Utilização de Padrões de Projeto na Arquitetura da FSoFIST

Cláudia Santos da Silva
Eliane Martins
Nandamudi L. Vijaykumar

claudia@dea.inpe.br, eliane@ic.unicamp.br, vijay@lac.inpe.br

Resumo

Este artigo apresenta um estudo sobre padrões de projeto para auxiliar o desenvolvimento de ferramentas com suporte a injeção de falhas por software. Um protótipo de uma ferramenta em estudo é a FSoFIST (Ferry-clip with Software Fault-Injetion Support Tool) que foi projetado para dar suporte à realização de testes em protocolos de comunicação e foi desenvolvido através da extensão da arquitetura Ferry clip. A idéia desta extensão é permitir a injeção de falhas em protocolos que possuam mecanismos de recuperação e tolerância a falhas. Este estudo vem contribuir com o estudo feito anteriormente a respeito de arquiteturas de testes para auxiliar na validação de Computadores de Bordo, uma vez que o objetivo é configurar a arquitetura ferry injection para testes multi partes, a mesma arquitetura utilizada no estudo sobre padrões. A idéia é mapear um sistema de padrões para injeção de Falhas por Software à FSoFIST visando deixá-la modular, extensível, facilitando assim o seu desenvolvimento.

Palavras-chaves: *injeção de falhas, padrão de projeto, padrão para injeção de falhas, ferramentas de injeção de falhas.*

1. Introdução

O uso de sistemas computacionais em diversos tipos de aplicações vem crescendo muito, seja para auxiliar nas atividades, ou para execução de forma essencial. Como exemplo das aplicações podemos mencionar controle de tráfego, aviônica, telecomunicações, satélites. Assim uma falha que ocorra em um desses sistemas pode provocar perdas humanas ou econômicas.

Como não é possível prevenir completamente a ocorrência de falhas que possam provocar o mau funcionamento do sistema, torna-se imprescindível o uso de alguns Mecanismos de Tolerância a Falhas (MTF), que façam com que o sistema continue operando, de forma degradada ou não, mesmo após a ocorrência de falhas no sistema. Porém, não basta somente implementar um MTF para se ter um sistema tolerante a falhas é preciso investir nos testes para verificar se o mecanismo de tolerância a falhas irá tratar a falha como esperado. O uso da técnica de injeção de falhas, vem sendo empregada com o objetivo de produzir ou simular a presença de falhas nos sistemas e observar qual sua resposta nestas condições.

Várias ferramentas que implementam a técnica de injeção de falhas estão sendo desenvolvidas. Neste trabalho propomos a utilização de padrões de projeto na construção de ferramentas de injeção de falhas por software, visando facilitar e economizar tempo e esforço no desenvolvimento de tais ferramentas.

Com o objetivo de dar suporte à estratégia de teste por injeção de falhas e também suporte a outros tipos de testes, tais como testes de conformidade [1], um protótipo da ferramenta FSoFIST(*Ferry-clip with Software Fault-Injetion Support Tool)* foi desenvolvido no Instituto da Computação da Unicamp e tem sido utilizado em alguns projetos pilotos no INPE [8] [9]. Assim, a proposta do trabalho é utilizar padrões de projeto na arquitetura da FSoFIST[2].

A Seção 2 apresenta o conceito de injeção de falhas, os tipos de injeções e o padrão para injeção de falhas por software, que será empregado no trabalho. A Seção 3 apresenta a arquitetura da FSoFIST. O uso de padrões de projeto aplicados na arquitetura FSoFIST será apresentado na Seção 4. Finalmente, a Seção 5 apresenta a conclusão do artigo e direcionamento para atividades futuras.

2. Injeção de Falhas

A injeção de falhas é uma técnica de teste de sistema. Essa técnica opera através da injeção de falhas em um sistema e observa se o mesmo continua funcionando como esperado. Dessa forma, essa abordagem pode dar maior confiabilidade sobre o funcionamento do sistema [3].

Existem diferentes técnicas que podem ser utilizadas, as mais comuns são: injeção de falhas por simulação, por hardware e por software. Essas técnicas são apresentadas brevemente a seguir.

2.1. Injeção de Falhas por Simulação

Esta técnica é usada na fase de projeto e as falhas são produzidas em um modelo do sistema alvo. A injeção de falhas por simulação é útil para avaliar a confiança no funcionamento (dependability) do sistema nas primeiras fases do desenvolvimento [3], também tem a vantagem de proporcionar grande controle e observação das falhas injetadas quando comparado com injeção em um protótipo ou no sistema final.

2.2. Injeção de Falhas por Hardware

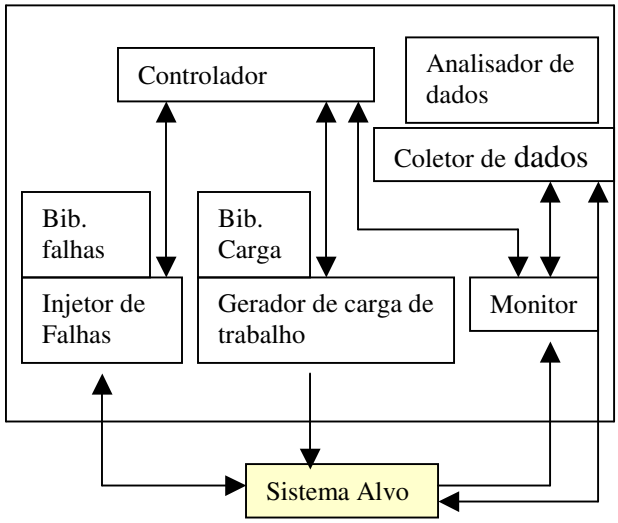
Nessa técnica, as falhas são originadas de algum hardware especial, projetado para esse propósito. Existem vários métodos para injetar falhas de hardware: pela alteração de níveis lógicos em pinos de circuitos integrados [4], por irradiação de íons pesados [5], entre outros.

2.3. Injeção de Falhas por Software

Essa técnica procura injetar falhas no software: seja no nível de sistemas operacionais, seja nas aplicações [6]. A vantagem dessa técnica é que ela não requer um hardware específico e os testes podem ser facilmente controlados.

2.4. Arquitetura Genérica para Injeção de Falhas

Muitas ferramentas para injeção de falhas já foram estudadas e desenvolvidas, e a partir disto uma arquitetura genérica para injeção de falhas foi criada [6]. Esta arquitetura é apresentada a seguir.



"Figura 1. Arquitetura genérica para injeção de falhas"

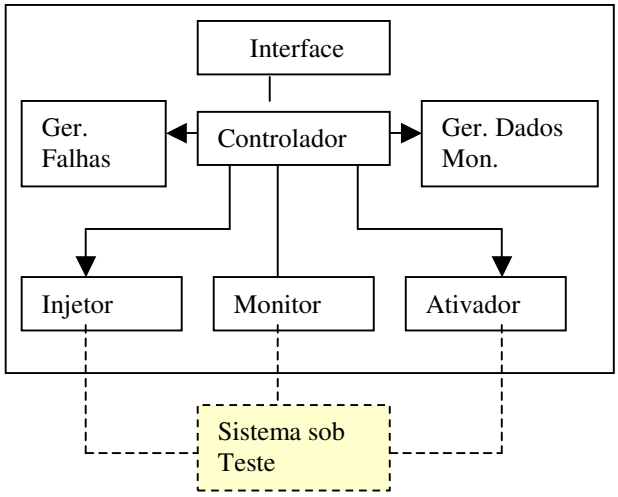
Neste diagrama tem-se um injetor de falhas que tenta simular a presença de falhas no sistema sob teste, e estas falhas são armazenadas na biblioteca de falhas. O monitor observa o comportamento do sistema alvo, o gerador de carga de trabalho passa os comandos para o sistema sob teste, de acordo com a biblioteca de carga de trabalho. Dados sobre o sistema são recolhidos pelo coletor para uma análise posterior, e o controlador coordena a execução de todo o sistema. Em [6] a arquitetura é descrita, mas não é discutido como é a dinâmica de execução dessa arquitetura, nem como esses componentes se relacionam. Essas informações podem ser vistas no padrão de arquitetura "Injetor de Falhas" apresentadas a seguir.

2.5. Sistema de Padrões para Injeção de Falhas por Software

O sistema de padrões visa facilitar o desenvolvimento de novos programas de injeção de falhas. Eles foram desenvolvidos baseado no estudo de diversas ferramentas de Injeção de falhas por software [10]. A seguir serão apresentados os três padrões.

2.5.1. Injetor de Falhas

É um padrão de arquitetura que visa solucionar o problema de como arquitetar um programa para fazer Injeção de Falhas. Quais seriam seus componentes, como se relacionariam e como seriam suas interfaces.



"Figura 2. Estrutura do padrão de arquitetura injetor de falhas"

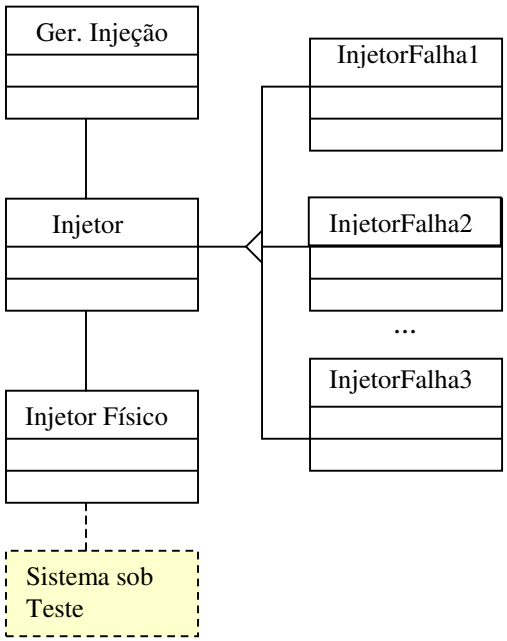
O padrão de arquitetura de injeção de falhas, foi estruturada em 7 subsistemas:

- *Ativador:* ativa a execução do sistema alvo para que ele possa ser testado em suas condições de funcionamento normais;
- *Injetor:* realiza a injeção propriamente dita das falhas dentro do sistema sob testes;
- *Monitor:* faz a monitoração do sistema alvo, para verificar se ele opera como o esperado;
- *Controlador:* controla os subsistemas acima, para que realizem suas atividades coordenadamente;
- *Interface com o usuário:* recebe as especificações do usuário para realização do experimento e devolve os resultados;
- *Gerenciador de falhas:* repositório de falhas que armazena as falhas a serem injetadas;
- *Gerenciador de dados monitorados:* armazena os resultados recebidos da monitoração do sistema sob teste.

2.5.2. Injetor

O injetor é um padrão de projeto que sugere uma estrutura de objetos que se comunicam com o sistema sob teste e simulam uma determinada falha dentro dele. Esta estrutura tem três componentes:

- Gerenciador de Injeção: esse componente controla o processo de injeção em si. Ele instancia os injetores, descritos a seguir, de acordo com o tipo de falha injetada.
- Injetor: é instanciado pelo gerenciador de injeção para cuidar da injeção de falha em específico. É definida uma classe abstrata, injetor, e a partir dessa classe, são criadas subclasses concretas que implementam tipos de falhas diferentes. Porém ele não se comunica diretamente com o sistema sob teste, isto é realizado através do injetor físico.
- Injetor físico: é o componente que se comunica diretamente com o sistema sob teste através de primitivas para comunicação.



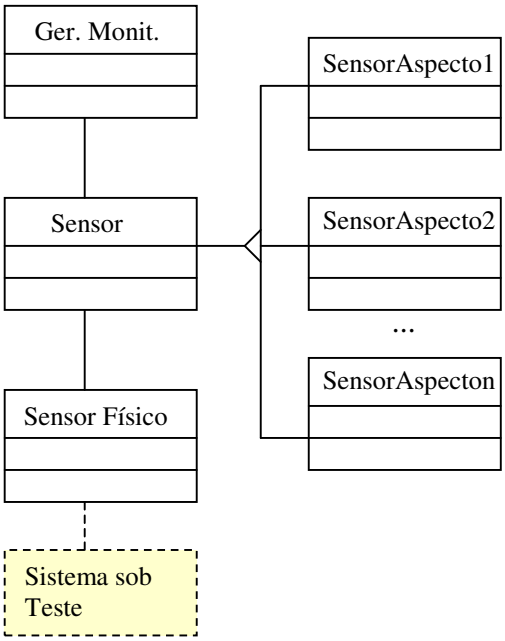
"Figura 3. Estrutura do padrão injetor"

2.5.3. Monitor

O monitor é também um padrão de projeto que procura fornecer uma solução para o problema de como criar uma estrutura que faça a monitoração do sistema sob teste. A estrutura apresenta três componentes:

- Gerenciador de Monitoração: este componente coordenaria o processo de monitoração.Para cada aspecto do sistema alvo, ele instancia um sensor para cada tipo de dado a ser obtido.
- Sensor: objeto encarregado de monitorar um determinado aspecto do sistema sob teste. É definida uma classe abstrata, sensor, que define uma interface comum para cada tipo de sensor. A partir dessa classe, são criadas as subclasses concretas que especificam sensores para cada tipo de aspecto monitorado.

- Sensor físico: objeto que fazem a comunicação dos sensores com o sistema sob teste.



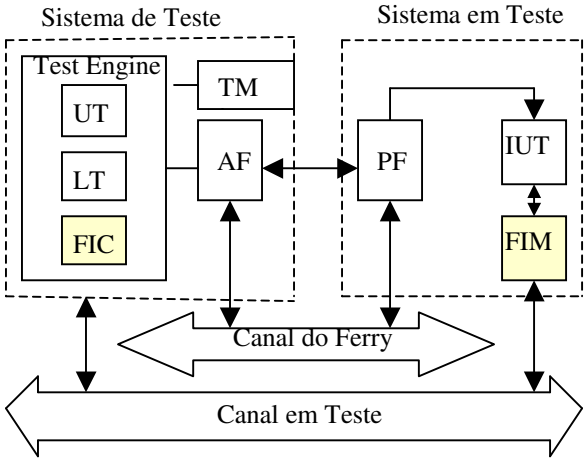
"Figura 4. Estrutura do padrão de projeto monitor"

3. Arquitetura FSoFIST

O protótipo da ferramenta FSoFIST foi projetado para dar suporte à realização de testes em protocolos de comunicação e foi desenvolvido através da extensão da arquitetura *Ferry clip* [7]. A idéia desta extensão é permitir a injeção de falhas em protocolos que possuam mecanismos de recuperação e tolerância a falhas. Esta modificação consistiu em acrescentar um injetor de falhas junto à IUT e um mecanismo simples de controle deste injetor junto ao Sistema de Teste. A Figura 5 apresenta a arquitetura da FSoFIST.

O Sistema de Teste é formado pelo AF, responsável por iniciar a conexão com o PF e por transferir os dados dos testadores para o sistema em teste, pelo *Test Engine* que agrupa as funções do Testador Superior e do Testador Inferior e o Fault Injection Controller (FIC) que implementa as funções de injeção que são independentes da IUT, suas funções incluem determinar o momento em que uma falha deve ser injetada e por quanto tempo, com base nas informações fornecidas no script de testes e pelo Test Manager (TM) que é responsável por iniciar e terminar uma sessão de testes.

O Sistema em Teste é formado pelo PF, quem efetivamente transfere os dados enviados para a IUT, pela Implementação em Teste (IUT) e pelo Fault Injection Module (FIM) que injeta falhas diretamente nas mensagens que circulam através dele, sendo responsável por três atividades: filtragem (interceptação de mensagens), injeção (injeção de falha em si) e entrega (devolução da mensagem, alterada ou não, ao sistema).



"Figura 5. Arquitetura da FSoFIST

4. Utilização de Padrões de Projeto na arquitetura FSoFIST

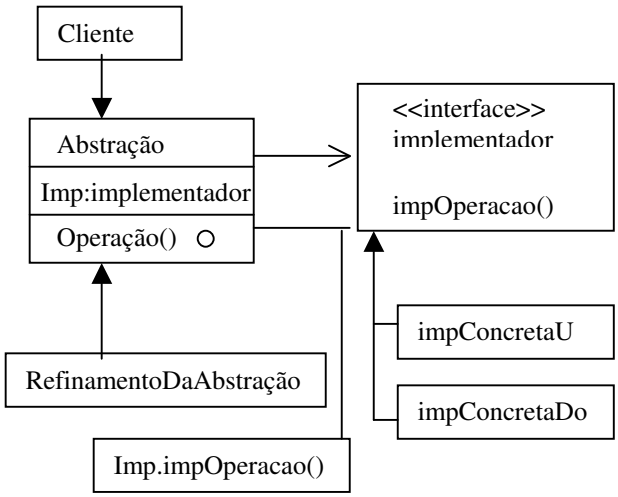
Baseado nos padrões de software apresentados na Seção 2 pode-se fazer o mapeamento deles à arquitetura da FSoFIST.

" Tabela 1. Mapeamento dos componentes"

Padrão	Componente no padrão	Arquitet. FSoFIST
Injetor de falhas	Controlador	Test Manager
Injetor de falhas	Ger. de Falhas	Script de entrada
Injetor de falhas	Ger. de dados monitorados	Log de saída
Injetor	Injetor e Ger. de injeção	FIC
Injetor	Injetor Físico	FIM
Monitor	Gerenciador de Mon., sensor.	
Monitor	sensor físico	Parte do PF
Ativador	Ger. De ativação e ativador	Test Engine
Ativador	Ativador Físico	Parte do PF (não é injeção de falhas)
Bridge	Interface	Interface

O padrão "Ativador" não foi definido junto com os padrões de projeto "Injetor" e "Monitor", mas será definido para este trabalho com o objetivo de deixar a ferramenta FSoFIST totalmente modular.

Para o padrão de interface será adotado o padrão "Bridge" que tem como função desacoplar uma abstração de interface de sua implementação para que ambos possam variar independentemente [11].



"Figura 6. Estrutura do padrão bridge"

O padrão "Bridge" é usado quando tem-se necessidade de evitar uma forte dependência entre a interface e a implementação.

5. Conclusão

O artigo descreveu brevemente o conceito de injeção de falhas, que é uma técnica utilizada em testes. A falha é injetada em um sistema e é feita a observação do funcionamento do sistema para verificar se o mesmo continua operando conforme o esperado. Existem três tipos de injeção de falhas: injeção por simulação, por hardware e por software.

Diante de várias ferramentas para injeção de falhas que já foram estudadas e desenvolvidas anteriormente [10], uma arquitetura genérica para injeção de falhas por software foi criada [6] e um sistema de padrões para injeção de falhas por software também [10], visando facilitar o desenvolvimento de novas ferramentas para Injeção de Falhas.

Este sistema de padrões apresentou três padrões: injetor de falhas, injetor e monitor. Com o objetivo de deixar a ferramenta FSoFIST modular um mapeamento desses padrões para a arquitetura da FSoFIST está sendo feito, esse trabalho irá facilitar no desenvolvimento dessa ferramenta e auxiliar na definição da arquitetura de testes para validação de Computadores de Bordo.

Foi apresentado também o padrão "Bridge" que é um padrão de interface utilizado para desacoplar a implementação da interface, podendo estas, agirem de forma independente.

A definição do padrão "Ativador" e um padrão para comunicação via LMAP, protocolo utilizado no canal do ferry, serão os próximos trabalhos.

Referências

[1] Martins, E. "ATIFS: um Ambiente de Testes baseado em Injeção de Falhas por Software" – Relatório Técnico DCC-95-24 – UNICAMP, dez 1995.

[2] Araújo, M. R. R. "fsofist – Uma ferramenta para teste de protocolos tolerantes a falhas" – Dissertação de Mestrado Instituto da Computação – Unicamp, out 2000.

[3] Clark, Jeffery; Pradhan, Dhiraj. "Fault Injection: A method for Validating Computer-System Dependability". IEEE Computer, Junho/1995, páginas 47-56.

[4] Avizienis, A. ;Rennels, D. "Fault-Tolerance Experiments with the JPL STAR computer". Proc. COMPCON ´72, páginas321-324.

[5] Gunnelo, U.;Karlsson, J.;Torire, J."Evaluation of Error Detection Schemes using Fault Injetion by Heavy-Ion Radiation". Proc. FTCS-19, Junho/1989, páginas 340-347.

[6] Hsueh, Mei-Chen; Tsai, Timothy; Iyer, Ravishankar. "Fault Injection Techniques and Tools". IEEE Computer, Abril/1997, páginas 75-82.

[7] Chanson, S. T; Voung, S.; Dany, H. "Multi-party and interoperability testing using the Ferry Clip approach", Computer communications vol 15, no 3, April 1992.

[8] Martins, E.; Mattiello-Francisco, M.F. A Tool for Fault Injection and Conformance Testing of Distributed Systems. Latin-American Dependable Computing Symposim, 1. São Paulo, Brasil, october 2003. In: **Lecture Notes in Computer Science**. Springer Verlag, p. 282-302, 2003b.

[9] Martins, E.; Mattiello-Francisco, M.F.; Morais, A.N.P., "Uso da ferramenta de testes FSOFIST na validação de uma aplicação espacial.
In: Anais da 2ª. Jornada Ibero-Americana de Engenharia de Software e
Engenharia de Conhecimento, Salvador, BA, Brasil, 2002.

[10] Leme, N. G. M.; "Um Sistema de Padrões para injeção de Falhas por Software" , Dissertação de Mestrado – Unicamp – Agosto 2001.

[11] Gamma, E., Johnson, Johnson R.; Helm, R., Vlissides, J.; *"Design Patterns: Elements of Reusable Object-Oriented Software"*. Addison Wesley, 1994.