanho da palavra *y*. A função *simFullName* recebe como parâmetros um par de nomes completos *nameA*, *nameB* e o nível de similaridade desejado *simLevel*. Cada nome completo é formado por um conjunto de nomes separados por um caracter de espaço. Portanto, $nameA_q$ é o q-ésimo nome contido no nome completo *nameA* e $nameB_q$ é o q-ésimo nome contido no nome completo *nameB*. Além dessas funções, o padrão utiliza duas funções recursivas que percorrem todos os dados já inseridos, procurando por similaridades. Estas funções são apresentadas na tabela 5.

Tabela 5: Funções que percorrem os dados em busca de similaridades

Função *detectedSimilarData*	Função *putSimilarData*
detectedSimilarData(readTuples, event) = *if simFullName* *(name(new(readTuples)), name(last(readTuples)),* *simLevel(event))* *then true* *else* *if (length(readTuples) > 0)* *then detectedSimilarData* *(diff (readTuples, last(readTuples)), simLevel(event))* *else false*	*putSimilar Data (readTuples , event) =* *if simFullNam e* *(name (new(readTuples)), name (last (readTuples)),* *simLevel (event))* *then* *if price(new(readTuples)) >= price(last(readTuples))* *then new(readTuples)* *else last (readTuples)* *else* *if (length (readTuples) > 0)* *then putSimilar Data* *(diff (readTuples , last (readTuples)), simLevel (event))* *else null*

A função *detectedSimilarData* verifica se a nova tupla inserida é similar a alguma das tuplas já existentes. A função *putSimilarData* escolha a tupla similar que será descartada pelo critério do maior valor da diária. Nestas funções, a lista de tuplas existentes é dada por *readtuplas*, a nova tupla inserida corresponde a *new(readTuples)*, a última tupla da lista é dada por *last(readTuples)*, enquanto o nível de similaridade é

dado por *simLevel(event)*. Dessa forma, o nível de similaridade é configurado como um atributo do evento que dispara o padrão. O valor da diária é obtido pela função *price*.

Considerando que o nível de similaridade estabelecido para a detecção de dados idênticos foi de 65%, então os hotéis "Caesar Park Ipanema" e "Caesar Park" serão considerados idênticos, pois seu nível de similaridade de acordo com as funções propostas é de 66% (2/3 do primeiro nome encontra equivalente no segundo nome). Os outros hotéis destacados como prováveis similares obtiveram o nível de similaridade igual a 100%. Finalmente, depois de incluir os dados obtidos dos sites através da aplicação do padrão *Discarding Similar Data*, obtém-se como resultado a eliminação de três dados considerados similares, restando no repositório os valores apresentados na Tabela 6.

Tabela 6: Dados mantidos no repositório

Ordem	Hotel	Diária (USD)
1	Caesar Park	347
2	Copacabana Palace	408
3	InterContinental Rio	140
4	Othon Palace	152
5	Sheraton Barra	197
6	Sofitel Rio de Janeiro	268
7	Augustos Copacabana	93
8	California Othon	97
9	Copacabana Mar	102
10	Ipanema Plaza	240
11	Sheraton Rio	161
12	Rio Othon	170
13	Luxor Regente	123
14	Windsor Palace	108
15	Orla Copacabana	119
16	Porto Bay Rio Internacional	167
17	Lancaster Othon Travel	91

7. Conclusão

Uma das principais contribuições deste trabalho é o desenvolvimento de uma estratégia de modelagem de processos utilizando redes de Petri e regras ativas. Para isso, foi desenvolvido um *framework* que suporta essa modelagem. Porém, nós acreditamos que o *framework* proposto não fica limitado ao desenvolvimento de padrões para eliminação de dados, podendo ser generalizado para a modelagem de diversos padrões de processos. Entretanto, neste trabalho o foco está sobre o desenvolvimento de processos voltados para o descarte de dados. Este desenvolvimento permite identificar processos já existentes e que, em virtude da sua excelência, possam ser reutilizados. No caso dos

padrões para eliminação de dados, o objetivo é identificar os padrões de descarte existentes, classificá-los, descrevê-los e correlacioná-los com os possíveis casos de uso. Uma vez que esses padrões existam, diversas aplicações poderão se beneficiar do seu uso, propiciando uma melhor manutenção de suas bases de dados. Os próximos passos deste trabalho objetivam, além da descoberta de novos padrões de processos, a criação de serviços que permitam que outras aplicações deleguem atividades via *web* que serão mediadas pelo *framework* e executadas remotamente.

Referências

Cavalcanti, E. P. (1995) Revolução da Informação: Algumas Reflexões. Caderno de Pesquisa em Administração, Vol. 1, São Paulo, Brasil.

de Carvalho, A. C. P. L. F., et al. (2006) Grandes Desafios da Pesquisa em Computação no Brasil – 2006 – 2016. Relatório sobre o Seminário realizado em 8 e 9 de maio de 2006, SBC.

Gamma, E., et al. (1994) Design Patterns: Elements of Reusable Object-Oriented Software, Pearson. ISBN: 0201633612.

Harper, R. (2005) Programming in Standard ML., Vol. Draft, Carnegie Mellon University.

Li, X., et al. (2004) An application of conditional colored Petri nets: active database system. Man and Cybernetics, IEEE International Conference on Systems. Seccion de Computacion, CINVESTAV-IPN, Mexico City, Mexico.

Moresi, E. A. D., Tarapanoff, and K. (2001) Inteligência Organizacional e Competitiva. Capítulo: Gestão da Informação e do Conhecimento, Brasília, Brasil, UnB.

Mulyar, N. A. and Van der Aalst, W. M. P. (2005) Patterns in Colored Petri Nets. Sixth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, University of Aarhus, Denmark, Department of Computer Science.

Ratzer, A. V., et al. (2003) CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets. Proceedings of the 24th ICATPN, Eindhoven, The Netherlands, Spring.

Russell, N., et al. (2006) Workflow Control-Flow Patterns: A Revised View. BPM Center Report BPM-06-22, BPMcenter.org.

Russell, N., et al. (2004) Workflow Data Patterns. QUT Technical report, Queensland University of Technology, Brisbane, FIT-TR-2004-01.

Trompedeller, M. A. (1995) Classification of Petri Nets. Disponível em: http://www.informatik.uni-hamburg.de/TGI/PetriNets/classification/

Van der Aalst, W. M. P. and ter Hofstede, A. H. M. (2005) YAWL: Yet Another Workflow Language. Information Systems, Vol. 30(4), pp. 245--275.

Van der Aalst, W. M. P., et al. (2003) Workflow Patterns. Distributed and Parallel Databases, Vol. 14(3), pp. 5--51.

Van Der Straeten, R., et. al. (2007) Aspect-Oriented Support for Workflow Languages. Thesis and/or Apprenticeship Proposals 2006-2007, Vrije Universiteit Brussel, Brussels, Belgium.