4
Proposta de arquitetura

4.1
Requisitos de um Sistema de Gerenciamento de Logs

Os requisitos aqui apresentados têm o objetivo de expressar as necessidades de um Sistema de Gerenciamento de Logs genérico, ou seja, que cubram as demandas da maioria de empresas e instituições. Contudo, as restrições e especificidades de uma grande empresa de *Internet* e, mais especificamente, a Globo.com, serão consideradas.

Vale lembrar, também, que por se tratar de uma pesquisa o sistema proposto deve ter ligação com sistemas frutos de pesquisas anteriores e permitir que novas pesquisas sejam realizadas. A seguir são listados os requisitos, divididos em requisitos funcionais e não-funcionais.

4.1.1
Requisitos Funcionais

- Análise léxica e transformação: o sistema deve permitir que diferentes arquivos de log em diferentes padrões possam ser analisados lexicamente e transformados para um padrão comum do sistema. A flexibilidade de se trabalhar com diferentes padrões de linhas de log é fundamental, pois em um ambiente dinâmico de *Internet* muitos sistemas e, conseqüentemente tipos de log, convivem simultaneamente em operação.
- Visualização em tempo real de eventos de log: o sistema deve permitir que eventos de log possam ser visualizados em tempo real. Isto deverá permitir que analistas acompanhem a execução de um sistema distribuído, ou ainda de diversos sistemas que componham um determinado serviço, através dos eventos de log gerados pelo sistema. A interface deve permitir filtros básicos de visualização por grupo de sistemas, níveis de severidade e hierarquia. Além disso, a interface deve ser *Web* e dinâmica no sentido de ser capaz de mostrar um fluxo de eventos sem a necessidade de uma recarga forçada pelo usuário.

- Comunicação: os componentes do sistema devem ser capazes de se comunicarem uns com os outros, de forma que um transmissor possa atingir muitos receptores. O meio de comunicação deve garantir que as mensagens com os eventos de log não sejam perdidas e que a ordem em que foram transmitidas seja a mesma que a ordem recebida.
- Processamento complexo de eventos em tempo real: o Sistema de Gerenciamento de Log deve permitir que operações complexas de correlação e consultas sejam feitas sobre eventos de log que estejam acontecendo. Estas operações de correlação podem ser expressas através de uma linguagem de regras ou através de linguagens do tipo SQL. Este processamento pode ter tanto um foco técnico quanto um foco de negócio.
- Hierarquia de eventos de log: o sistema deve permitir uma diferenciação de eventos produzidos a partir de uma correlação de um conjunto de eventos. É esperado que eventos produzidos após a correlação de outros eventos tenham uma maior semântica, ou seja, sintetizam em um padrão reconhecido ou representam uma situação de interesse e, por isso, precisam de alguma forma ser diferenciados de eventos brutos, que não são fruto de uma análise.
- Armazenamento: o sistema deve permitir que os eventos de log sejam armazenados para futuras consultas. Estas consultas podem ter uma motivação de auditoria ou investigação, ou ainda de entendimento de padrões e mineração de informação. Considerando o volume alto que armazenar logs de uma empresa de *Internet* representa, o mecanismo de armazenamento deve estar associado a algoritmos de compactação. Vale ressaltar, também, que o armazenamento pode se dividir em dois sistemas diferentes, um voltado para consultas mais freqüentes em espaços de tempo curtos, entre uma semana e poucos meses, e outro voltado para consultas pontuais, que podem atuar sobre registros mais antigos de log.
- Pesquisa e visualização de histórico: o sistema deve permitir que os eventos de log armazenados possam ser pesquisados através de consultas e visualizados em uma interface *Web*. Esta funcionalidade está fortemente relacionada com a atividade de armazenamento para visualização freqüente.
- Mineração de padrões sobre os logs: o sistema deva permitir que através de uma análise mais refinada sobre os eventos de log possam ser

inferidos padrões de interesse. Estes padrões serão expressos então através de regras ou consultas nos Processadores Complexos de Eventos.

4.1.2 Requisitos Não-funcionais

- Baseado em sistemas de código aberto e gratuito: o Sistema de Gerenciamento de Logs não deve incorporar sistemas proprietários e de código fechado em sua arquitetura. A razão para isto é permitir que qualquer empresa ou instituição possa aplicar o sistema proposto sem custo de licenças. Outra vantagem em se basear em sistemas de código aberto é permitir acesso aos detalhes de implementação, permitindo assim um melhor conhecimento e possíveis modificações no sistema. Além disso, está de acordo com o direcionamento de muitas empresas de *Internet* que possuem suas arquiteturas baseadas em *software* livre.
- Não intrusivo e desacoplado: o processamento de eventos de log deve onerar o mínimo possível os sistemas em produção, ou seja, os sistemas que atendem aos usuários ou clientes. Desta forma, todo o processamento gerado pela análise léxica e transformação e pela correlação de eventos e comunicação não pode ser significante para o sistema em produção. Além de não ser intrusivo, o Sistema de Gerenciamento de Log deve estar desacoplado ao máximo dos sistemas em produção. Dessa forma, qualquer falha no Sistema de Gerenciamento de Log não deverá afetar os sistemas em produção.
- Escalável e capaz de atender a alto volume: o Sistema de Gerenciamento de Log deve ser projetado para poder atender a altos volumes de eventos. Em uma grande empresa de *Internet*, o volume de um único portal pode chegar a dezenas de G*iga bytes* por dia. Além disso, o sistema deve ser projetado de maneira a crescer de forma de forma horizontal, ou seja, através da adição de mais servidores, e não através da compra de servidores mais poderosos. É importante também que o Sistema de Gerenciamento de Log possa ser particionado, caso seja inviável atender a toda a demanda da empresa ou instituição em uma única estrutura.

4.2
Conceito de Arquitetura Orientada a Eventos

A arquitetura a ser apresentada na seção 4.3 foi concebida tendo em vista o padrão de projeto de Arquitetura Orientada a Eventos, do inglês *Event-Driven Archictecture* (EDA). Este padrão de arquitetura, conforme apresentado em [26, 50, 51], possui as seguintes características principais:

- Comunicação por difusão (*broadcast*): componentes participantes difundem eventos para qualquer outro componente interessado. Desta forma, mais de um componente pode receber e processar um evento.
- Fluxo temporal: componentes publicam eventos conforme ocorrem, ao invés de guardá-los localmente esperando por um ciclo de processamento em lote.
- Baixo acoplamento: o componente que gerou o evento não tem conhecimento algum e não é dependente de quaisquer processamentos posteriores que o evento possa sofrer por outros componentes.
- Assíncrona: o componente que publica um evento não espera pelo processamento dos componentes que recebem o evento.
- Eventos de baixa granularidade: componentes publicam eventos concretos e não somente eventos sintéticos.
- Taxonomia: o sistema como um todo define uma nomenclatura para classificar os eventos, tipicamente em forma hierárquica.
- Processamento Complexo de Eventos: conforme já explicado na seção 3.2, corresponde a correlacionar, identificar padrões e sintetizar eventos.

Em 26 são definidos quatro componentes de uma Rede de Processamento de Eventos, ou EPN (*Event Process Network*). Os componentes e a suas integrações formam a EPN, que por suas vez, está de acordo com os padrões de uma EDA. Os componentes são:

- Produtor de evento: componente que publica eventos em um canal de eventos.
- Consumidor de evento: componente que consome eventos de um canal de eventos.
- Canal de eventos: mecanismo responsável pela entrega dos fluxos de eventos produzidos pelos Produtores de evento e pelos Agentes processadores de eventos para Consumidores de evento e Agentes processadores de eventos.

- Agente Processador de evento: componente responsável por detectar padrões em eventos concretos, processar estes eventos e criar eventos sintéticos.

Para um melhor entendimento do que seria uma EPN, a visão em camadas apresentada em 50 ajuda a compreender a interação entre os componentes. Embora, a autora denomine como camadas, esta denominação não é a mais adequada por se referir as fases de um fluxo e não a níveis de um fluxo. Sendo assim, uma boa adaptação seria denominá-los como estágios, ao invés de camadas.

Vale ressaltar que a imagem de interações meramente seqüenciais não corresponde ao propósito de uma EDA. Contudo, as fases de um fluxo onde as interações acontecem seqüencialmente ajudam no entendimento da arquitetura. Os quatro estágios do fluxo de interações são: geração de eventos, comunicação pelo canal, processamento de eventos e disparo de atividades. A Figura 5 representa estes estágios.

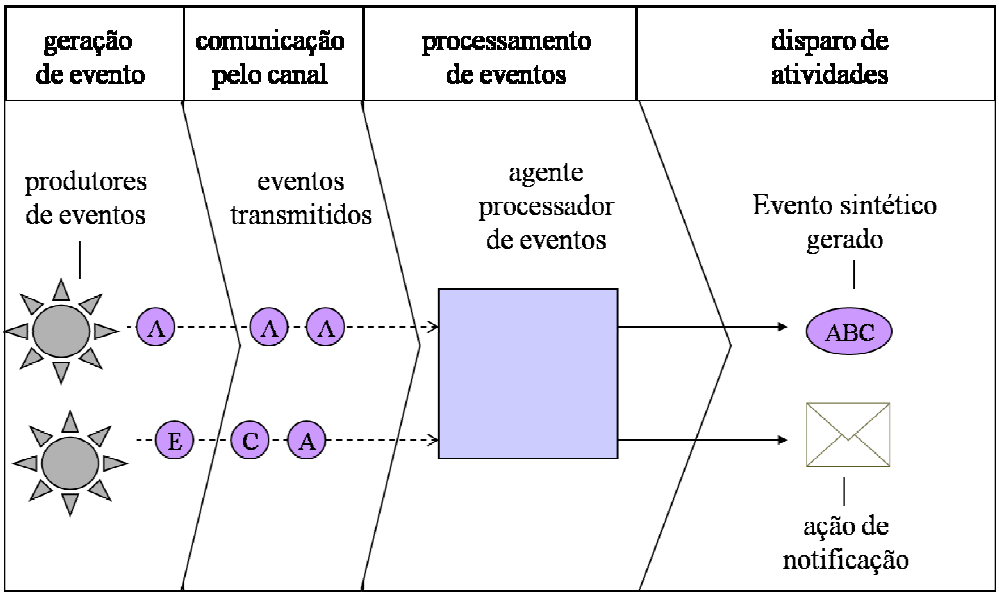

Figura 5 – Estágios de interações fluxo de eventos

4.3
Arquitetura Orientada a Eventos para Gerenciamento de Eventos de Log

Fischer 5 em sua tese cita a abordagem de uma Arquitetura Orientada a Eventos como sendo um possível próximo passo na evolução dos sistemas de gerenciamento de log. Contudo, o autor não aprofunda esta discussão na elaboração de uma arquitetura, e no posterior desenvolvimento de um protótipo. Neste sentido, esta dissertação complementa o trabalho de Fischer 5, apresentando um modelo de arquitetura, tópico abordado nesta seção, e desenvolvendo um protótipo, tópico do Capítulo 5.

A proposta da arquitetura está baseada nos requisitos apresentados na seção 4.1, e tem como orientação o padrão EDA apresentado na seção 4.2. Nesse sentido, os conceitos de uma arquitetura orientada a eventos são utilizados para projetar uma arquitetura para o gerenciamento de logs.

No centro da arquitetura proposta está o Canal Principal de Comunicação de Eventos. Este canal deve suportar as funcionalidades básicas de uma comunicação em grupos, permitindo a criação e exclusão de grupos, inclusão e exclusão de nós (apresentados a seguir) nos grupos, e o envio e recepção de eventos em um grupo. Além disso, o Canal Principal de Comunicação deve garantir que todo o evento publicado seja entregue aos participantes do grupo e que a ordem dos eventos gerados seja a mesma dos eventos recebidos.

Os grupos de um Canal Principal de Comunicação de Eventos representam uma área de interesse onde os demais nós podem se inscrever para publicar ou receber eventos. Apesar de grupos não serem considerados como componentes do sistema, a existência de grupos no Canal Principal de Comunicação de Eventos traz a possibilidade de se criar segmentações para os eventos de log, podendo assim balancear o volume e a carga, ou especializar os nós que realizarão atividades sobre os eventos.

Conectados ao Canal Principal de Comunicação de Eventos estão os nós. Define-se como *nó*, nesta arquitetura, o conjunto de componentes e Canais Locais de Comunicação capazes de publicar ou consumir eventos do Canal Principal de Comunicação de Eventos.

A Figura 6 ilustra um Canal Principal de Comunicação de Eventos que possui três nós. Por exemplo, o Nó 1 analisa lexicamente todos os arquivos de log relacionados a um portal na *Internet*. O Nó 1 publica eventos de log relacionados a erros com servidores *Web* no “grupo A”, eventos de log relacionados a erros com a aplicação no “grupo B” e eventos de log relacionados

a eventos de acesso no “grupo C”. O Nó 2 está associado aos grupos A e B e consome os eventos publicados nos dois grupos com a finalidade de correlacionar os eventos e identificar os erros gerados pela aplicação que refletiram em páginas de erro para usuário na *Internet*. O Nó 3 está associado ao grupo C e correlaciona os eventos de acesso para identificar picos e vales de acesso em uma determinada página publicada.

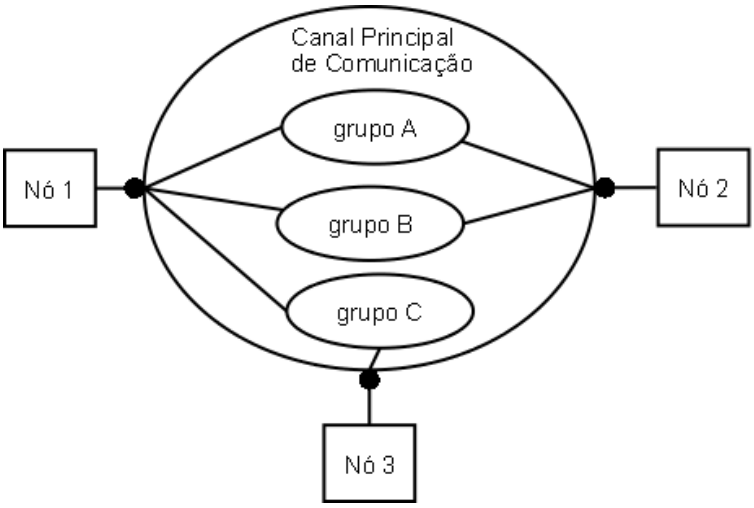

Figura 6 – Canal Principal de Comunicação de Eventos, grupos e Nós.

Olhando para a composição de um nó, tem-se agora os *componentes*.,que são os elementos de mais baixo nível nesta arquitetura, ou seja, não são subdivididos em mais elementos. Cada componente pode desempenhar o papel de um Produtor, Consumidor ou Agente Processador de Eventos. Para se comunicar com outros componentes de um mesmo nó, os componentes utilizam os Canais Locais de Comunicação; para se comunicar com o Canal Principal de Comunicação, os componentes utilizam os Canais Externos de Comunicação.

Em outras palavras, os Canais Locais de Comunicação são os meios pelos quais os processos de um componente se comunicam com processos de outro componente em um mesmo nó. Já os Canais Externos de Comunicação são os meios pelos quais um componente publica ou consome eventos para do Canal Principal de Comunicação de Eventos.

A Figura 7, ilustra a composição de um nó onde, por exemplo, pode-se ter um componente responsável pela análise léxica de um arquivo de log, que se comunica com outro componente que faz uma correlação simples de eventos e repassa para um terceiro componente que será responsável por transmitir estes eventos no Canal Principal de Comunicação de Eventos.

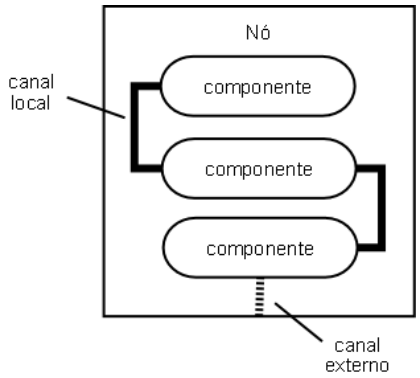

Figura 7 – Composição de um Nó

É importante ressaltar que mais de um Canal Principal de Comunicação de Eventos pode existir em um Sistema de Gerenciamento de Log. Isto depende da necessidade em se segmentar o sistema, motivada principalmente pelo volume de eventos.

Depois da descrição dos elementos que compõem a estrutura de um Sistema de Gerenciamento de Log, é preciso determinas as funções que os nós devem tipicamente desempenhar para atender os requisitos descritos na seção 4.1.1.

- Nó de análise léxica, transformação e publicação de eventos de log: este tipo de nó está posicionado junto as servidores que geram os arquivos textos com as linhas de log. É seu papel interpretar cada linha gerada e transformá-las em um padrão do sistema. Em seguida, este nó deve encaminhar os eventos para que sejam publicados no Canal Principal de Comunicação de Eventos.
- Nó de processamento complexo de eventos: este tipo de nó deve ser capaz de correlacionar os eventos de log, segundo regras e consultas previamente definidas. Este tipo de nó deve ainda ter a capacidade de produzir eventos sintéticos a partir de um conjunto de eventos que correspondam a um padrão.
- Nó de conhecimento adaptativo: este tipo de nó deve ser capaz de analisar os eventos de log, aprender sobre a relação entre os eventos e situações interessantes para, então, inferir padrões interessantes de eventos. Este nó é importante em ambientes extremamente dinâmicos, possibilitando que novas regras possam ser inferidas de acordo com as mudanças nos ambientes.

- Nó de visualização em tempo real de eventos de log: este tipo de nó deve permitir que os eventos de log que estejam acontecendo em um determinado instante de tempo sejam visualizados.
- Nó de armazenamento de eventos de log: este tipo de nó deve armazenar os registros de eventos de log para assim possibilitar uma possível pesquisa sobre os eventos.
- Nó de pesquisa de eventos de logs: este tipo de nó deve permitir que usuários façam pesquisas sobre uma base de eventos. Os resultados da pesquisa devem ser mostrados em uma interface *Web.*

Com isto temos a visão geral da arquitetura proposta, com o Canal Principal de Comunicação de Eventos e os nós desempenhando papéis específicos de um LMS. A Figura 8 a seguir mostra os componentes da arquitetura proposta:

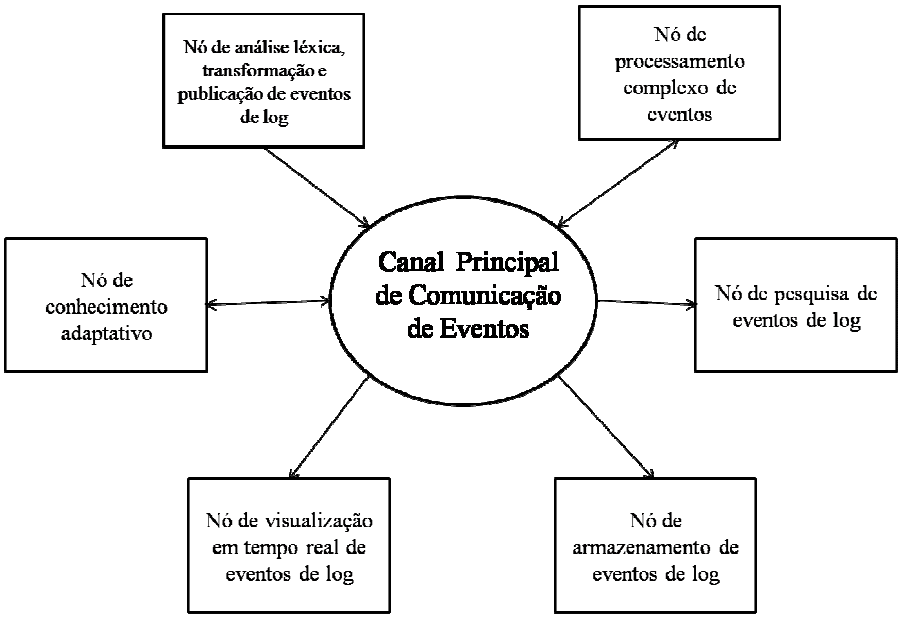

Figura 8 – Arquitetura proposta e seus componentes

Por fim, após descrever os elementos da arquitetura e os tipos de nós que devem estar presentes, é necessário descrever o formato de um evento de log. A primeira decisão importante consiste em determinar como as informações de um evento de log devem ser estruturadas. Considerando as informações básicas de um registro de log, levantadas no Capítulo 2, e os requisitos de um Sistema de Gerenciamento de Log, foram definidas as seguintes informações:

- Identificação do nó de origem: identifica o nó responsável por publicar o evento de log.
- Marcação temporal: data e hora de ocorrência do evento.
- Nível hierárquico: indica grau de síntese do evento, ou seja, o quanto aquele evento é fruto de análises anteriores. Inicia-se em 0, para os eventos que apenas representam uma linha de log, e a cada vez que um evento é gerado a partir de um conjunto de eventos adiciona-se 1.
- Severidade: representa o quão crítico é o evento, por exemplo, se se trata de um erro ou apenas uma informação.
- Palavras-chave: uma forma livre de se classificar o evento. As palavras-chave permitem classificação de um evento sobre diferentes aspectos, técnicos ou não.
- Descrição: texto livre que complementa a informação sobre um evento.

Para a formatação do evento de log, optou-se por um formato bastante conhecido e utilizado principalmente por sua simplicidade: o CSV (*Comma-separated values*) 52. Como o nome indica, o CSV utiliza o caractere vírgula para separar os campos. A não utilização de um formato mais estruturado, como o XML 53, justifica-se pela economia de banda. Em um teste simples, pode-se notar que percentualmente a adoção de XML, por exemplo, oneraria em até três vezes mais o consumo de banda para o tráfego de eventos de log. A Tabela 4 mostra um mesmo evento de log em formato CSV e um possível formato XML.

Evento de log em formato CSV:
no1,2008-10-10 10:00:00,0,ERROR,site1|banco|busca,Erro no sistema de banco de dados
Evento de log em formato XML:
<event> <source>no1</source> <timestamp>2008-10-10 10:00:00</timestamp> <hierarchy>0</hierarchy> <severtiy>ERROR</severtiy> <tags> <tag>site1</tag> <tag>banco</tag> <tag>site2</tag> </tags> <description> Erro no sistema de banco de dados</description> </event>

Tabela 4 – Diferentes formatos de eventos de log

Neste simples teste, a diferença de tamanho entre os eventos foi percentualmente significativa, de 83 *bytes* no formato CSV para 284 *bytes* no formato XML, o que corresponde a aproximadamente três vezes o tamanho de um evento. Ao se trabalhar com grandes volumes de dados, o aumento de banda que seria introduzido pelo padrão XML pode ser crítico para a operação do sistema.

Quanto ao uso de palavras-chave nos eventos de log, pode-se dizer que sua importância está em permitir que um evento possa ser classificado e posteriormente filtrado ou correlacionado utilizando-se desta informação. Com o uso de palavras-chave, mais de uma árvore de classificação de eventos pode ser criada no sistema, ou seja, os eventos podem ser classificados sob diferentes aspectos, como por exemplo através de uma árvore de classificação baseada em componentes de tecnologia (servidor *Web*, aplicação, banco de dados) e de uma árvore baseada em componentes de um portal (página principal, menu lateral, espaço comercial).

Vale ressaltar que, para o campo severidade, além dos níveis utilizados geralmente por aplicações, como “ERROR”, “INFO”, “DEBUG”, são também

utilizados códigos[1] HTTP que também trazem informação sobre a severidade de uma requisição, como código 200 para uma página entregue, código 404 para uma página não encontrada ou código 500 na ocorrência de um erro.

[1] Uma listagem completa com os códigos HTTP podem ser encontrada na referência 9.