Teste de protocolos tolerantes a falhas em presença de falhas*

Eliane Martins
LAC-INPE
Avenida dos Astronautas 1758
12201 S.José dos Campos - SP.

RESUMO

Um protocolo tolerante à falhas deve continuar a fornecer o serviço especificado mesmo em presença de um certo número de falhas em processadores ou no meio de comunicação. Uma das dificuldades no desenvolvimento de tais protocolos diz respeito à sua validação, pois deve ser considerado neste caso não somente o seu comportamento em situações normais, mas também em presença de falhas. Este artigo aborda o problema da validação de protocolos tolerantes à falhas em presença de entradas para as quais eles foram projetados e implementados para tratar: as falhas. Mais especificamente, o método usado é baseado em *injeção física de falhas*, que consiste em aplicar falhas físicas diretamente aos pinos dos circuitos integrados que compõem um protótipo do hardware e do software do sistema a ser validado. Sendo portanto aplicável nas fases finais do ciclo de desenvolvimento do sistema, a injeção física de falhas pode ser usada para complementar os testes de uma implementação de um protocolo tolerante a falhas. Os problemas comuns ao teste de implementação de protocolos, quais sejam: a definição de uma arquitetura de testes, a geração das entradas de teste e a análise dos resultados são abordados no texto. A aplicação da metodologia proposta na validação da implementação de um protocolo de difusão atômica, desenvolvido no âmbito do projeto Delta-4, do programa europeu ESPRIT, é também apresentada. Os resultados obtidos serviram para mostrar não somente a viabilidade do método, mas também a sua utilidade na complementação de outros métodos de validação.

ABSTRACT

A fault-tolerant protocol must continue to provide specified services even in the presence of a certain number of processor or communication faults. One of the dificulties in the development of such protocols concerns its validation: one must consider its behavior not only in normal situations, but also in the presence of faults. This paper addresses the problem of the use of fault injection for validating fault-tolerant protocols with respect to inputs they have been developped to deal with: the faults. More specifically, the method used is physical fault injection, where physical faults are applied to the pins of the integrated circuits that compose a hardware and/or software prototype of the system to be validated. Being applied only at final stages of system development, *physical fault injection* can be used to complement the tests of a fault-tolerant protocol implementation. The problems common to the test of a protocol implementation are addressed in the paper, that are: definition of a test architecture, generation of test inputs and analysis of the results obtained. The proposed methodology has been applied to the

---

* Este estudo foi desenvolvido junto ao Laboratoire d'Automatique et d'Analyse de Systèmes (LAAS) do CNRS em Toulouse, França, tendo sido parcialmente financiado pelo CNPq. Este trabalho fez parte do projeto Delta-4, do programa europeu ESPRIT, o qual foi financiado parcialmente pela Comissão da Comunidade Européia.

validation of an atomic multicast protocol implementation, developped in the framework of the ESPRIT Delta-4 project. The results obtained showed not only the feasibility of the method but also its usefulness in complement to other validation methods.

I. INTRODUÇÃO

Os mecanismos e algoritmos usados na construção de sistemas distribuídos tolerantes à falhas acidentais de hardware (falhas fisicas) são baseados na *replicação* de componentes de software em processadores distintos e fisicamente independentes. O uso de replicação requer um serviço de comunicação de grupo confiável, que garanta, entre outras, a consistência entre as réplicas, mesmo em presença de falhas de processadores ou no meio de comunicação. Existe um grande número de protocolos propostos para a comunicação entre grupos de réplicas [Chang 84, Cooper 85, Birmann 87, Verissimo 89, entre outros]. Uma grande dificuldade no desenvolvimento de tal tipo de protocolo diz respeito à validação dos mesmos, pois além das funcionalidades realizadas em situações normais deve-se também considerar aquelas realizadas em presença de falhas de componentes do sistema.

A **validação** é um conjunto de atividades que permite obter confiança na capacidade de um sistema em fornecer o serviço especificado, e compreende tanto a **eliminação de falhas** (visando a reduzir, por **verificação**, as falhas de concepção/implementação existentes no sistema) quanto a **previsão de falhas** (visando a obter, por **avaliação**, medidas que permitam caracterizar o comportamento do sistema em presença de falhas) [Laprie 92].

O uso de técnicas formais de verificação (como prova de programas), é altamente desejável para garantir a correção de tal tipo de protocolo, assim como o uso de métodos analiticos (modelos de Markov, ...) para a avaliação oferecem medidas mais exatas dos parâmetros desejados. No entanto, devido à complexidade de tais tipos de protocolos, elas são limitadas a um pequeno número de nós, ou a partes do protocolo ou ainda a um número reduzido de falhas. Por isto é comum recorrer-se a técnicas experimentais em complemento a estes métodos.

Este trabalho aborda o problema do uso de **injeção de falhas** para a validação experimental de protocolos tolerantes a falhas em presença de entradas particulares para as quais eles forma desenvolvidos para tratar: **as falhas**. Primeiramente é feita uma breve descrição do método de injeção de falhas. Em seguida é apresentada a metodologia proposta para a validação de protocolos tolerantes a falhas utilizando o método descrito anteriormente. E por último é apresentado um exemplo de aplicação da metodologia proposta na validação de um protocolo de difusão atômica desenvolvido no âmbito do projeto Delta-4, do programa europeu ESPRIT.

II. O METODO DE INJEÇÃO DE FALHAS

O método de injeção de falhas consiste na introdução deliberada de falhas no sistema. Acelerando a ocorrência de erros e defeitos[1], a técnica de injeção de falhas permite validar os mecanismos de tolerância a falhas (MTF) de um sistema em presença de entradas particulares: *as falhas*.

---

[1] Uma *falha* é a causa direta de um erro. Um *erro* é então a manifestação de uma falha no sistema, e um *defeito* é a manifestação do erro no serviço fornecido pelo sistema [Laprie 92].

A injeção de falhas contribui para os dois aspectos da validação, quais sejam, a **eliminação e a previsão de falhas**:

- no caso da eliminação de falhas o objetivo é a **verificação** dos MTF com relação às hipóteses de falhas consideradas;
- no caso da previsão de falhas, o objetivo é de se fazer uma **avaliação** da *eficiência* dos MTF em presença de falhas passíveis de ocorrer na fase operacional do sistema.

A injeção de falhas é caracterizada por um *domínio de entrada* e um *domínio de saída* [Arlat 90]. O *domínio de entrada* corresponde a um conjunto de falhas a injetar **F** e um conjunto de ativações **A** que especifica as entradas destinadas a exercitar o sistema e, em consequência, ativar as falhas injetadas. O *domínio de saída* é caracterizado por um conjunto **R** de dados coletados e por um conjunto de medidas **M** derivadas da análise e tratamento dos conjuntos **F**, **A** e **R**. Os conjuntos **FARM** constituem então os principais atributos da injeção de falhas.

Na prática a validação por injeção de falhas é baseada na execução de uma série de **experiências**, cada qual sendo caracterizada por um ponto no espaço **F×A×R**. O número de experiências a serem realizadas depende do objetivo da validação (verificação ou avaliação). As experiências necessárias para a realização de um determinado objetivo, consistindo na injeção de várias falhas, constituem uma **campanha de injeções**.

Existem diferentes formas de aplicação da injeção de falhas, as quais dependem do tipo de falhas consideradas (físicas ou lógicas) e do nível de abstração utilizado para representar o sistema a ser validado (modelo empírico ou protótipo físico) [Arlat 90a]. A técnica de injeção de falhas considerada aqui é denominada **injeção física de falhas**, na qual falhas físicas (bit preso em 0/1, ...) são diretamente aplicadas aos pinos dos circuitos integrados que compõem o sistema a ser validado.

A injeção física de falhas vem sendo largamente utilizada, desde a validação de circuitos [Kurlak 81] à validação de sistemas completos [Crouzet 82, Lala 83, Damm 88, Arlat 90b, ...]. Apesar desta forma de injeção ser aplicável unicamente nas fases finais do desenvolvimento de um sistema, quando já se dispõe de um protótipo do mesmo, ela apresenta duas principais vantagens, quais sejam: (i). permitir a validação de um protótipo que é próximo do produto final e (ii). permitir a validação global do sistema, integrando tanto o hardware quanto o software.

A metodologia apresentada a seguir trata da caracterização dos atributos **FARM** para a validação da implementação de protocolos usando injeção física de falhas.

III. O USO DE INJEÇÃO FÍSICA DE FALHAS NA VALIDAÇÃO DE PROTOCOLOS TOLERANTES A FALHAS

A técnica de injeção física de falhas sendo aplicável quando já se dispõe de um protótipo físico do sistema, pode ser então utilizada como complementar aos testes de implementação de protocolos de comunicação. Nesta seção é feito um paralelo entre o contexto normativo (trata do teste de uma implementação protocolos de comunicação do modelo de referência OSI da ISO) e o contexto experimental no qual se situa este trabalho.

A ISO e a CCITT desenvolveram uma norma -ISO 9646- para a aplicação dos testes de uma implementação de protocolos de comunicação, mais precisamente para os chamados *testes de conformidade*. Os *testes de conformidade* de uma implementação de um protocolo de comunicação visam a determinar se ela satisfaz à especificação do protocolo. Dentre os aspectos abordados pela norma para a realização desses testes, podem ser citados: a descrição das arquiteturas de teste possíveis, a estruturação dos testes e a definição de uma linguagem para a descrição dos mesmos [ISO9646-1 a ISO9646-4]. Outros pontos importantes no teste de implementação de protocolos, que não são abordados diretamente pela norma, dizem respeito à geração das entradas utilizadas para os testes e à análise dos resultados. A definição de uma arquitetura de testes, bem como a geração dos testes e a análise dos resultados são tratados mais em detalhes a seguir.

III.1. Arquitetura de teste

A norma ISO 9646 identifica diferentes arquiteturas para o teste *isolado* de uma implementação de protocolo [ISO9646-1]. Nestas arquiteturas, a implementação em teste (**IUT**, de "Implementation Under Test") é controlada e observada por um *sistema de teste* a partir de suas interfaces de serviço superior e inferior, também denominadas de pontos de acesso aos serviços (**SAP**, de "Service Access Point"), mostradas na Figura 1 como USAP e LSAP, respectivamente. A entidade par desta implementação é "simulada" pelo sistema de teste.

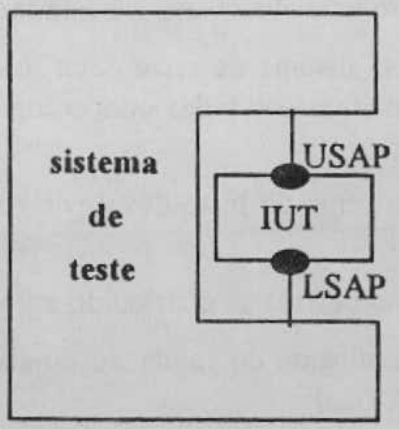

Figura 1. Esquema geral dos testes no contexto normativo.

No caso de protocolos tolerantes à falhas, o teste em isolado de uma implementação não é suficiente para determinar a sua conformidade, pois propriedades do serviço fornecido por estes protocolos dependem da cooperação entre várias entidades, e também das hipóteses sobre os modos de falhas dos processadores e dos meios de comunicação [Cristian 91].

Do exposto acima, temos que o ambiente de testes para protocolos tolerantes a falhas deve incluir diversas implementações do protocolo, e também o hardware onde elas devem residir, bem como o meio de comunicação. Assim sendo, a arquitetura de testes considerada neste estudo é baseada em um banco de testes distribuído [Berg 82], constituído basicamente por:

- um **sistema alvo**, contendo o hardware e o software do sistema a ser validado. Este sistema é composto por diversos sítios interligados por um meio de comunicação;
- um **sistema de teste**, que compreende o hardware e o software necessários para a realização dos testes.

Na concepção deste banco de testes, deve-se ter em conta os seguintes aspectos:

- *suporte para injeção física de falhas*: o sistema de teste deve incluir uma ferramenta especial - um injetor de falhas, capaz de injetar falhas físicas diretamente ao nível dos pinos dos circuitos que compõem o sistema;
- *flexibilidade*: o banco de testes deve ser capaz de se adaptar à diferentes arquiteturas tanto de hardware quanto de software;
- *transparência*: deve-se evitar ao máximo que haja interferência das atividades de validação sobre o funcionamento do sistema a ser validado;
- *automação*: afim de permitir a realização de um grande número de testes, é interessante reduzir as intervenções humanas a um mínimo indispensável;
- *controle e observação*: os mecanismos de tolerância a falhas podendo ser implementados tanto a nível do hardware quanto do software, deve ser possível controlar e observar o sistema tanto em um nível quanto no outro;
- *teste global*: o sistema de teste deve ter a capacidade de realizar um teste integrando diversas entidades cooperantes, residentes em sítios distintos do sistema;
- *exatidão*: o sistema de teste deve evitar a introdução de erros nas medidas obtidas.

O uso de um banco de testes distribuído apresenta, entre outras vantagens:

- fornecer um ambiente de validação do sistema similar ao que o sistema terá na fase operacional;
- permitir uma validação global do sistema, integrando o hardware e o software;
- permitir que sejam tratados os aspectos tanto de verificação quanto de avaliação do sistema.

A apresentação de um banco de testes distribuído utilizado na validação de um sistema aberto real é dada na seção IV.

III.2. Geração da sequencia de teste

A geração de testes consiste na obtenção das entradas utilizadas para estimular o sistema a ser validado. Os testes podem ser classificados de acordo com o critério utilizado para selecionar estas entradas e o modo pelo qual elas são geradas [Laprie 92]. A maneira pela qual as entradas podem ser selecionadas depende de diversos critérios, entre eles, se a estrutura da implementação é ou não considerada. Caso ela o seja, temos um *teste estrutural*; em caso contrário, a implementação é considerada como uma "caixa preta" que pode ser observada unicamente do exterior - tem-se então um *teste funcional*.

No contexto normativo geralmente é utilizado um teste funcional, a seleção dos testes sendo baseada na especificação do protocolo. As diferentes técnicas existentes se distinguem pelo formalismo utilizado para a especificação do protocolo. Temos assim diversas técnicas baseadas em especificações na forma de máquinas de estado finitas [Wang 87], ou redes de Petri e suas variantes [Baumgarten 89], ou ainda baseadas em técnicas de descrição formais (Estelle, Lotos ou SDL) [Bochman 88].

Qualquer que seja o critério adotado (funcional ou estrutural), a geração dos testes pode ser *determinística* ou *estatística* (também chamada de *aleatória*) [Laprie 92]. Em um *teste determinístico*, as entradas são escolhidas com base em uma análise detalhada das funções do sistema definidas na sua especificação ou da estrutura do programa que o implementa, de modo a cobrir o máximo possível do critério adotado. Em um *teste estatístico*, as entradas são escolhidas segundo uma distribuição de probabilidades associada ao domínio de entrada.

Conforme foi apresentado na seção II, o domínio de entrada no caso de um teste por injeção de falhas é constituído pelas falhas a serem injetadas (**F**) e pelos modos de ativação do sistema (**A**). No que concerne a escolha destes últimos considera se a realização de testes funcionais, como no contexto normativo. Para a escolha das falhas, no entanto, é necessário um mínimo de conhecimento da estrutura física do sistema. Uma vez determinado o critério de seleção das entradas <f,a> que irão estimular o sistema, e mais precisamente, seus M.T.F., resta determinar como vão ser geradas estas entradas. Esta geração é largamente dependente dos objetivos da validação, como mostra a Tabela I [Arlat 90a]:

Tabela I. Os conjuntos **F** e **A** e os objetivos da validação.

	verificação	**avaliação**
objetivo	revelar o maior número possível de falhas de concepção/implementação do sistema.	obter medidas da eficiência dos M.T.F. do sistema.
F	pequeno número de falhas específicas determinadas a partir das hipóteses feitas na fase de projeto do sistema.	número elevado de falhas representativas das falhas passíveis de ocorrer em fase operacional.
A	entradas destinadas a exercitar as funções do sistema e ativar as falhas injetadas.	entradas representativas do perfil operacional do sistema.

Do exposto na Tabela I pode-se concluir que, para a verificação, o ideal seria a realização de testes determinísticos, onde uma análise detalhada dos M.T.F. do sistema permitiria escolher os pares **<f,a>** que garantissem a sensibilização destes mecanismos, para que eles sejam exercitados ao máximo. Tal análise é no entanto impossível de ser realizada na prática, especialmente quando o sistema em teste é considerado como uma "caixa preta", onde os M.T.F. não são sequer identificáveis. Além disso os testes determinísticos não são os mais adequados para a avaliação, pois a escolha subjetiva das entradas pode levar à obtenção de estimativas tendenciosas dos parâmetros desejados.

Assim sendo, a realização de testes estatísticos é a alternativa adotada. Este tipo de teste é utilizado tanto para o teste do hardware quanto para o teste de programas (c.f. [Thévenod 91]), e uma de suas principais vantagens é permitir integrar tanto a verificação quanto a avaliação. No entanto, sua realização apresenta duas dificuldades principais :

1. atribuição da distribuição de probabilidades ao domínio de entrada,
2. determinação do tamanho da sequencia de testes.

No que diz respeito ao ponto (2), o teste estatístico requer a realização de uma grande quantidade de testes, pois por um lado, é necessário que todas as entradas tenham probabilidade não nula de serem selecionadas, e por outro lado, quanto mais testes forem realizados, maior o nível de confiança que se pode ter nas estimativas obtidas.

A escolha da distribuição de probabilidades a atribuir ao domínio de entrada é importante, pois ela também influi sobre o número de testes a serem realizados. O ideal para a avaliação é que cada par <**f**,**a**> seja escolhido segundo a probabilidade de ocorrência que eles terão em fase operacional. Desta forma, as estimativas obtidas serão representativas das medidas reais. A determinação das probabilidades de ocorrência associadas aos pares <**f**,**a**> é no entanto uma tarefa complexa. Na prática a escolha desses pares pode ser feita da seguinte forma:

- escolha de uma ativação **a** segundo a distribuição associada ao conjunto **A**,
- aplicação de **a** ao sistema alvo,
- escolha de uma falha **f** segundo a distribuição associada ao conjunto **F**,
- aplicação de **f** ao sistema alvo de forma a que haja um retardo aleatório Δt entre a aplicação de **a** ao sistema e a injeção de **f**.

No que diz respeito à escolha da distribuição associada ao conjunto **A**, duas alternativas comumente utilizadas no teste de programas são [Thévenod 91]: o uso de uma distribuição uniforme ou o perfil operacional do programa, definido para um tipo de usuário. Outras distribuições foram propostas; especificamente no caso do teste de implementação de protocolos de comunicação, o estudo apresentado em [Sidhu 89] trata da seleção aleatória de testes baseada na distribuição de probabilidades associada aos estados de uma máquina de estados finita.

No que diz respeito à escolha da distribuição associada ao conjunto **F**, o que se faz comumente nos testes por injeção fisica de falhas é obter a probabilidade de ocorrência de uma falha **f** com base nas informações sobre a taxa de falhas dos circuitos integrados que compõem o sistema [Kurlak 81, Hummel 88]. Uma outra alternativa consiste em caracterizar uma falha segundo uma série de atributos (modelo, localização, etc); desta forma a probabilidade de ocorrência de uma falha pode ser decomposta no produto das probabilidades associadas a cada um de seus atributos. Esta alternativa é descrita em [Martins 92], e um exemplo de aplicação é dado na seção IV.

III.3. Análise dos resultados

Um dos problemas do teste é a determinação de uma referência a partir da qual seja possível determinar se a saída observada é válida ou não. É o conhecido "problema do oráculo".

No contexto normativo, a referência utilizada é a especificação do protocolo, a qual considera-se como já tendo sido submetida a processo de validação. Distinguem-se nesse caso duas técnicas para a análise dos resultados [Bochman 89]:

- *análise integrada aos testes*, na qual a sequência de testes contém, para cada entrada, qual a saída esperada;
- *análise separada dos testes*, na qual o comportamento observado da IUT é comparado à especificação do protocolo. Esta comparação pode ser feita em linha, executando-se a especificação e a IUT em paralelo [Ayache 79, Molva 86], ou após os testes, através de análise do traço de execução [Dssouli 86, Bochman 89].

Conforme foi visto na seção II, no caso da injeção de falhas o domínio de saída é caracterizado pelos conjuntos **R** (dados coletados) e **M** (medidas obtidas). Como no caso do domínio de entrada, as características destes conjuntos também dependem dos objetivos da validação, como mostra a Tabela II.

Tabela II. Características do domínio de saída segundo os objetivos da validação.

	verificação	avaliação
R	indicação de ocorrência de eventos (e/ou medidas de tempo associadas) + informações para o diagnóstico.	indicação de ocorrência de eventos e medidas de tempo associadas.
M	veredito: valor binário indicando se o sistema se comporta ou não conforme o especificado.	estatísticas sobre a ocorrência de eventos e do intervalo de tempo entre as ocorrências.

Segundo essa tabela, no caso da validação de protocolos por injeção falhas, a análise dos resultados comporta dois aspectos:

a. determinar se o protocolo se comporta conforme foi especificado,
b. tratar os dados coletados durante as experiências afim de obter as medidas referentes à eficiência dos M.T.F., o que inclui a estimação de parâmetros tais como: o *fator de cobertura* [Bouricius 69], definida como a probabilidade condicional de que o sistema continua a fornecer o serviço correto dada a existência de uma falha; ou a *latência de erros*, que é o intervalo de tempo entre a ativação de um erro e a sua detecção.

Neste texto serão abordados com mais detalhes unicamente os aspectos relativos ao ponto (a) acima. Os estudos apresentados em [Arlat 90b, Martins 92] tratam em detalhes dos aspectos referentes à estimação de parâmetros, notadamente do fator de cobertura dos M.T.F.

Primeiramente, cabe determinar qual a referência a ser utilizada. No contexto normativo, a referência é a especificação do protocolo. No contexto experimental de validação utilizando injeção de falhas, interessa determinar se o sistema continua a fornecer o serviço especificado em presença de falhas; assim sendo, será considerada como referência a *especificação do serviço*. Esta especificação pode ser dada na forma de predicados caracterizando as propriedades locais e globais do serviço fornecido pelo protocolo [Bochman 80]. Nosso enfoque neste artigo é na verificação das propriedades globais do serviço oferecido por um protocolo tolerante à falhas.

Para determinar se os resultados são válidos, foi visto acima que há duas alternativas. Na primeira, a análise é integrada à sequencia de testes, o que requer que sejam conhecidas previamente as respostas do sistema a cada entrada fornecida. Dada a natureza não determinística dos testes por injeção de falhas (entre outras razões, porque

as entradas são escolhidas de maneira aleatória), essa alternativa não é viável. Por outro lado, a realização de uma análise separada, em linha, tem como um inconveniente o fato de que a duração das experiências seria aumentado devido à análise, o que pode vir a limitar o número de testes a serem realizados.

Considera-se então neste trabalho que a análise é realizada após o término da experiência, e consiste em comparar as interações de entrada e saída observadas, também chamada de *traço* [Dssouli 86], com a especificação do serviço (na forma de propriedades globais do mesmo), conforme é ilustrado na Figura 2.

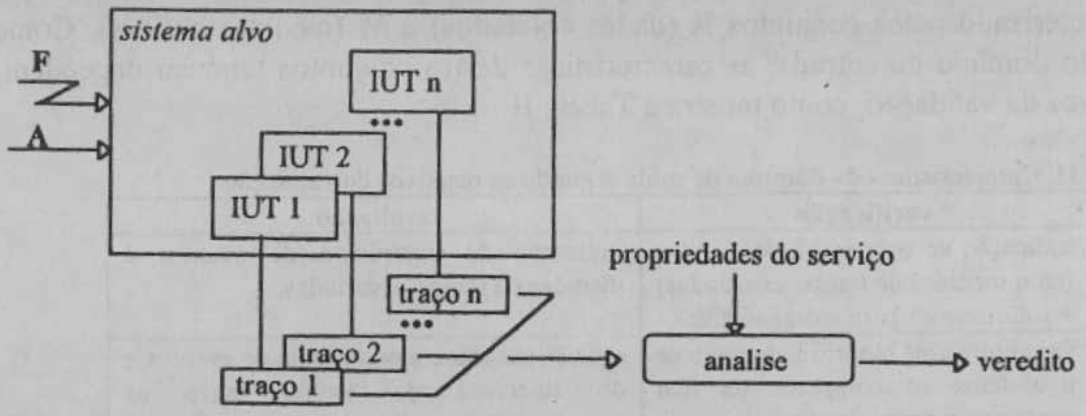

Figura 2. Análise dos traços de execução.

A comparação pode ser realizada por um módulo *analisador de traços*, residente fora do sistema alvo. Um ponto importante a considerar é a obtenção deste analisador, de preferência a partir da especificação de referência [Bochmann 89]. Esta especificação sendo dada na forma de predicados caracterizando as propriedades globais do serviço, o analisador de traços, na sua forma mais simples, pode ser definido como um algoritmo representando estas propriedades. Este algoritmo pode ser obtido mesmo a partir da descrição informal das mesmas, como ilustra o exemplo a seguir.

Exemplo. Suponhamos que um protocolo de difusão atômica é implementado para uma rede composta de n estações. Seja S_i uma estação e T_i o traço observado para esta estação. Considera-se também uma variável *corretas*, designando o conjunto de estações que não apresentem defeito após os testes de injeção de falhas. Seja a seguinte propriedade global do serviço desse protocolo:

P: *Se uma implementação do protocolo que reside em uma estação correta entrega a mensagem m a seu usuário local, então todas as outras estações corretas devem entregar esta mensagem a seus usuários.*

O algoritmo correspondente à análise dessa propriedade pode ser escrito como:

```
P=verdade;
para todo par Si, Sk ∈ corretas
faça
  se existe m ∈ Ti tal que m ∉ Tk
  então          /* erro: Si e Sk não entregam as mesmas mensagens */
       P=falso;
       abandonar a análise;
  fim;
fim;
```

Se as propriedades forem expressas em algum outro tipo de formalismo, um tradutor pode ser utilizado para a obtenção do algoritmo do analisador, como por exemplo em [Groz 89], onde é a mostrada a obtenção de um analisador dos traços (gerados a partir de uma simulação) em que as propriedades podem ser expressas através de fórmulas de lógicas temporais lineares.

IV. Aplicação: o teste de um protocolo de difusão atômica

Esta seção trata da aplicação da metodologia descrita anteriormente na validação da implementação de um protocolo de difusão atômica, desenvolvido no projeto Delta-4 [Powell 91] do programa europeu ESPRIT.

IV.1. O sistema alvo

O objetivo do projeto Delta-4 é desenvolvimento de uma arquitetura aberta, distribuída e com segurança de funcionamento. A arquitetura Delta-4 visa tolerar principalmente as falhas acidentais de hardware (falhas fisicas). O princípio básico da tolerância a falhas é a replicação de componentes de software (aplicações do usuário) em hospedeiros distintos interconectados por uma rede local. O sistema de comunicação de Delta-4, **MCS** (de "Multicast Communication System"), provê serviços multiponto permitindo a comunicação confiável entre grupos de réplicas. A base para este serviço de comunicação de grupos confiável é um protocolo de difusão atômica (**AMP**, de "Atomic Multicast Protocol"). Na versão testada, o AMP foi implementado como uma extensão da camada **MAC** ("Medium Access Control") para uma rede local em anel com passagem de permissão ("token ring"), compatível com a norma ISO 8802.5.

O MCS reside em servidores de comunicação específicos do projeto Delta-4 denominados NAC (de "Network Access Controller"). Os NACs possuem mecanismos de auto-teste implementados a nível do hardware que permitem garantir que eles são do tipo falha-parada ("fail-silent"), i.e., a existência de uma falha se caracteriza pela parada de todo o envio de mensagens. Quando um erro é detectado, os mecanismos de auto-teste do NAC acionam um relé, fazendo com que o NAC seja extraído do anel.

Os mecanismos de extração do NAC e as propriedades do serviço de difusão atômica fornecido pelo AMP formam o suporte de base da tolerância a falhas da arquitetura Delta-4. A descrição destas propriedades, em linguagem natural, são mostradas, resumidamente, na Tabela III. Somente as propriedades consideradas para os testes estão aí descritas, a totalidade delas estando apresentadas em [Powell 91].

O objetivo das experiências de injeção de falhas foi duplo:

- **avaliar** dois níveis de cobertura fornecidos pelos mecanismos de tolerância a falhas: a) *cobertura local* dos mecanismos de auto-teste dos NACs e b) *cobertura global*, que inclui os mecanismos de tolerância a falhas do NAC e do AMP;
- **testar**, em presença de falhas, o serviço fornecido pelo AMP.

Tabela III. Propriedades do serviço de difusão atômica consideradas nos testes.

P1.	**Unanimidade**: toda mensagem* entregue a um participante correto** é entregue a todos os participantes corretos.
P2.	**Não trivialidade**: toda mensagem entregue a um participante correto foi emitida por um emissor*** correto.
P3.	**Ordem**: se duas mensagens são entregues a um par de participantes corretos, estas mensagens são entregues na mesma ordem.
P4.	**Visão consistente do grupo**: se um ou mais participantes são extraídos do grupo devido à existência de falhas, as modificações no grupo são indicadas aos participantes corretos, numa ordem consistente. Além disso, nenhum participante correto recebe mensagens provenientes dos participantes defeituosos**** depois que as indicações de modificação no grupo tiverem sido entregues.

* A mensagem é uma unidade de dados de serviço do AMP.

** Um participante é um membro de um grupo de comunicação pré-estabelecido e um participante correto é aquele residente em uma estação isenta de falhas.

*** Um emissor é um participante que faz um pedido de difusão de mensagem aos outros membros do grupo. Um emissor também recebe suas próprias mensagens.

**** Um participante defeituoso é aquele residente em uma estação onde existem falhas.

IV.2. A arquitetura de testes

A configuração do banco de testes distribuído utilizado na validação da arquitetura Delta-4 é mostrado na Figura 3. Para as experiências realizadas, o *sistema alvo* foi composto de 4 **estações**, cada qual constituída por um **hospedeiro** e por um **NAC** através do qual a estação estava conectada à **rede do sistema alvo**. As falhas foram injetadas em um único NAC; deste modo, as estações foram divididas em duas categorias: uma contém a estação injetada (S1) e a outra contém as 3 estações "corretas" **(não injetadas)**.

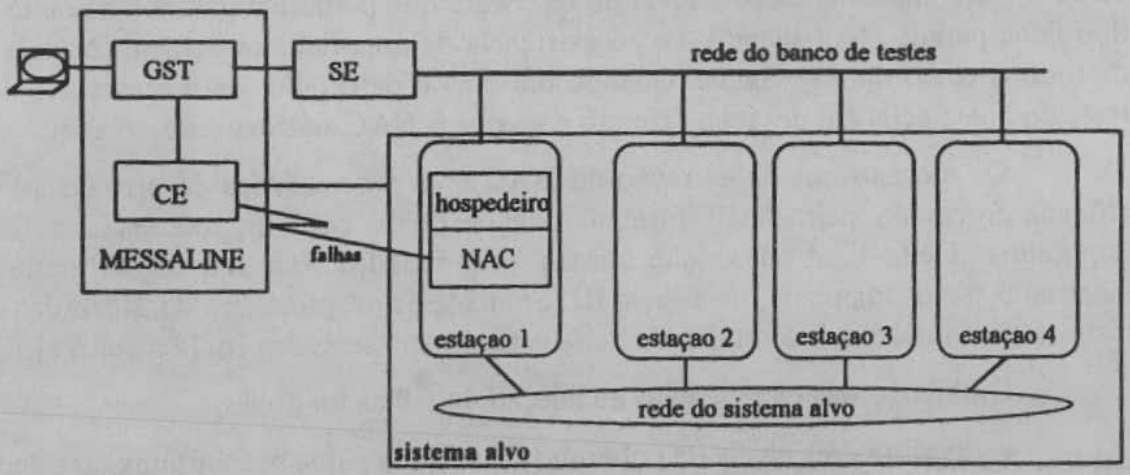

Figura 3. Configuração do banco de testes.

Cada NAC contém uma implementação do AMP. Nos hospedeiros reside uma parte do sistema de teste, responsável pelo estímulo e observação do AMP através da interface de serviço local.

As estações são controladas à distância pelo **supervisor de experiências (SE)**, através da **rede do banco de testes**. Além de comandar as estações, o SE é também responsável pela análise do traço global, obtido pela concatenação dos traços locais coletados pelos diversos hospedeiros. O SE é portanto o componente do sistema de teste

encarregado da verificação. O analisador do traço global é implementado na forma de algoritmo representando as propriedades do serviço de difusão atômica, como foi mostrado na seção III.3.

O suporte para a injeção de falhas é fornecido por MESSALINE, uma ferramenta geral para injeção de falhas a nível dos pinos de circuitos integrados, que foi desenvolvida no LAAS-CNRS. Uma descrição detalhada da ferramenta é apresentada em [Arlat 89]. Serão mencionados aqui unicamente seus dois principais módulos, mostrados na Figura 3.

O módulo de **coordenação das experiências (CE)** tem como função controlar e observar o sistema alvo a nível do hardware. A injeção de falhas no NAC da estação injetada é uma destas funções. Uma outra característica desse módulo é permitir que a ocorrência de um erro (i.e., a ativação da falha injetada) seja identificada no ponto de injeção. Esta facilidade permite determinar se as falhas foram ou não ativadas, e possibilita medir a dormência da falha (intervalo entre a injeção da falha e sua ativação). O CE é também responsável pela re-inicialização física automática do sistema alvo: os NACs são re-inicializados após cada experiência. Como o tempo necessário para re-inicializar um hospedeiro é relativamente importante (de 2 a 15 min), este só é re-inicializado em caso de "crash" da estação. Deste modo é possível ao banco de teste tolerar os defeitos das estações que venham a ocorrer em consequência das falhas injetadas.

O módulo de **gestão da sequencia de teste (GST)** oferece a interface com o operador que permite definir as experiências e iniciar uma campanha de testes. Ele é responsável pelo controle da realização das experiências, e também pelo armazenamento dos dados coletados pelo **SE** (observação do AMP) e pelo **CE** (observação do hardware). Esses dados serão tratados ulteriormente para a obtenção das medidas requeridas (coberturas local e global). Este é portanto o componente do sistema de teste que trata do aspecto avaliação.

Para os testes efetuados, o sistema alvo era preliminarmente homogêneo, constituído por quatro hospedeiros BULL SPS7. O NAC destas estações continha um auto-teste limitado aos mecanismos previstos na norma ISO 8802.5, sendo por isto designado de *LSC-NAC* (de "Limited Self-Checking NAC"). Foi também posteriormente submetida aos testes uma configuração heterogênea, constituída de 3 estações BULL SPS7, munidas de LSC-NACs, e de uma estação Ferranti Argus 2000 contendo um NAC com um auto-teste ampliado, designado por *ESC-NAC* (de "Enhanced Self Checking NAC"). O **SE** foi realizado por um BULL DPX 2000, a ligação entre este componente e os hospedeiros do sistema alvo sendo feita através de uma rede do tipo Ethernet. O módulo GST foi implementado por um Macintosh II, conectado ao SE e ao CE por meio de linhas seriais.

IV.3. Descrição das experiências

Nesta seção são apresentados os atributos da injeção de falhas -os conjuntos **F, A, R e M** (seção II)- utilizados na validação da arquitetura Delta-4.

1. **O conjunto F**: uma falha **f** foi caracterizada por uma série de atributos, e a seleção de uma falha consistiu em escolher aleatoriamente os seus diferentes atributos. Esses atributos, bem como a ordem e a forma de seleção dos mesmos são apresentados a seguir:

1° *Multiplicidade* (*mx*): uma falha podia afetar simultaneamente 1, 2 ou 3 pinos de um circuito, com uma frequencia de 50%, 30% e 20% respectivamente.

2° *Localização*: seleção de *mx* pinos dentre uma lista de pinos injetáveis, segundo uma distribuição uniforme.

3° *Modelo*: seleção dos modelos {bit preso em 0, bit preso em a 1} de maneira equiprovável.

4° *Características temporais*: como as falhas injetadas eram essencialmente intermitentes, seus parâmetros temporais eram constituídos dos seguintes valores:

- *período* (intervalo entre duas aplicações sucessivas), com distribuição logarítmica no intervalo [10μs, 30ms];
- *duração* da aplicação da falha, com distribuição uniforme no intervalo [2μs, 1ms], com uma razão cíclica (duração/período) < 50%.

5° *Retardo* entre o início da ativação do sistema e a injeção da falha, segundo uma distribuição uniforme no intervalo [1s, 40s].

A distribuição uniforme foi adotada todas as vezes em que não se dispunha de informações ou dados mais precisos. No tocante à multiplicidade, a limitação a no máximo 3 pinos foi baseada em parte no estudo apresentado em [Gunneflo 89], no qual um microprocessador é submetido à radiação de íons pesados: os resultados mostraram que mais de 80% das injeções se traduziu pela aparição de uma primeira combinação de erros afetando no máximo 3 pinos. Quanto à escolha da distribuição logarítmica para o *período* teve por objetivo favorecer a geração de um número significativo de falhas com período de recorrência curto, mantendo porém um vasto domínio de seleção.

2. **O conjunto A**: a ativação dos serviços de difusão atômica é baseada no perfil operacional. Assim sendo, foram gerados dois fluxos de mensagem, denominados de *tráfego observado* (*TO*), utilizado para o teste das propriedades do serviço do AMP, e de *tráfego de fundo* (*TF*), visando a manter um nível de carga representativo no sistema alvo.

3. **O conjunto R**: os dados coletados para cada experiência são de tres tipos:
   - valores binários: ativação das falhas injetadas (**E**, verdadeiro se a falha injetada foi ativada); status dos relés de cada NAC ($\mathbf{R_i}$, verdadeiro se a estação $\mathbf{S_i}$ permanece conectada à rede ao final da experiência); resultado da análise das propriedades do AMP ($\mathbf{P_a}$, verdadeiro se a propriedade P_a - da Tabela IV- for satisfeita);
   - parâmetros temporais: ativação das falhas, extração das estações;
   - contabilidade das mensagens: número de mensagens emitidas e recebidas, etc;
   - informações de ajuda ao diagnóstico: códigos de erros, "dumps" de memória, entre outras.

4. **O conjunto M**: as medidas consideradas para a análise apresentada aqui são de dois tipos: *predicados* e *parâmetros temporais*. Como predicados temos:
   - **D**, caracterizando a detecção de um erro por parte dos mecanismos de auto-teste do NAC:

$$D = E \bullet \overline{R_1}$$

A cobertura local caracterizando a eficiência dos mecanismos de auto-teste do NAC foi obtida como:

$$C_D = \frac{|D|}{|E|}$$

A notação A•B designa a conjunção dos predicados A e B; $\overline{A}$ designa a negação do predicado A e |A| indica o número de observações do predicado A.

- C, caracterizando o confinamento do erro (i.e., as estações corretas permanencem conectadas ao anel ao término da experiência):

$$C = R_2 \bullet R_3 \bullet R_4$$

- P, associado à conjunção de todas as propriedades do AMP:

$$P = P_1 \bullet P_2 \bullet P_3 \bullet P_4$$

- T, definindo a tolerância a falhas do protocolo de difusão atômica:

$$T = E \bullet C \bullet P$$

A cobertura global, observada a nível do serviço fornecido pelo AMP pode ser expressa como:

$$C_T = \frac{|T|}{|E|}$$

Como parâmetros temporais temos:

- a dormência de falhas: se T_F designa o momento da injeção e T_E o momento da ativação da falha injetada, então a dormência é dada por:

$$T_d = T_E - T_F$$

- a latência de extração: se T_X designa o momento da extração da estação injetada, a latência é dada por:

$$T_l = T_X - T_E$$

IV.4. Experiências realizadas e alguns resultados obtidos

Como é mostrado na Tabela IV, foram realizados testes sobre 61 circuitos, dos quais 40 eram do LSC-NAC e 21 do ESC-NAC, perfazendo um total de 12099 experiências.

Foram realizadas 150 experiências por circuito, os testes de um circuito sendo realizados de forma inteiramente automatizada; a intervenção do operador era necessária somente para posicionar a sonda de injeção sobre o circuito escolhido para os testes.

Tabela IV. Sumário das experiências realizadas.

Tipo de NAC	Versão do AMP	Falhas injetadas	# de circuitos testados	# de experiências
LSC-NAC	V1	intermitentes	40	4799
		transientes	3	600
		permanentes	3	750
	V2	intermitentes	8	1200
	V2.3	intermitentes	8	1200
	total		*40*	*8499*
ESC-NAC	V2.5	intermitentes	*24*	*3600*

No total, foram realizadas 8499 experiências sobre o LSC-NAC. Deste total, 3750 experiências (correspondendo ao teste de 22 circuitos) tiveram fins específicos: i) analisar o efeito da persistência temporal da falha (1350 experiências, correspondendo a 9 sequencias de teste aplicadas sobre 3 circuitos); e ii) teste de duas versões sucessivas do AMP (2400 experiências realizadas sobre 8 circuitos). Cumpre notar que nos primeiros testes realizados, consistindo na injeção de falhas intermitentes sobre 40 circuitos, nem todos os resultados foram utilizáveis (dos 6000 esperados, somente 4799 foram observados). Esta falta de observações é devida principalmente a problemas de mutações parasitas no sistema alvo em consequência da interferência causada pelo posicionamento da sonda de injeção sobre o circuito, fazendo com que o sistema alvo não funcione corretamente, mesmo quando não há falhas injetadas.

As falhas de concepção/implementação reveladas pelos testes permitiram à equipe que desenvolveu o protocolo de efetuar correções, em muito ajudadas pelas informações de diagnóstico coletadas ao fim das experiências. Deste modo, foram elaboradas várias versões da implementação do AMP, as quais foram submetidas a novos testes.

Neste artigo será focalizado apenas os resultados referentes aos testes das diferentes versões do AMP e uma comparação entre os dois tipos de NAC. Outros resultados são apresentados em [Martins 89, Arlat 90b, Kanoun 91].

A Figura 4 mostra a distribuição dos erros encontrados quando um comportamento errôneo do AMP é observado. Este comportamento errôneo é caracterizado pela combinação $D \bullet \overline{T}$. A cada experiência para a qual foi observada esta combinação, foram coletadas, entre outras informações: "dumps" de memória e os códigos de erros gerados pelas implementações residentes nas estações corretas. Uma análise destas informações permitiu obter a distribuição dos erros encontrados entre os 4 módulos principais do AMP: *Monitor* (garante a coerência interna do grupo), *Emitter* e *Receiver* (executam a transferência de mensagens por difusão atômica) e *Driver* (faz a interface entre o AMP e a camada MAC).

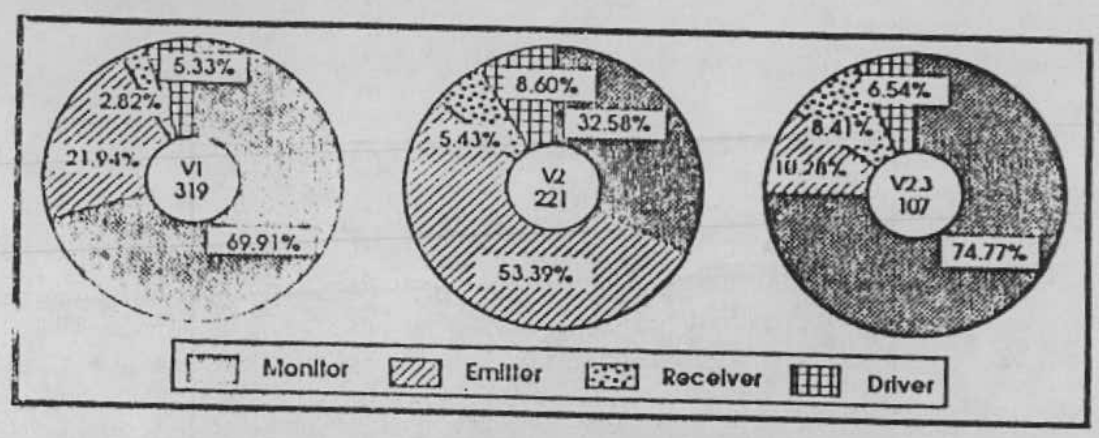

Figura 4. Síntese dos códigos de erros por módulo do AMP.

No centro de cada anel está indicado o total de códigos de erros encontrados; as percentagens indicam a distribuição relativa desses erros entre os quatro módulos do AMP. De princípio pode-se constatar que o número de erros encontrados diminuiu continuamente de uma versão à outra (houve praticamente uma redução de 1/3 entre as versões V1 e V2.3). Pode-se notar também que entre as versões V1 e V2, as correções visaram principalmente o Monitor. Como consequência, houve um aumento no número de erros observados no módulo Emitter; o mais provável é que a observação desses erros tenha sido mascarada devido à grande proporção de erros existentes no Monitor na versão V1. Note-se que a proporção de erros do Emitter diminuiu entre as versões V2 e V2.3, voltando o Monitor a ser o módulo com o maior número de erros. A maioria dos erros do Monitor foram devidos a estouro de temporização durante a reconfiguração do sistema. Como será mostrado mais adiante, uma parte destes erros não foi devida unicamente ao comportamento errôneo do AMP.

A ocorrência de casos $D \bullet \overline{T}$ pode resultar de:

- deficiência na concepção/implementação do protocolo em tratar a extração de uma estação;
- latência excessiva dos mecanismos de auto-teste do NAC, favorecendo deste modo a propagação dos erros.

Afim de determinar de forma mais precisa a causa desses erros, forma realizadas experiências específicas de extração (cada experiência consistiu simplesmente em forçar a extração da estação) afim de observar o comportamento do AMP. Estas experiências permitiram constatar que efetivamente nem todas as ocorrências de $D \bullet \overline{T}$ eram devidas unicamente a deficiências na implementação do protocolo. A Figura 5 mostra as proporções relativas dos casos $D \bullet T$ (caracterizando o comportamento esperado do sistema) e $D \bullet \overline{T}$ em função da latência de extração da estação injetada.

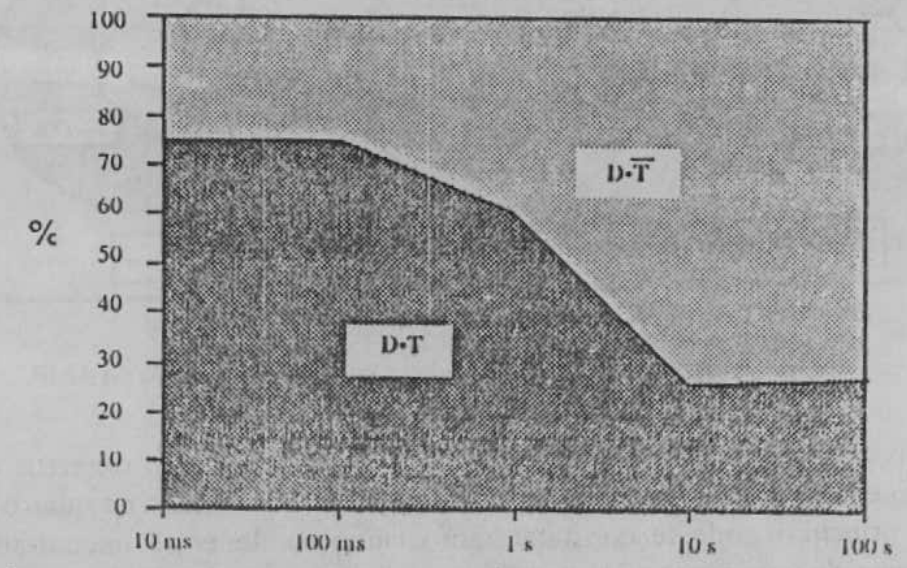

Figura 5. Proporção relativa de casos $D \bullet T$ e $D \bullet \overline{T}$ em função da latência de extração.

Estes resultados mostram que a proporção de casos $D \bullet T$ é quase o triplo dos casos $D \bullet \overline{T}$, para latências de extração entre 10-100ms, e que essa proporção é praticamente revertida para o intervalo 10-100s (a duração de uma experiência era de 110s). Dessa forma foi mostrada a influência da latência na ocorrência de casos $D \bullet \overline{T}$. Como os mecanismos de auto-teste não são controlados pelo AMP, a redução da proporção de erros devido à latência excessiva requeria modificações a nível do hardware.

A Figura 6 mostra a distribuição acumulada da latência para os dois tipos de NAC. As curvas mostram que, enquanto para o LSC-NAC somente 30% dos erros são detectados em menos de 1s, esta proporção excede 80% para o ESC-NAC. Além disso os valores obtidos em 100s representando a cobertura de detecção (C_D), vemos que houve um aumento desta última para o ESC-NAC.

Os resultados mostrados correspondem a testes em diferentes versões do protocolo (V2.3 para o LSC-NAC e V2.5 para o ESC-NAC). No entanto, além de não ter sido notada uma diferença significativa nos resultados dos testes destas duas versões realizados sobre um LSC-NAC, as análises estatísticas realizadas com os resultados da cobertura local para as versões V1, V2 e V2.3 serviram para mostrar que as correções realizadas no AMP não tiveram nenhum efeito significativo sobre C_D [Arlat 91]. Portanto a melhoria observada na eficiência dos mecanismos de auto-teste do ESC-NAC pode ser atribuída à melhoria na arquitetura fisica desse NAC.

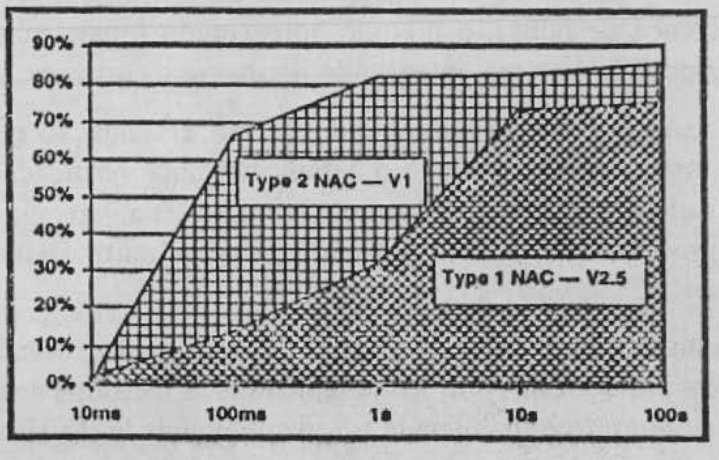

Figura 6. Comparação das distribuições acumulada da latência para o LSC-NAC e o ESC-NAC.

V. Conclusão

Neste artigo foi proposta uma metodologia para a validação de sistemas distribuídos tolerantes à falhas, mais precisamente dos protocolos tolerantes à falhas, utilizando injeção de falhas.

Sendo aplicável quando se dispõe de um protótipo do hardware e do software do sistema, o método de injeção física de falhas pode ser considerado como um método de teste de uma implementação. Os problemas comuns à validação da implementação de protocolos de comunicação (definição de uma arquitetura de testes, geração das sequências de teste, análise dos resultados) foram abordados neste artigo.

O método de injeção física de falhas apresenta algumas dificuldades para a sua aplicação: domínio de entrada duplo (falhas e ativações do sistema), esforço necessário para a realização de um banco de testes confiável, sincronização entre injeção de falhas e a ativação do sistema, necessidade de realizar um número elevado de testes. Além destes, tem-se também os problemas físicos, como mutações parasitas, inerentes à técnica de injeção utilizada.

Apesar destas dificuldades, o método de injeção física de falhas apresenta vantagens que tornam sua aplicação fortemente atrativa na validação de sistemas tolerantes à falhas em geral. Uma de suas principais vantagens é o fato de permitir que os dois aspectos da validação (verificação e avaliação) sejam tratados:

- na verificação, as falhas injetadas têm por objetivo colocar em evidência as falhas de concepção e /ou implementação do sistema,

- na avaliação, o tratamento estatístico dos dados coletados durante as experiências permite obter medidas da eficiência dos mecanismos de tolerância a falhas (fator de cobertura, latência, etc).

Mais especificamente no que diz respeito a protocolos tolerantes a falhas, onde as propriedades do serviço fornecido dependem não somente da cooperação entre as diversas entidades, mas também do comportamento de processadores e dos meios de comunicação em presença de falhas, o método apresentado é particularmente útil, pois permite a validação global do sistema, integrando hardware e software.

Uma outra vantagem do método é que, devido à validação global do sistema, ele permite detectar erros que dificilmente o seriam em uma verificação de um modelo ou em um teste isolado da implementação. A influência da latência de extração no comportamento do protocolo de difusão atômica da arquitetura Delta-4, mostrado na seção IV, é um exemplo desse tipo de erro.

Além de complementar outras técnicas de verificação, o método de injeção física de falhas também pode servir como complemento aos métodos analíticos usados na avaliação de medidas como a confiabilidade e a disponibilidade do sistema, fornecendo estimativas de parâmetros importantes (como o fator de cobertura) para estes modelos. Um exemplo desta complementaridade pode ser visto em [Kanoun 91], onde os resultados referentes à eficiência dos mecanismos de auto-teste do NAC foram utilizados na avaliação da confiabilidade da arquitetura Delta-4.

REFERÊNCIAS

[Arlat 89] J.Arlat, Y.Crouzet, J.-C.Laprie. Fault injection for dependability validationof fault-tolerant computing systems. *Proc. FTCS-19*, Chicago, IL, USA, jun. 1989.

[Arlat 90a] J.Arlat. Validation de la sûreté de fonctionnement par injection de fautes: méthode-mise en œuvre-application. Tese de doutorado de estado, INPT, 1990.

[Arlat 90b] J.Arlat, M.Aguera, L.Amat, Y.Crouzet, J.-C.Fabre, J.-C.Laprie, E.Martins, D.Powell. Fault injection for dependability validation - a methodology and some applications. *IEEE Transactions on Software Engineering*, vol. 16, fev. 1990.

[Arlat 91] J.Arlat, Y.Crouzet, E.Martins, D.Powell. Dependability testing report LA3 - fault injection on the Extended Self-Checking NAC. *Relatório internodo projeto Delta-4*, nº 91.396, dez.. 1991.

[Ayache 79] J.M.Ayache, P.Azema, M.Diaz. Observer: a concept for on-line detection of control errors in concurrent systems. Proc. FTCS-9, Wisconsin, Madison, USA, 1979.

[Baumgarten 89] B.Baumgarten, A.Giessler, R.Platten. Test derivation from net models. *Proc. IFIP TC6 2nd. Workshop on Protocol Test Systems*, Berlim, Alemanha, 1989.

[Berg 82] H.K.Berg. Distributed system testbeds. *Computer*, 15(10), out. 1982.

[Birman 87] K.Birman, T.Joseph. Reliable communication in the presence of failures. *ACM Transactions on Computer Systems*, 5(1), fev. 1987.

[Bochmann 80] G.v.Bochmann. A general transition model for protocols and communication services.. *IEEE Transactions on Communications*, COM-28(4), 1980.

[Bochmann 88] G.v.Bochmann, R.Dssouli, B.Sarikaya. Méthodes de test de protocoles: architecture et sélection de tests. *CFIP'88- Colloque Francophone sur l'Ingénierie des Protocoles*, Bordeaux, 1988.

[Bochmann 89] G.v.Bochmann, R.Dssouli, J.R.Zhao. Trace analysis for conformance and arbitration testing. *IEEE Transactions on Software Engineering*, 15(11), 1989.

[Bouricius 69] W.G.Bouricius, W.C.Carter, P.R.Schneider. Reliability modeling techniques for self-repairing computer systems. *Proc. 24th. National Conference of ACM*, 1969.

[Chang 84] J.Chang, N.Maxemchuck. Reliable broadcast protocols. *ACM Transactions on Computer Systems*, 2(3), ago. 1984.

[Cooper 85] E.C.Cooper. Replicated distributed programs. *ACM Operating systems Review*, 19(5), 1985.

[Cristian 91] F.Cristian. Understanding fault tolerant distributed systems. Communications of the ACM, 34(2), 1991.

[Crouzet 82] Y.Crouzet, B.Decouty. Measurements of fault detection mechanisms efficiency: results. *Proc. FTCS-12*, Santa Monica, CA, USA, jun 1982.

[Damm 88] A.Damm. Experimental evaluation of error-detection and self-checking coverage of components of a distributed real-time system. Tese de doutorado, Tech. Univ. Viena, 1988.

[Dssouli 86] R.Dssouli, G.v.Bochmann. Error detection with multiple observers. *Protocol Specification, Testing and Verification V*, 1986.

[Groz 89] R.Groz. Vérification de propriétés logiques des protocoles et systèmes répartis par observation de simulations. Tese de doutorado, Univ. de Rennes, 1989.

[Gunneflo 89] U.Gunneflo, J.Karlsson, J.Torin. Evaluation of error detection schemes using fault injection by heavy-ion radiation. *Proc. FTCS-19*, Chicago, IL, USA, 1989.

[Hummel 88] R.A.Hummel. Automated fault injection for digital systems. *Proc.Reliability and Maintenability Symposium*, Los Angeles, California, USA, 1988.

[ISO9646-1] ISO9646-1, ISO OSI Conformance testing methodology and framework, part 1: general concepts. n° ISO/IEC JCT 1/SC 21 DIS 9646, jul 1988.

[ISO9646-2] ISO9646-2, ISO OSI Conformance testing methodology and framework, part 2: abstract test suite specification. n° ISO/IEC JCT 1/SC 21 DIS 9646, jul 1988.

[ISO9646-3] ISO9646-2, ISO OSI Conformance testing methodology and framework, part 3: Tree and Tabular Combined Notation (TTCN). n° ISO/IEC JCT 1/SC 21 DIS 9646, jul 1988.

[ISO9646-4] ISO9646-2, ISO OSI Conformance testing methodology and framework, part 4: test realization. n° ISO/IEC JCT 1/SC 21 DIS 9646, jul 1988.

[Kanoun 91] K.Kanoun, J.Arlat, L.Burrill, Y.Crouzet, S.Graf, E.Martins, A.MacInnes, D.Powell, J.L.Richier, J.Voiron. Delta-4 architecture validation. *ESPRIT Conference Week*, Bruxelas, Bélgica, 1991.

[Kurlak 81] R.P.Kurlak, J.Chobot. CPU coverage evaluation using automatic fault injection. *American Institute of Aeronautics and Astronautics*, 1981.

[Lala 83] J.H.Lala. Fault detection, isolation and reconfiguration in FTMP: methods and experimental results. *5th. IEEE/AIAA Digital Avionics System Conference*, 1983.

[Laprie 92] J.-C.Laprie. Sûreté de fonctionnement: concepts de base et terminologie. *Dependable Computing and Fault Tolerance*. Springer Verlag, 1992.

[Martins 89] E.Martins, J.Arlat, Y.Crouzet, J.C.Fabre, D.Powell. Testing multipeer protocols in the presence of faults. *Proc. IFIP TC6 2nd. Workshop on Protocol Test Systems*, Berlim, Alemanha, 1989.

[Martins 92] E.Martins. Validation de systèmes répartis par injection de fautes. Tese de doutorado, ENSAE, 1992.

[Molva 86] R.Molva, M.Diaz, J.M.Ayache. A run-time checking tool for local area networks. *Protocol Specification, Testing and Verification V*, 1986.

[Powell 91] D.Powell. Delta-4: a generic architecture for dependable distributed computing. Research Reports ESPRIT, Springer-Verlag, 1991.

[Sidhu 89] D.P.Sidhu, C.S.Chang. Probabilistic testing of protocols. *ACM SIGCOMM'89*, Austin, Texas, USA, set. 1989.

[Thévenod 91] P.Thévenod. From random testing of hardware to statistical testing of software. *IEEE Comp'Euro91*, Bolonha, Itália, 1991.

[Verissimo 89] P.Veríssimo, L.Rodrigues, M.Baptista. AMp: a highly parallel atomic multicast protocol. *ACM SIGCOMM'89*, Austin, Texas, USA, set. 1989.

[Wang 1987] B.Wang, D.Hutchison. Protocol testing techniques. *Computer Communications*, 10(2), 1987.