3
Eventos: definição e processamento

3.1
Definição de eventos

Antes de tratar o processamento de eventos, é preciso relembrar a definição de um evento, apresentada na seção 2.2. A definição mais simples de evento é a ocorrência de uma atividade 12. No contexto de sistemas de computação, entende-se por evento a representação de uma atividade que ocorreu em um sistema de computação.

Magid 25 faz uma definição mais completa de um evento qualificando-o como uma ocorrência atômica (que acontece inteiramente, ou não acontece), instantânea (que acontece em um ponto específico do tempo) e que tem significado em seu domínio de interesse. Magid diferencia também eventos *concretos* de eventos *inferidos*, ou *complexos*. Desse modo, eventos concretos seriam a manifestação concreta de uma atividade em um sistema. Enquanto que eventos inferidos, ou complexos, seriam eventos criados a partir da avaliação de um conjunto de eventos e condições.

A denominação eventos complexos, apresentada por Magid, parece não ser a mais adequada, tendo em vista que o evento em si continua estruturalmente simples, ou seja, permanece sendo atômico e instantâneo, apenas contendo uma informação mais elaborada. Nesta dissertação será utilizada a denominação de evento sintético, em lugar de evento complexo, ou seja, defini-se como evento sintético o evento criado a partir da avaliação de um conjunto de eventos e condições que representam um padrão. Esta definição de eventos sintéticos e eventos concretos é importante pois será utilizada nas seções a seguir.

Outro conceito importante é a noção de que eventos sintéticos devem ser entendidos como eventos hierarquicamente superiores, ou seja, eventos que foram sintetizados a partir de um conjunto de outros eventos, possivelmente representando um padrão de comportamento.

No escopo deste trabalho, a representação da atividade de um sistema é materializada em linhas de texto em arquivos de log, caracterizando um evento

concreto de log. A apresentação e discussão sobre conceitos e técnicas sobre como processar e correlacionar estes eventos são o ponto central deste capítulo.

3.2
Processamento de eventos

O tema abordado nesta seção é bastante amplo, percorrendo diferentes áreas de pesquisas. A intenção desta seção é mostrar as principais linhas de pesquisa relacionadas ao processamento de eventos e chegar a uma visão única dos conceitos.

A primeira linha de pesquisa tem origem na correlação de alarmes em redes de comunicação e explicita as operações de correlação que poderiam ser feitas. Nesta linha, Jakobson e Weissman [28, 29] publicaram artigos importantes sobre correlação de alarmes.

Enxergando o alarme como um evento, pode-se extrapolar a definição de correlação de alarmes para a definição de correlação de eventos como sendo a interpretação conceitual de múltiplos eventos, que leva à atribuição de um novo significado aos eventos originais, ou seja, produz um evento sintético. A correlação aumenta o nível de informação dos eventos, ou seja, cria eventos hierarquicamente superiores.

Jakobson e Weissman também definiram operações de correlação de eventos:

- Compressão: reduz múltiplas ocorrências de eventos idênticos em um único evento.
- Filtragem: suprime um evento se um de seus parâmetros tem um determinado valor.
- Supressão: suprime um evento se certo contexto operacional está presente.
- Contagem: consiste em contar o número de ocorrências de um determinado tipo de evento.
- Escalonamento: se um contexto operacional está presente, assinala um valor maior a um parâmetro.
- Generalização: substitui um evento por uma de suas superclasses.
- Especialização: substitui um evento por uma de suas subclasses.
- Relação temporal: correlaciona eventos dependentemente de sua ordem e tempo de chegada.

- Agrupamento: aplica um padrão de correlação complexo, onde os componentes, operações de correlação, eventos e testes externos são pré-definidos.

Ainda sobre o tópico operações de correlação de eventos, Magid 25 descreve cinco padrões para correlação de evento:

- Junção: detecta uma situação onde todos os eventos requeridos estão presentes, estejam estes eventos em uma ordem pré-definida, ou qualquer ordenação.
- Contagem: detecta uma situação aonde ao menos, no máximo ou exatamente *n* eventos chegaram.
- Temporal: detecta uma situação em um tempo absoluto, em um intervalo definido ou um evento após um intervalo determinado ou tempo absoluto.
- Agregação: detecta uma situação dependente de uma condição de agregação de dados sobre eventos (mínimo, máximo, média, soma e percentagem).
- Ausência: detecta uma situação onde certos eventos não aconteceram.

A segunda linha de pesquisa tem origem em 1998 em um relatório técnico da Universidade de Stanford 69 que cunhou o termo CEP, *Complex Event Processing*, ou o Processamento Complexo de Eventos. Esta linha ganhou força nos últimos dois anos. No mercado grandes empresas estão investindo em soluções de CEP para monitoramento de processos de negócio e para o mercado financeiro. No meio acadêmico este tema também ganhou mais apelo, o que pode ser evidenciado, por exemplo, pela criação de um congresso específico de sistemas distribuídos de eventos onde o termo CEP é bastante utilizado.

Nesta linha de pesquisa, o *processamento de eventos* é definido em 12 como uma computação que realiza operações em eventos, como leitura, criação, transformação, exclusão, entre outras. As referências [5, 12, 27] distinguem três tipos de processamento de eventos, separados pela complexidade de processamento.

O primeiro tipo é o Processamento Simples de Evento (*Simple Event Processing*): neste tipo de processamento um evento simples dispara uma ou mais ações. Já o segundo tipo é o Processamento Complexo de Evento (*Complex Event Processing* - CEP): neste tipo de processamento um conjunto

amplo de eventos é avaliado para a tomada de ação. Os eventos podem ser de diferentes tipos e ocorrer sobre um período longo de tempo. A correlação entre os eventos pode estar relacionada a um contexto causal, temporal ou espacial. Para realizar este processamento é preciso de interpretadores de evento, definições e reconhecimento de padrões de evento, e técnicas de correlação. Quando estes eventos estão ordenados em um fluxo contínuo, tem-se o terceiro tipo, que é na verdade um caso especial do segundo.

Uma terceira linha de pesquisa sobre processamento de fluxos de eventos é representada pelos chamados *Data Stream Management Systems*, sistemas capazes de processar fluxos de eventos através de consultas contínuas. Esta terceira linha de pesquisa tem origem da área de banco de dados e é muito citada na literatura relacionada ao CEP, sendo um importante tipo de processador complexo de eventos. Os processadores baseados em consulta serão apresentados com mais detalhes na seção 3.4.2.

Considerando as abordagens de Magid, Jakobson e Weissman sobre correlação de eventos, e contrapondo o conceito de processamento complexo de eventos, pode-se notar que apenas o contexto espacial não é explicitamente abordado nas operações de correlação de eventos. No entanto, a condição espacial poderia ser tratada como um atributo do evento. Os demais contextos, causal e temporal, são explicitamente tratados pelas operações de correlação de evento.

Desta forma, o conceito de processamento complexo de eventos está intimamente ligado com o que é denominado por correlação de eventos. Não existe hoje um entendimento único sobre as diferenças entre os conceitos de correlação de eventos e processamento complexo de eventos, ou seja, não foram encontrados na literatura artigos que procurassem entender e contrapor estas linhas de pesquisa com a finalidade de uniformizar os termos e conceitos.

Para se tentar determinar a diferença entre correlação de eventos e processamento complexo de eventos, é necessário contrapor as definições apresentadas para os dois termos. Da maneira como foi definida a correlação de eventos nesta dissertação, tendo como base a definição de correlação de alarmes de Jakobson e Weissman [28, 29], pode-se dizer que os conceitos são equivalentes e serão utilizados sem diferenciação nesta dissertação.

Com relação à diferença entre processamento complexo de eventos e processamento de fluxos de eventos, pode-se dizer que a principal diferença está no objeto analisado. Enquanto, o segundo limita-se a processar fluxos, ou seja, uma seqüência ordenada, o processamento complexo é capaz de

processar *nuvens* de eventos 12, que incluem fluxos de eventos, mas também incluem eventos parcialmente ordenados. Por exemplo, uma seqüência de ordens de compra e venda de uma ação específica seria um fluxo de eventos, enquanto que o conjunto de ordens de compra e venda de todas as ações em um dia seria visto como uma nuvem de eventos.

O diagrama da Figura 3 ilustra os conceitos apresentados sobre evento concreto, eventos sintéticos e processamento complexo de eventos, ou correlação de eventos.

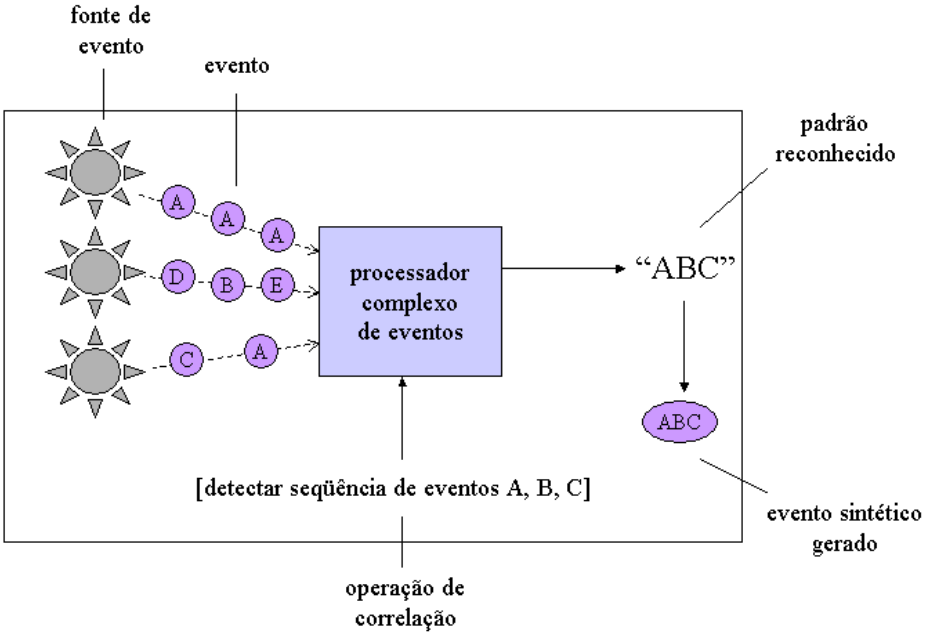

Figura 3 – Diagrama ilustrando os conceitos relacionados a eventos

3.3 Cenários de uso para o Processamento Complexo de Eventos

Os cenários para o uso de processamento complexo de eventos vão desde a monitoração de processos de negócio 67, atividade conhecida como *Business Activity Monitoring*, a sistemas aplicados a monitoração de pacientes neonatais 68 e detecção de defeitos em redes de computadores.

Esta dissertação explora o cenário do processamento complexo de eventos aplicado à correlação de eventos de log em empresas de *Internet*. Estão presentes neste cenário: um grande volume de eventos, uma necessidade de resposta em tempo real e motivações técnicas e de negócio.

3.4
Processadores complexos de eventos

Existe hoje um grande número de ferramentas capazes de realizar o processamento complexo de eventos. Estas ferramentas se dividem basicamente em dois tipos, as que são *baseadas em regras* e as que são *baseadas em consultas* do tipo SQL.

Outros processadores complexos de eventos baseados em técnicas um pouco mais avançadas, como processadores baseados em redes neurais, em algoritmos bayesianos, ou em *hidden Markov models*, costumam ser desenvolvidos para um propósito específico, não havendo ferramentas genéricas.

3.4.1
Processadores baseados em regras

As ferramentas baseadas em regras possuem um mecanismo de comparação que confronta os eventos presentes com um conjunto de regras. Se o conjunto de eventos presentes representar um padrão expresso por uma regra, ou uma combinação de regras, o mecanismo de comparação deverá perceber esta situação e tomar uma ação, seja a criação de um evento sintético, uma ação efetiva em um sistema.

Cada sistema baseado em regras possui sua própria sintaxe para expressar regras. A semântica das regras está baseada nas operações de correlação citadas na seção 3.2. Um exemplo de regra, usando linguagem natural, no contexto de identificação de um problema de conexão com banco de dados que esteja causando erros para o usuário final, seria: “Identifique se um ou mais eventos de *conexão com banco de dados indisponível* ocorreram, seguidos de eventos de *erro HTTP de código 500*”.

É desejável que o sistema de regras tenha uma interface gráfica para criação e edição de regras. Isto permite que o usuário foque na construção lógica das regras, e não tenha que se preocupar com a sua sintaxe.

Outro requisito funcional importante é a possibilidade de criar, excluir ou editar uma regra em tempo de execução. Desta forma, o mecanismo de comparação não teria que ser parado e reinicializado.

Espera-se também que estes sistemas tenham um tempo de resposta baixo, ou seja, o mecanismo de comparação não pode consumir um grande espaço de tempo entre a realização e a detecção de uma situação de interesse.

Isto é importante, pois as ações derivadas de uma detecção podem não fazer mais sentido depois de um intervalo longo de tempo.

Entre os sistemas baseados em regras destacam-se tanto ferramentas proprietárias quanto ferramentas de código aberto. Entre as ferramentas proprietárias, o IBM Active Middleware Technology™ 43 da IBM atende aos requisitos levantados anteriormente. Seu funcionamento é explicado detalhadamente em 25.

Como ferramenta de código aberto, o *Simple Event Correlator* (SEC) 30 31 destaca-se por sua simplicidade e eficiência. Embora não tenha uma interface gráfica para criação e edição de regras e não permita realizar isto em tempo de execução, esta ferramenta possui um conjunto completo de tipos de regras e excelente desempenho. O SEC foi utilizado no protótipo desenvolvido nesta dissertação e será detalhado no Capítulo 5.

3.4.2
Processadores baseado em consultas

Dentre os sistemas capazes de realizar o Processamento Complexo de Eventos existe uma classe de sistemas conhecidos como *Data Stream Management Systems* (DSMS) ou Sistemas de Gerenciamento de Fluxos de Dados. No contexto do processamento de eventos de log, os fluxos de dados a serem gerenciados pelo DSMS correspondem a seqüências de eventos gerados por uma aplicação.

Os Sistemas de Gerenciamento de Fluxos de Dados estão ligados a uma ampla área de conhecimento e pesquisa. Esta dissertação procura resumir os principais conceitos relacionados a estes tipos de sistema.

Os DSMS´s são sistemas que realizam o processamento de dados que não são modelados como relações persistentes, mas como fluxos contínuos e transientes. O modelo tradicional de banco de dados se mostrou ineficiente para estas aplicações e, junto ao crescente interesse neste tipo de sistema, acarretou inúmeras publicações sobre o tema. Entre essas publicações destacam-se 33 e 34, onde é feito um estudo sobre outros trabalhos já realizados e apresentadas as principais questões.

Um fluxo de dados 34 é definido como uma seqüência de itens em tempo real, contínua e ordenada, seja implicitamente pela ordem de chegada ou explicitamente por marcações de tempo. As principais diferenças deste tipo de dados para o modelo tradicional de dados são 33:

- Os dados chegam continuamente;
- O sistema não tem controle sobre a chegada dos elementos de dados, seja em um único fluxo de dados, ou entre fluxos de dados;
- Fluxos de dados podem ter tamanhos infinitos, sendo inviável seu armazenamento completo;
- Após o processamento de um elemento de um fluxo de dados, este deve ser descartado ou arquivado.

Estas diferenças sobre os modelos de dados, ilustradas na Figura 4, implicam em diferenças sobre o modelo de consultas. Em [33, 34], é citado o modelo contínuo de consultas (*continuos queries)*, que avalia continuamente um fluxo de dados, ao contrário do modelo de consulta tradicional, que atua sobre um conjunto fixo de dados e retorna uma resposta.

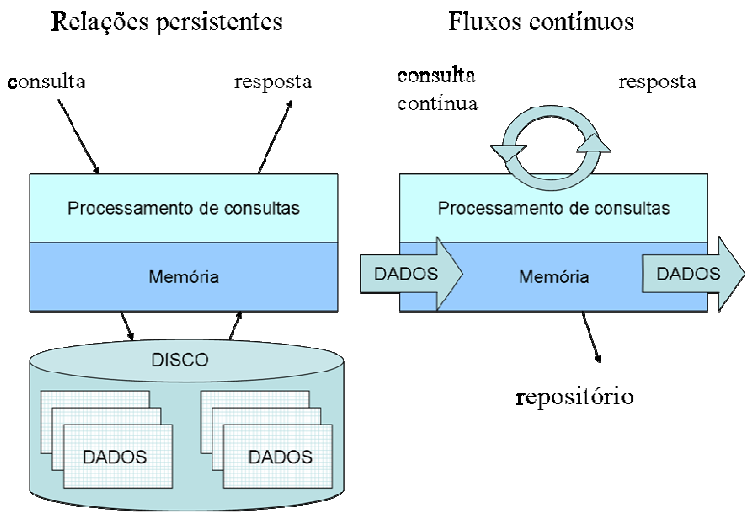

Figura 4- Comparação entre modelos

Este novo modelo de consulta contínua sobre fluxo de dados possui desafios a serem resolvidos. Em 33, são listados aqueles que os autores consideram serem os mais interessantes. São eles:

- Requisitos de Memória ilimitada: como fluxos de dados podem ter tamanho ilimitado, processar todo o fluxo de dados exigiria uma memória infinita. Da mesma forma, quando a taxa de chegada de novos elementos é maior que a velocidade do processamento das consultas o tamanho da fila de entrada irá crescer infinitamente, requerendo uma memória infinita.

- Resposta aproximada de consulta: como não é possível se ter memória infinita, deve ser possível produzir uma resposta aproximada satisfatória ao atuar sobre um subconjunto dos elementos de dados, ou seja, uma parte do fluxo de dados.
- Janelas deslizantes: é uma forma de reduzir o intervalo de tempo sobre o qual a consulta é executada, ou seja, ao invés de atuar sobre todo o passado, a consulta deve atuar sobre um período limitado de tempo.
- Processamento em lotes, amostragem e resumos (*synopses*): o processamento em lotes resolve o problema de uma baixa velocidade de consulta, fazendo com que ela não seja executada continuamente, mas sim periodicamente; a amostragem realiza a consulta apenas sobre parte dos elementos de dados; e os resumos modificam a estrutura dos dados criando um fluxo resumido que pode ser processado sem problemas.
- Operadores bloqueadores: é o caso de operadores que só produzem a primeira tupla da resposta após analisar toda a entrada, como é o caso da ordenação, soma, contagem, mínimo, máximo e média. Estes operadores também exigem que a entrada seja finita.
- Consulta a dados passados: no modelo computacional de fluxo de dados uma vez que um elemento de dado foi processado ele não pode ser mais consultado. Isto limita a realização de consulta a dados passados.

Em [14], são apresentados três paradigmas de consultas sobre fluxo de dados: baseado em relações, baseado em objetos, ambos declarativos, e procedimentais. O *paradigma baseado em relações* considera os fluxos e as janelas de tempo como relações ordenadas por marcações de tempo. O *paradigma baseado em objetos* classifica os fluxos em algum tipo de hierarquia. O *paradigma procedimental* permite ao próprio usuário modelar os fluxos.

Vale ressaltar que, além de consultas pré-definidas, ou seja, criadas antes dos dados serem processados, consultas sobre fluxo de dados podem ser feitas sob demanda 33. Estas consultas sob demanda podem ser tanto pontuais quanto contínuas.

Em 34 são apresentadas cinco linguagens propostas para fluxo de dados. Foi feito um estudo comparativo entre as linguagens classificando-as sob diversos aspectos como o paradigma de consulta, motivação, entradas

permitidas, operadores básicos e operadores customizados e janelas suportadas.

Também é apresentado um levantamento sobre os projetos acadêmicos de sistemas de gerência de fluxo de dados, sendo os principais:

- Aurora 35: um sistema orientado a *workflow* que cria planos de consulta baseados em caixas (operadores) e setas (fluxos).
- COUGAR 36: um DSMS para sensores.
- Gigascope 37: uma arquitetura distribuída para monitoração de redes.
- NiagraCQ 38: um sistema de consultas contínuas que permite consultas XML-QL sobre conteúdos *Web* dinâmicos.
- WebCQ 39: outro sistema de consultas contínuas para monitoração de conteúdo *Web* com foco em processamento escalável de orientado a eventos.
- StatStream 40: sistema projetado para fornecer estatísticas em tempo real sobre os fluxos de dados.
- STREAM 41: um DSMS de propósitos gerais com foco na gerência de memória e respostas aproximadas de consultas, desenvolvido pela Universidade de Stanford.
- TelegraphCQ 42: um sistema de consultas contínuas que foca em consultas compartilhadas e processamento adaptativo de consultas desenvolvido pela Universidade de Berkley.
- Esper 44: um sistema para processamento de fluxo de eventos disponível em Java e .NET que se utiliza de programação para consultas no formato SQL.
- Cayuga 45: um sistema de processamento de fluxos de eventos através de consultas SQL que propõe superar limitações dos DSMS em direção dos Processadores Complexos de Eventos.

Além do grande apelo acadêmico, os DSMS´s se tornaram produtos comerciais de grandes empresas. Atualmente são vendidos como Processadores Complexos de Eventos, ou simplesmente CEP, mas todos possuem linguagem do tipo SQL para realizar consultas sobre os fluxos de eventos. Exemplos destes produtos são o Coral8 46 e o StreamBase 47.

No artigo onde é descrito o Cayuga 45, é apresentada a diferença entre um sistema do tipo CEP e um DSMS. Segundo os autores, a primeira limitação de um DSMS seria quanto à identificação de um padrão seqüencial de eventos.

Embora seja possível, detectar este tipo de padrão, geralmente resultam em consultas de difícil compreensão e otimização. Como segunda limitação, é apresentada a dificuldade de se executar um número alto de consultas correntes em DSMS. Segundo os autores, os sistemas do tipo CEP escalam mais neste sentido do que DSMS.

Contrapondo este limitação de desempenho, estudos publicados [48, 49] questionaram o uso de DSMS para análise de tráfego de rede, uma aplicação com alto volume de tráfego e diferentes tipos de consultas. O resultado deste estudo preliminar que usou o TelegraphCQ 42 como DSMS foi satisfatório tanto do ponto de vista funcional, pois conseguiram exprimir o que necessitavam através de consultas simples, quanto de desempenho, pois conseguiram chegar a taxas de até 5.8 MB/s em um pequeno servidor. Os detalhes do teste podem ser encontrados no artigo 48.

Os testes, no entanto, não compreendem o comportamento do TelegraphCQ com relação a um número grande de consultas simultâneas, mas sim a influência de diferentes tipos de consultas, sobre fluxos de diferentes cardinalidade e fluxos de diferentes tamanhos.

Estes resultados são motivadores para esta tese, pois demonstram que o TelegraphCQ, assim como foi útil e capaz de lidar com altos volumes para a análise de eventos de tráfego rede, pode ser um componente factível para análise de um fluxo de eventos de logs. Por esta razão, o TelegraphCQ foi utilizado como componente de análise no protótipo desenvolvido no Capítulo 5.