

Geração de *logs* de Experimentos de Injeção de Falhas para Análise de Dependabilidade de Aplicações Distribuídas*

Joana M. F. Trindade, Gabriela Jacques-Silva,
Taisy Silva Weber, Ingrid Jansch-Pôrto

Instituto de Informática – Universidade Federal do Rio Grande do Sul
Caixa Postal 15064 – 90501-970 Porto Alegre, RS

{jmftrindade,gjsilva,taisy,ingrid}@inf.ufrgs.br

Resumo. *Sistemas distribuídos, por suportarem aplicações de alta disponibilidade, necessitam que sua dependabilidade seja validada. A injeção de falhas, nestes casos, torna-se indispensável, pois possibilita a inserção do tipo de falha desejada e no momento adequado aos testes de validação. Para facilitar uma análise do comportamento do sistema em resposta às falhas, é necessário monitoramento. O objetivo deste artigo é apresentar o sistema de monitoramento de FIONA quando em arquitetura local e distribuída, a geração de logs e sua relação com análise de dependabilidade de sistemas distribuídos.*

Abstract. *The support of high availability applications by distributed systems makes the validation of system dependability necessary. The use of fault injection mechanisms, in such cases, is essential, because it makes it possible to inject desired kinds of faults, at appropriate moments, for the validation tests. To provide a centralized analysis of the system's behaviour in response to the injected faults, monitoring is required. This paper presents FIONA's system to monitor its local and distributed architectures, the log generation and how it relates to distributed systems dependability analysis.*

1. Introdução

Em sistemas que suportam aplicações de alta disponibilidade, como *clusters*, sistemas distribuídos, servidores *WEB*, uma das características mais desejáveis é a confiabilidade desses sistemas. Objetivando o aumento na dependabilidade [Avizienis et al., 2004] de tais sistemas, são utilizadas mecanismos de tolerância a falhas. Uma técnica muito utilizada, com o fim de assegurar a eficácia de sistemas no fornecimento de seus serviços, é a validação através de testes de injeção de falhas.

A injeção de falhas consiste na inserção de falhas em um sistema ou aplicação, em um ambiente controlado, e na determinação do comportamento deste sistema em resposta às falhas nele injetadas [Hsueh et al., 1997]. Em sistemas de grande complexidade, em que soluções tradicionais de teste de software e confiabilidade não bastam para revelar importantes problemas de implementação, injeção de falhas é geralmente a única alternativa. E, por isso, na avaliação de dependabilidade do sistema, a injeção de falhas se torna essencial. Em ambientes de testes de validação, o tempo de espera por uma falha real é indeterminado, o que aumentaria o tempo de condução dos testes e prejudicaria uma

*Financiado pelos projetos ACERTE/CNPq (472084/2003-8) e DepGriFE/HP P&D Brasil

monitoração satisfatória de seu comportamento. Através da injeção de falhas, no entanto, é possível introduzir falhas no momento e do tipo necessários ao estudo do comportamento do sistema a ser analisado.

Entretanto, o comportamento do sistema alvo nem sempre pode ser facilmente observado por quem conduz os experimentos de injeção de falhas. É necessária, portanto, a implementação de mecanismos de monitoramento para observar o comportamento do sistema em resposta às falhas injetadas. Além disso, é preciso observar tanto as informações relativas ao experimento geradas pelo injetor de falhas, quanto os *logs* gerados pelo mecanismo de tolerância a falhas da aplicação alvo. Assim, com estas informações e a utilização de uma ferramenta de análise de dependabilidade apropriada ao experimento, é possível obter medidas confiáveis de dependabilidade.

Este artigo visa apresentar o sistema de monitoramento do ambiente distribuído para injeção de falhas FIONA (*Fault Injector Oriented to Network Applications*) [Jacques-Silva et al., 2004]. Serão explicados o funcionamento do sistema de coleta de dados acerca do experimento, quando injetadas falhas em arquiteturas locais e distribuídas. A geração de *logs* contendo os dados coletados na etapa anterior e, em seguida, a centralização dos vários *logs* gerados por FIONA no sistema distribuído, em um único arquivo ordenado.

A estrutura é como segue: A Seção 2 apresenta o sistema de monitoramento de FIONA. A Seção 3 descreve a coleta e a gravação dos dados de monitoramento em arquiteturas locais e distribuídas. E, por fim, a Seção 4 apresenta as considerações finais do artigo.

2. Injetor de Falhas FIONA

FIONA é um injetor de falhas de comunicação para aplicações em Java, que tem como objetivo a validação de sistemas distribuídos de pequena e larga escala. Apresentando uma arquitetura distribuída, FIONA visa facilitar a condução de experimentos de injeção de falhas, através da configuração centralizada de cenários de falhas que afetam o sistema sob teste. Na busca por escalabilidade, sua arquitetura distribuída foi baseada em uma proposta para arquitetura de monitoramento de larga escala. Tal estratégia foi utilizada no estabelecimento de uma hierarquia entre os injetores, classificando-os em três tipos de comportamento: *injetor principal*, *injetor de site* e *injetor local*.

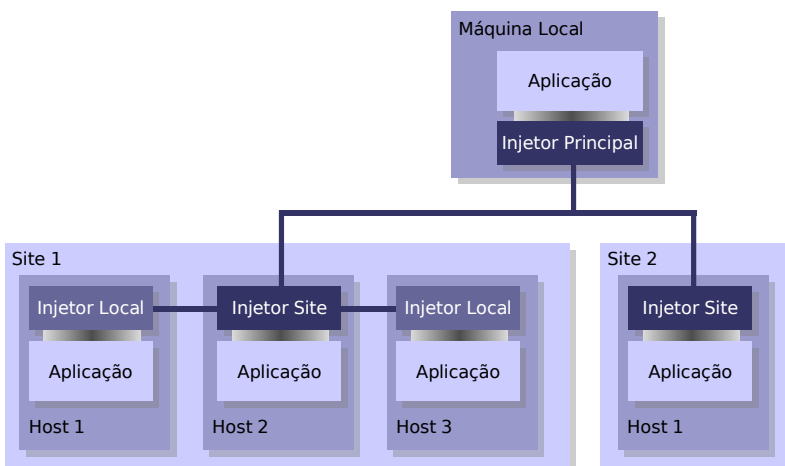


Figura 1: Arquitetura distribuída do FIONA

Um *injetor local* executa em uma máquina e se comunica com o *injetor de site*. Cada injetor é responsável pela geração do seu próprio log. Os *injetores de site* se encarregam do recolhimento de tais *logs*, repassando-os ao *injetor principal*. O *injetor principal* é o responsável pelo gerenciamento do experimento. É ele que distribui a configuração do experimento e faz a centralização dos *logs* de todos os nodos participante do sistema, além de ser o responsável pela ativação de falhas que afetam mais de um nodo simultaneamente, como particionamento de rede.

FIONA baseia-se no modelo de falhas para sistemas distribuídos, proposto por Birman [Birman, 1996]. Por objetivar a validação de sistemas distribuídos de larga escala, FIONA possibilita a injeção das falhas mais frequentes em ambientes deste tipo. São elas: falhas de *omissão*, *colapso*, *temporização* e *particionamento*. Uma comparação entre FIONA e outros injetores de falhas encontra-se em [Jacques-Silva, 2005].

FIONA possui implementado um sistema de monitoramento, o qual visa permitir, de forma centralizada, a análise do *log* do experimento. Porém, é importante ressaltar que para obtenção de medidas de dependabilidade após o experimento de injeção de falhas, é necessário observar tanto o *log* da ferramenta de injeção de falhas quanto o gerado pela aplicação alvo. Os *logs* da aplicação são dependentes da mesma, e não são tratados pelo sistema de monitoramento de FIONA. Estes devem ser passados como entrada para uma ferramenta apropriada de análise de dependabilidade, em conjunto com os *logs* do experimento de injeção de falhas.

O sistema de monitoramento de FIONA é dividido em duas partes: o monitoramento realizado localmente e o monitoramento de todos os nodos componentes do sistema (monitoramento distribuído).

2.1. Monitoramento Local

O monitoramento local é a geração de *logs* do experimento de injeção de falhas no próprio nó. Durante a execução, a cada falha injetada, é gerada uma entrada no arquivo de *log*. Cada entrada é composta por um conjunto de informações, tais como: qual o momento em que a falha foi injetada, o tipo de falha injetada, o número da mensagem no experimento, o endereço IP dos nodos de origem e destino e as portas utilizada na troca de mensagens entre os nodos.

2.2. Monitoramento Distribuído

Todos os nodos do sistema gravam um conjunto de informações sobre cada falha injetada no seu *log*, seja ele um injetor local, de *site* ou principal. Ao início de um experimento de injeção distribuída, cada nodo assume a responsabilidade por seu próprio *log*, e ao término da execução, este é enviado ao nodo de posição diretamente acima na hierarquia (Figura 1). Se o nodo em questão for um injetor de *site* ou principal, ele aguarda até todos os outros nodos a ele conectados um nível abaixo na hierarquia, enviarem-lhe seus *logs*. Um *injetor de site*, ao receber um *log*, repassa-o ao *injetor principal*, que também espera a recepção das informações de todos os nodos a ele conectados (*injetores de site*) para finalizar sua execução.

Uma vez que a análise dos resultados deve ser feita de forma centralizada, o salvamento final das informações é realizado apenas pelo *injetor principal*, podendo este ser tanto um injetor de falhas da aplicação, como somente um coordenador do experimento, não a penalizando em desempenho neste segundo caso.

Cada *log* recebido de um *injetor de site* pelo *injetor principal* é gravado em disco. Com o *log*, são armazenados também o endereço da máquina a que pertenciam

as informações e o número do processo correspondente ao experimento. Nos nodos injetores locais e de *site*, as informações referentes às falhas injetadas são manipuladas por meio de *buffers*, salvos em memória, e enviados ao nodo a que estão conectados.

3. Ordenação de Logs

Para uma obtenção de medidas de dependabilidade confiáveis, a ferramenta injetora de falhas deve ser eficiente tanto na injeção de falhas propriamente dita, quanto na coleta dos dados referentes ao experimento e ao comportamento do sistema em reação às falhas injetadas. Em FIONA, seu sistema de monitoramento é responsável pela coleta e salvamento centralizado dos dados gerados durante a execução do experimento de injeção de falhas. Como há a possibilidade de mais de um nodo estar injetando falhas em uma aplicação, a busca pelas informações de todas as falhas injetadas por um nodo, por exemplo necessita de uma visão global de toda a execução do experimento. Para isto, os registros de cada falha injetada devem estar devidamente ordenados.

Entretanto, por se tratarem de várias máquinas com tempos locais diferentes entre si, uma ordenação direta dos tempos salvos em *log*, acabaria fornecendo uma ordem de acontecimentos inconsistente. Assim, é preciso que os dados referentes ao experimento sejam ordenados levando-se em consideração as diferenças de relógio entre cada uma das máquinas.

Neste sentido, foi desenvolvido o programa LOrd (*Log Orderer*). Implementado em Java, LOrd é um ordenador de *logs* gerados pelo sistema de monitoramento do ambiente distribuído de injeção de falhas FIONA. Esta seção é dividida em duas subseções: A Subseção 3.1 explica o método de ordenação utilizado e a Subseção 3.2 descreve sua implementação.

3.1. Método de Ordenação

Cada arquivo de *log* é gravado em disco com o horário local de seu nodo. Para fins de ordenação, então, é preciso que sejam padronizados os diversos horários em uma única referência temporal base. Isso é feito utilizando-se o algoritmo de sincronização de relógios, proposto por [Maillet e Tron, 1995]. Este método propõe a criação de um único relógio lógico através do cálculo das diferenças de hora local entre cada uma das máquina do sistema e uma máquina de referência r .

Esta diferença é calculada, para cada máquina, uma vez no início e outra ao final do experimento. r envia uma mensagem a cada máquina. A máquina remota, então, obtém sua hora local e envia este valor de volta a r . O horário local de r é lido antes do envio e após o recebimento desta mensagem. Calcula-se o atraso na comunicação, que corresponde a uma média aritmética entre os dois valores acima obtidos. Este atraso é considerado equivalente ao valor lido na segunda máquina.

Segundo o algoritmo, a relação entre o horário local registrado pela máquina i , no tempo absoluto t e a hora local registrada pela máquina de referência r no mesmo tempo absoluto, t é regida pela equação:

$$lt_i(t) = \alpha_i + \beta_i lt_r(t) + \delta_i$$

Onde o tempo absoluto é representado por t , a constante α_i equivale ao valor do relógio local da máquina i , quando r possui $t = 0$. A diferença do horário local da máquina i em relação ao horário de r é igual a β_i . A granularidade e todas as outras

perturbações independentes de t , pequenas o suficiente para serem descartadas, são representadas por δ_i .

LORD corrige o horário local de i lhe atribui um horário global $LC_i(t)$, sendo estimado por:

$$LC_i(t) = lt_r(t) \approx \frac{lt_i(t) - \alpha_i}{\beta_i}$$

3.2. Implementação

LORD é executado duas vezes, uma antes do início do experimento e outra logo após o seu término. Através de uma operação de *ping-pong* é feita a troca de mensagens entre a máquina referência r e cada uma das máquinas i . Por meio desta troca, são coletadas as amostras a serem utilizadas por LORD no cálculo dos coeficientes de correção de horário para cada máquina. Em seguida, utilizando-se de uma classe de leitura, LORD lê os *logs* gerados por FIONA.

Cada *log* é formado por uma ou mais linhas, em número igual ao de falhas injetadas. A primeira linha contém o nome do nodo a que pertence o arquivo. As demais linhas possuem informações sobre a falha correspondente. Cada registro de entrada no arquivo possui um número de campos dependente do tipo de falha injetada, podendo ser: *hora local*, *tipo de falha*, *número da mensagem*, *ip e porta de origem*, *ip e porta de destino*, *injeção realizada no envio ou na recepção*, *tempo de atraso da mensagem*.

Por ser o que contém informação referente ao horário de injeção da falha, *hora local* é o único campo além do nome da máquina (obtido na primeira linha do arquivo) a ser utilizado na geração dos coeficientes de correção. Com base nestes coeficientes, são ajustados os horários contidos nos *logs*, seguindo-se o algoritmo de sincronização apresentado. Como saída do programa, é produzido um único arquivo de *log*, em que todas as falhas injetadas estão ordenadas por tempo num horário global.

LORD pode, adicionalmente, ser adaptado para quaisquer formatos de *log*, além daquele utilizado por FIONA. Para isso, basta a simples implementação de uma classe de leitura correspondente, assim permitindo a sincronização de outras aplicações que necessitem de sincronização *offline*.

4. Conclusão

Em injeção de falhas, mecanismos de monitoramento se fazem necessários para uma eficiente coleta de informações sobre as falhas injetadas na aplicação alvo. E a geração de *logs*, por estes mecanismos de monitoramento, é essencial na obtenção de métricas, na análise de dependabilidade e na determinação do comportamento das aplicações em resposta às falhas injetadas.

Este artigo apresentou FIONA, um ambiente para condução de experimentos de injeção de falhas, e seu sistema de monitoramento. FIONA possibilita a injeção de falhas localmente (em apenas um nodo) ou em arquiteturas distribuídas e, nesta segunda opção, gerando *logs* independentes. Seu próprio sistema de monitoramento recolhe estes *logs* ao final do experimento.

Para sincronização dos *logs*, foi desenvolvida a ferramenta (LORD), a qual utiliza um método de sincronização *offline*. A escolha deste método não penaliza em desempenho a execução da aplicação, pois, por ser um algoritmo *offline*, é executado após o

término do experimento, permitindo a cada nodo a geração de seu *log* de maneira independente. A utilização de LOrd garante consistência de ordem no histórico dos eventos ocorridos, tornando mais simples o cruzamento do *log* do experimento com o da aplicação alvo, facilitando a análise de dependabilidade da aplicação.

Referências

- Avizienis, A., Laprie, J.-C., Randell, B., e Landwehr, C. E. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33.
- Birman, K. P. (1996). *Building Secure and Reliable Network Applications*. Manning Publications, Co, Greenwich.
- Hsueh, M.-C., Tsai, T. K., e Iyer, R. K. (1997). Fault injection techniques and tools. *IEEE Computer*, 30(4):75–82.
- Jacques-Silva, G. (2005). Injeção distribuída de falhas para validação de dependabilidade de sistemas distribuídos de larga escala. Mestrado em ciência da computação, Instituto de Informática, UFRGS, Porto Alegre.
- Jacques-Silva, G., Drebes, R. J., Gerchman, J., e Weber, T. S. (2004). FIONA: A fault injector for dependability evaluation of Java-based network applications. In *Proc. of the 3rd IEEE Intl. Symposium on Network Computing and Applications*, páginas 303–308, Cambridge, MA. IEEE Computer Society Press.
- Maillet, E. e Tron, C. (1995). On efficiently implementing global time for performance evaluation on multiprocessor systems. *Journal of Parallel and Distributed Computing*, 28(1):84–93.