Detecção de Falha para Redes de Larga Escala no Fault-Tolerant CORBA

Lau Cheuk Lung, Joni da Silva Fraga
Laboratório de Controle e Microinformática (LCMI) – DAS – CTC – UFSC
Campus Universitário – Caixa Postal 476 – Trindade - CEP 88040-900 – Florianópolis – SC – Brasil
E-mail: {lau, fraga}@lcmi.ufsc.br

Resumo

Este trabalho apresenta uma proposta de extensão da especificação do serviço de detecção de falha do padrão FT CORBA [OMG99]. A motivação disso, é que muitos dos requisitos de tolerância falha para sistemas de larga-escala, tal como a internet, ainda não foram discutidos, de forma precisa, nessa especificação. A solução proposta visa atender aos requisitos de sistemas de larga escala (inerentemente assíncronos). Um protocolo de acordo baseado em voto majoritário é adaptado de forma a melhor se adequar à especificação FT CORBA.

***Palavras chave**: detecção de falha, sistemas de larga escala, tolerância a falhas, CORBA e sistemas abertos.*

1. Introdução

Tolerância a falhas em sistemas distribuídos tem um importante papel no desenvolvimento de aplicações confiáveis. O objetivo primário desta área de pesquisa é a proposição de soluções que permitam às aplicações de terem seus serviços oferecidos continuamente, mesmo na presença de falhas parciais no sistema computacional. Por outro lado, nos últimos anos, a área de sistemas distribuídos tem avançado na direção da padronização aberta. Esta tendência vem no sentido de combater soluções proprietárias devido aos altos custos de desenvolvimento de sistema provenientes, por exemplo, da dificuldade de manutenção de software, interoperabilidade e portabilidade.

Atualmente, muitas aplicações tolerantes a falhas estão seguindo o paradigma de orientação a objetos e considerando o CORBA (*Common Object Request Broker Architecture*) [OMG 96] como uma alternativa de se adequar a sistemas abertos. Como resultado disso, a OMG (*Object Management Group*) apresentou, após alguns anos de trabalho, uma proposta de especificação revisada do Fault Tolerant CORBA (FT CORBA) [OMG 99]. As especificações FT CORBA, que ainda devem passar por varias revisões (e extensões), definem um conjunto de interfaces de serviços e facilidades úteis para a implementação de técnicas de replicação em ambientes distribuídos heterogêneos. As especificações atendem apenas aos requisitos básicos para tolerância a falhas, definindo algumas interfaces bem genéricas, de fácil entendimento, e aplicáveis em todas aplicações que requerem tolerância a falhas.

O comitê da OMG responsável pela padronização do FT CORBA optou, para esse primeiro momento, por assumir apenas alguns compromissos. Em particular, tratar o aspecto da interoperabilidade entre implementações de ORB, com tolerância a falhas, de diferentes fabricantes de software. Todavia, é previsto pelo grupo responsável pelo FT CORBA que, ao longo de um processo de amadurecimento, novas extensões para tolerância a falhas, como interfaces e protocolos, e melhor interoperabilidade sejam agregadas à esta especificação presente [OMG99]. A OMG aconselha também que algumas necessidades, além daquelas definidas no FT CORBA, sejam atendidas, no momento, por soluções proprietárias. Além dos esforços da OMG, são encontrados, na literatura, diversos trabalhos com propostas de extensão do CORBA para tolerância a falhas [Felber98, Shung98, Moser98, Lau99].

Em uma análise mais detalhada dessas especificações, e mesmo destas experiências da literatura, podemos verificar que muitos dos requisitos de tolerância a falhas para sistemas de larga-escala, tal como a internet, ainda não foram discutidos, de forma precisa, na extensão destas especificações CORBA. Os sistemas de larga-escala, inerentemente assíncronos, são caracterizados pela grande distribuição espacial, com um caráter aberto, integrando quantidades significativas de recursos computacionais assinalados pela sua heterogeneidade. Esses sistemas são não deterministas no sentido em que os parâmetros temporais, tais como padrões de tráfego, taxas de transferência, atrasos de comunicação e diferentes velocidades de processamento do processador, não são conhecidos *a priori*. Neste artigo, apresentamos um estudo sobre as especificações FT CORBA e discutimos um algoritmo de detecção de falhas, descrito na forma de abstrações FT CORBA, que tem como finalidade o uso em sistemas de larga escala.

Este artigo está dividido da seguinte maneira. Na seção 2 é apresentada uma descrição resumida das especificações de tolerância a falhas no CORBA. Na seção 3 é feita uma critica em relação a especificação de detecção de falha do FT CORBA. Na seção 4 são apresentados o *GroupPac* e a sua

proposta para detecção de falhas em sistemas de larga escala. Finalmente, na seção 5 são levantadas as conclusões deste trabalho.

2. A especificação Fault Tolerant CORBA

A OMG definiu, recentemente, uma especificação para introduzir tolerância a falhas nos padrões CORBA (FT CORBA) [OMG99]. Esta especificação define um conjunto de serviços essenciais para o desenvolvimento de aplicações confiáveis em sistemas abertos. Nessa primeira especificação foi apresentado um conjunto de interfaces e protocolos para gerenciamento de replicação, gerenciamento de falhas, gerenciamento de recuperação e *logging* e interoperabilidade. O *gerenciamento de replicação* é constituído pelos serviços de gerenciamento de propriedades, gerenciamento de grupo de objetos e de fábricas genéricas. O *gerenciamento de falhas* tem definidas as interfaces de detecção de falhas, notificação de falhas e análise de falhas. Finalmente, no *gerenciamento de recuperação e logging* são definidas os mecanismos para transferência de estado de objeto e recuperação de réplicas faltosas.

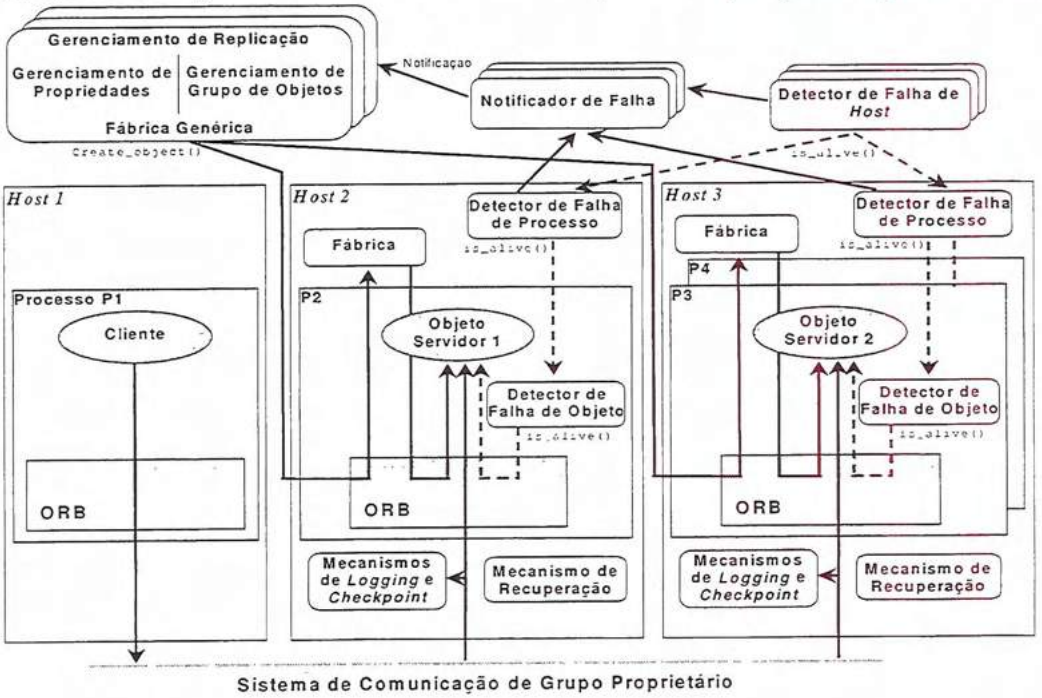

Figura 1. Arquitetura de tolerância a falha CORBA.

Na figura 1, é apresentada a arquitetura de tolerância a falhas do CORBA. Nela são apresentados os objetos de serviço (*Common Object Services Specification* [OMG96]), que fornecem no *middleware*, funcionalidades básicas para a construção de servidores tolerantes a falhas. Segundo as especificações, o gerente de replicação é responsável pelo gerenciamento de grupo, exercendo um controle dinâmico nas entradas e saídas (normal ou por falha) de objetos de um grupo pela manutenção de listas atualizadas de seus membros (*membership*). Na criação ou remoção de réplicas, o objeto de serviço gerente de replicação usa um objeto fábrica genérica para criar/remover membros do grupo de objetos da aplicação. O objeto fábrica genérica negocia com objetos fábrica locais para a criação/remoção de réplicas nas diferentes estações de um sistema distribuído. No processo de criação de réplicas é utilizado o serviço de *logging* e *checkpoint*, que é um mecanismo para registro e atualização de estados que, a partir de uma réplica identificada como primária (objeto servidor 1, na figura 1), faz transferências de estado para novas réplicas que se juntam ao grupo. No serviço de gerenciamento de propriedades são definidas e manipuladas as propriedades de tolerância a falhas dos grupos de objetos mantidos pelo serviço de gerenciamento de replicação.

O suporte fornecido pelo serviço de *gerenciamento de falhas* é constituído por objetos de serviço divididos em detectores de falhas de objeto, de processo e de *host* (figura 1). Todos os objetos detectores de falhas são baseados em mecanismos de *timeout*. O serviço notificador de falhas é replicado e tem a função de enviar mensagens de notificação ao gerente de replicação, a partir dos registros de falhas

recebidos dos detectores de falhas dos três níveis citados. Esses registros são usados no gerente de replicação na atualização das listas de membros. As especificações definem, também, as técnicas de replicação que podem ser suportadas pela arquitetura FT CORBA: Replicação Ativa, Replicação Passiva Fria e Replicação Passiva Quente [OMG99].

A OMG, como é de sua característica, se limita em definir interfaces de serviço genéricas, tentando atender diferentes abordagens de tolerância a falhas. Por exemplo, protocolos de comunicação de grupo confiável, base fundamental para a implementação de técnicas de replicação, não são padronizados – como era de se esperar, pela complexidade dos mesmos e pela quantidade de algoritmos envolvidos. A OMG enfatiza, neste caso, o uso de soluções proprietárias (figura 1). Todavia, para garantir um mínimo grau de interoperabilidade, os sistemas de comunicação de grupo devem adotar a IOGR (*Interoperable Object Group Reference*), a referência de grupo de objetos definida em [OMG99]. A IOGR é uma extensão da IOR (*Interoperable Object Reference*), de um simples objeto, para uma referência interoperável de grupo de objetos. Uma IOGR permite a um cliente referenciar um grupo de objetos como uma entidade única. De forma genérica, a IOGR contém o conjunto de IORs de cada membro de um grupo de objeto.

3. Considerações sobre as especificações FT CORBA

As especificações FT CORBA definem um gerente de replicação para cada *domínio de tolerância a falhas* (TF). Um domínio de TF pode conter diversos grupos de objetos, onde cada grupo possui suas próprias propriedades de tolerância a falhas. O gerente de propriedades, que está associado com o gerente de replicação, se responsabiliza pela gestão dessas propriedades no domínio. Portanto, um domínio TF é constituído de grupos, sujeitos a mesma gestão de replicação e de propriedades, que são formados por *hosts*, processos e objetos. Os objetos pertencentes a um domínio podem não estar limitados a uma única rede local (LAN – *Local Area Network*) ou ainda a uma rede metropolitana (MAN - *Metropolitan Area Network*), mas distribuídos ao longo de uma rede de longa distância (WAN - *Wide Area Network*).

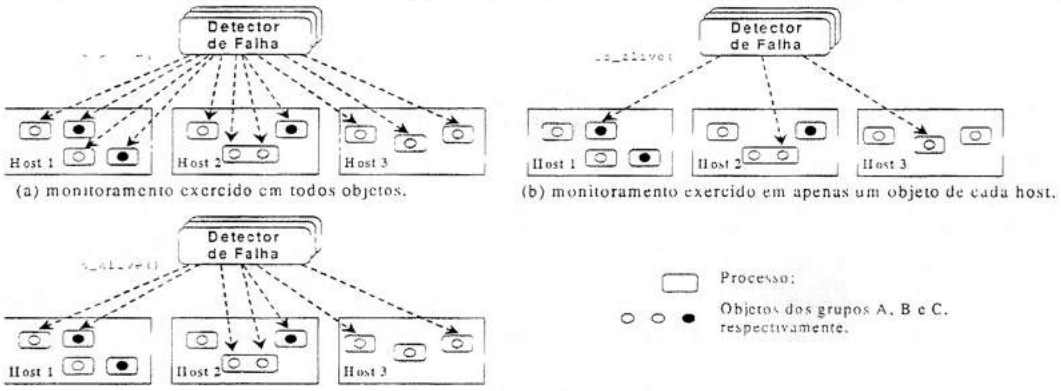

Figura 2. Tipos de monitoração de falha no FT CORBA.

Essas diferentes opções na topologia do domínio podem implicar na definição de diferentes propriedades de tolerância a falhas. O serviço de gerenciamento de grupo de objetos, que faz parte do serviço de gerenciamento de replicação, depende do serviço de gerenciamento de falhas para detectar objetos falhos. Nas especificações dos detectores de falhas são definidos três *granularidades* de monitoramento (detecção) de falhas: sobre cada membro individual de um grupo (figura 2a); sobre um membro representativo em um *host H* contendo mais de um objeto – a falha do membro representativo é considerado como a falha de todos objetos pertencente ao *host H*, independente se os mesmos são membros do mesmo grupo ou não (figura 2b); e finalmente, onde o monitoramento é feito sobre um membro representativo de cada grupo presente através de suas réplicas em um *host H* (figura 2c) – a falha do membro representativo indica a falha de todos os membros do mesmo grupo no *host H*.

As especificações FT CORBA definem que todos os *hosts*, contendo objetos de grupo, devem ser monitorados por pelo menos um detector de falha. Os detectores de falhas de um domínio podem ser replicados atuando sobre os mesmos *hosts* ou os mesmos objetos. Mas situações diversas também são admitidas: detectores podem não monitorar, necessariamente, os mesmos conjuntos de *hosts* ou objetos do domínio. Isto é, dentro do domínio de tolerância a falhas, podemos ter detectores monitorando

diferentes subconjuntos de *hosts* que compõe o domínio. Por exemplo, é possível um *host* sendo monitorado por um único detector de falha em um domínio de TF.

Detecção de falhas em sistemas assíncronos é uma classe de problemas onde só são permitidas soluções probabilistas [Fischer85, Ricciardi91, Chandra96]. As especificações FT CORBA fornecem alguns recursos que nas suas definições não são bem adequados a estes ambientes. É o caso do monitoramento de um *host* a partir de um conjunto de detectores de falhas onde o *host* é assumido como falho quando não responder dentro de *timeouts* estipulados a pelo menos um dos detectores do conjunto. Se considerarmos as características assíncronas de um sistema de larga escala este tipo de procedimento é extremamente penalizante.

Para tratar com as dificuldades destes ambientes estendemos a noção de detectores de falhas do FT CORBA, sem que isto represente na modificação das interfaces da especificação. Assumimos então que um *host* é considerado em *crash* se não responder a um certo número de detectores de falhas dentro de *timeouts* específicos. Um protocolo de acordo [Ricciardi91, Hufin99] entre os detectores de falhas para "determinar" se um processo está falho é necessário. O monitoramento de um *host* a partir de um grupo de detectores de falhas minimiza os erros de detecção. Em sistemas com características assíncronas esta condição não é suficiente para uma precisão determinista na detecção de um *host* ou processo falho. Porém, se associarmos algumas semânticas a estes detectores de falhas, as condições de detecção podem ainda ser melhoradas na precisão. Em [Chandra 96], várias semânticas podem ser assumidas para detectores de falhas não confiáveis a partir de duas propriedades: *completitude* ("*completeness*") e a *exatidão* ("*accuracy*"). A Completitude especifica que um detector de falha terminará por suspeitar de todos processos que realmente estiverem falhos. E a exatidão restringe os enganos (suspeitas errôneas) que um detector possa cometer. Assumimos neste trabalho detectores de falha da classe *Temporizada forte* ("*Eventually Strong*") $\Diamond S$[1].

4. GroupPac

O *GroupPac* [Lau99, Lau00] consiste de um conjunto de serviços específicos para o desenvolvimento de aplicações tolerante a falhas. Adere as recomendações da OMG como objetos de serviços, tais como as especificações COSS [OMG96]. Os serviços de tolerância a falhas do GroupPac têm suas interfaces baseadas nas especificações FT CORBA (serviços de gerenciamento de replicação, gerenciamento de falha, e gerenciamento de recuperação e *logging*). A diferença em relação ao padrão é que apresenta um conjunto de soluções (extensões) no sentido de atender aos requisitos de tolerância a falhas em sistemas larga escala. As extensões, propostas no *GroupPac*, foram incluídas no serviço de gerenciamento de falhas – especificando um protocolo para detecção de falhas de *hosts*. Além disso, é proposto um serviço de comunicação de grupo para sistemas de larga-escala, com propriedades de difusão atômica ("*Atomic Broadcast*"). Esta solução é adaptada ao ORB de forma transparente à aplicação. Num contexto de sistema de larga-escala com vários grupos de objetos em domínios de tolerância a falhas dispersos geograficamente, o GroupPac assume a nível de transporte canais de comunicação ponto-a-ponto confiáveis, servindo de suporte aos objetos se comunicando por troca de mensagem. Como hipótese de falhas é assumido que objetos (ou *hosts*) falham por *crash*, podendo se recuperar através dos serviços do FT CORBA (figura 1).

4.1. Detecção de falhas em sistemas de larga escala

Conforme discutido no item 3, a detecção de falha do FT CORBA não é adequada para sistemas de larga escala. A solução que adotamos neste trabalho foi especificar um protocolo baseado em voto majoritário para alcançar acordo na decisão sobre as suspeitas de falhas de *hosts*. Diferente de [Chandra96], em que um módulo detector é agregado a cada processo[2], a nossa solução, para melhor se

---

[1] A semântica *Temporizada forte* ("*Eventually Strong*") $\Diamond S$ é definida com as propriedades [Chandra96]: (1) *Completitude forte* ("*strong completeness*"): todo processo falho estará permanentemente presente em todas as listas de suspeitos de falhas dos processos corretos do sistema; (2) *Exatidão fraca temporal* ("*eventual weak accuracy*"): a partir de um certo tempo finito um processo correto deixa de pertencer à lista de suspeitos de qualquer processo correto.

[2] O conceito de *detectores de falhas não confiáveis* foi introduzido em [Chandra 96] para serem usados na obtenção de consenso em sistemas com características assíncronas. Neste caso, a cada processo da aplicação no sistema é associado um módulo detector cuja função é notificar ao processo associado a suspeita de falha de outros processos do sistema. Todas as falhas são por *crash* e é assumido um canal de comunicação confiável entre os processos. As suspeitas de falhas são implementadas basicamente através de *prazos estipulados* para tempos (*timeout*) de resposta dos processos.

adequar as especificações FT CORBA, propõe um conjunto de detectores dedicados, baseado na abordagem cliente/servidor orientado a objeto, dispostos na rede, podendo ou não estar no mesmo *host* da aplicação. Tomando como exemplo os detectores de falha de *host* (figura 1 e 3), é estabelecido no nosso esquema que todos os detectores monitoram todos os *hosts* dentro de um domínio de tolerância a falhas. Assumimos que os detectores são sempre completos, após um certo tempo suspeitará de todo processo permanentemente faltoso ou desconectado dentro do domínio de tolerância a falhas.

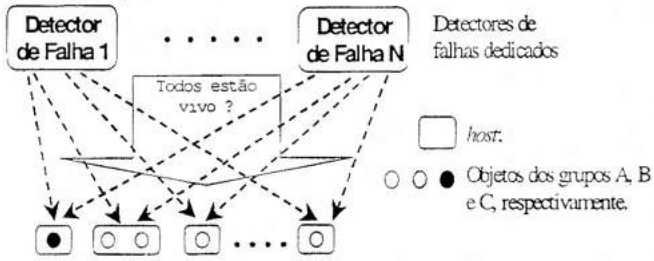

Figura 3. Monitoração de falha no GroupPac.

O serviço de detecção de falha é concretizado em dois níveis de monitoramento: em nível de detectores e em nível de *hosts*. No primeiro, os detectores de falhas de *hosts* formam um grupo autogerenciável. Isto é, controlam as entradas e saídas (normal ou por falha) dos detectores membros do grupo. O monitoramento neste nível se baseia no protocolo apresentado em [Ricciardi91, Lau99].

Os membros do grupo de detectores compõem um anel virtual, e cada membro monitora o parceiro imediatamente anterior na seqüência definida no anel. O serviço de gerenciamento de pertinência (*membership*) no grupo de detectores usa um protocolo de *commit* centralizado (no primário) de duas fases (três fases no pior caso) para alcançar acordo entre os membros em relação a nova composição do grupo de detectores. O monitoramento de falhas de detectores é necessário para indicar o número de membros corretos no grupo de detectores.

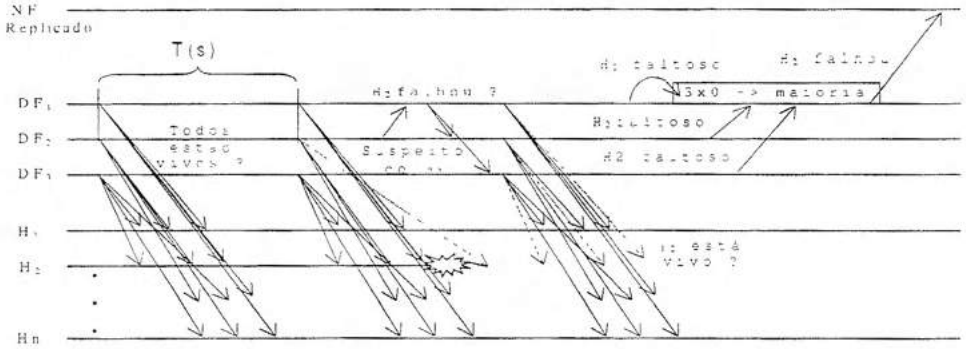

Figura 4. Uma instância do protocolo de detecção de falha.

A detecção de falhas de *hosts* é implementada pelo grupo de detectores de falhas de *hosts* (DF1, DF2 e DF3 da figura 1 e 4). Na falha de um *host* é assumida a falha de todos processos, e conseqüentemente dos objetos, neste *host*. Todos os DFs monitoram periodicamente, dentro de um intervalo T(s), o mesmo conjunto de *hosts* dentro de um domínio de tolerância a falha. São estes detectores que decidem em consenso se um determinado *host* é falho (*crash*) ou não. Caso qualquer um dos detectores suspeite da falha de um dos *hosts* no domínio de TF, o protocolo executa um procedimento de acordo baseado em voto majoritário para alcançar consenso entre os detectores. A coordenação do processo de votação é sempre realizada pelo detector de falha identificado como *primário* na ordenação do anel. Por exemplo, na figura 4, o DF2 suspeita do *host* H2 e avisa ao DF1 (o primário) sobre esta suspeita. A partir daí é iniciado o protocolo, em três passos, para se alcançar o consenso sobre a falha ou não de H2. Na primeira fase, o DF1 solicita aos outros detectores (DF2 e DF3 da figura 4) para que no próximo intervalo de monitoração (T(s)) verifiquem o status (vivo ou falho) do *host* H2. No segundo passo, após a nova monitoração, cada detector envia seus resultados sobre o status de H2 ao DF1. Finalmente, no passo três, o detector de falha primário (DF1) executa a função de consenso sobre os valores enviados de status de H2 (na figura 4). O detector DF1 (primário), através do notificador de falhas (NF – figura 1 e 4), informa ao gerenciador de replicação a falha de H_2 que, por sua vez, deve determinar uma nova IOGR (item 2) removendo H_2 da lista. A nova IOGR é enviada ao grupo de detectores para que estes atualizem suas listas de *hosts* a serem monitorados.

Em um contexto de rede de larga escala, com diversos grupos em diferentes domínios de tolerância a falhas (TF), o serviço de gerenciamento de replicação (SGR) de cada domínio de TF, o qual

utiliza o serviço de detecção de falha para gerenciamento de grupo (figura 1), deve disponibilizar sua lista de membros aos outros domínios de TF do sistema. A interação dos diferentes domínios de TF é concretizada estabelecendo um SGR universal, usualmente replicado para tolerância a falhas, coordenando os SGRs de cada domínio de TF. A solução de adotar uma estrutura hierárquica de SGRs é eficiente e escalável, pois limita o gerenciamento de grupo em cada domínio de TF diminuindo o número de mensagens do tipo atualização de lista (*view*) e de detecção ("todos estão vivos ?") no sistema de larga escala.

5. Conclusão

Em termos de trabalhos relacionados, a solução apresentada em [Shung98] usa um conjunto de monitores, em uma rede local, similares aos detectores identificados nas especificações FT CORBA. Basta que um dos chamados de *WatchDog* sinalize que um dos objetos da aplicação é falho para que o mesmo seja retirado da configuração da aplicação. No OGS [Felber98] o monitoramento de falha é, também, similar ao das especificações. No entanto, o OGS apresenta um objeto de serviço de consenso que poderia ser adaptado em detecções envolvendo consensos. Em relação as interfaces destes serviços, ambos possuem interfaces diferentes das descritas nas especificações FT CORBA.

Este artigo apresentou um serviço de detecção de falha para sistemas de larga escala que se adapta bem às especificações FT CORBA. A solução adotada não modifica as especificações no sentido de que as interfaces continuam a apresentar as mesmas funcionalidades. Os protocolos executados por trás da interface são transparentes na arquitetura como um todo.

O protocolo, proposto neste trabalho, é baseado em servidores dedicados de detecção de falha e de replicação. Isto é, ao invés de todos processos da aplicação se preocuparem com a detecção de falha, cada um gerenciando localmente uma lista dos membros suspeitos e ativos do grupo, tal como [Chandra96]. Nesta proposta, esta função é realizada por um número de detectores dedicados (por exemplo, uma dúzia) dentro do domínio de TF (figura 3), o que reduz, consideravelmente, o tráfego na rede. Outra característica deste protocolo é o estabelecimento de uma estrutura hierárquica de SGRs para uma melhor eficiência em redes de larga escala.

Este trabalho, em andamento, faz parte do projeto GroupPac que visa propor soluções para tolerância a falha às aplicações CORBA em um contexto de sistema de larga escala. Atualmente, está em pesquisa soluções de protocolos de difusão atômica e a sua integração, de forma transparente, na arquitetura FT CORBA [Lau00a].

Bibliografia

[Chandra96] T. Chandra, S. Toueg, **"Unreliable Failure Detectors for Reliable Distributed Systems"**, JACM, 43(1): 225-267, March 1996.

[Felber98] P. Felber, **"The CORBA Object Group Service – A Service Approach to Object Groups in CORBA"**, PhD. Thesis, École Polytechnique Fédérale de Lausanne, Lausanne, 1998.

[Fischer85] M. J. Fischer, N. A. Lynch, M. S. Paterson, **"Impossibility of Distributed Consensus with One Faulty Process"**, Journal of the ACM, 32(2): 374-382, Apr 1985.

[Hurfin99] M. Hurfin, R. Macêdo, M. Raynal, F. Tronel, **"A General Framework to Solve Agreement Problems"**, 18th IEEE Symp. on Reliable Distributed Systems - SRDS'99, Lausanne, Suiça, October 1999.

[Lau99] Lau L., J. Fraga, J-M. Farines, M. Ogg, A. Ricciardi, **"CosNamingFT – A Fault-Tolerant CORBA Naming Service"**, 18th IEEE Symp. on Reliable Distributed Systems - SRDS'99, Lausanne, Suiça, October 1999.

[Lau00] Lau L., J. Fraga, J-M. Farines, **"Experiências com Comunicação de Grupo nas Especificações Fault Tolerant CORBA"**, SBRC'2000, SBC, Belo Horizonte – MG. Maio de 2000.

[Lau00a] Lau L., J. Fraga, J-M. Farines, **"Adapting the Fault-Tolerant CORBA Specifications for Large Scale Networks"**, Documento Interno 07-00. Em preparação.

[Moser98] L. E. Moser, P. M. P. Melliar-Smith, P. Narasimhan, **"Consistent Object Replication in the Eternal System"**, Theory and Practice of Object Systems, 4(2): 81-92, 1998.

[OMG96] Object Management Group, **"The Common Object Request Broker 2.0/IIOP Specification"**, Revision 2.0, OMG Document 96-08-04, 1996.

[OMG99] Object Management Group, **"Fault-Tolerant CORBA"**, Joint Revised Submission Document orbos/99-12-08, December 1999.

[Ricciardi91] A. Ricciardi and K. Birman, **"Using Process Groups to Implement Failure Detection in Asynchronous Systems"**. In Tenth ACM Symposium on the Principles of Distributed Computing. August 1991.

[Chung98] P. E. Chung, Y. Huang, S. Yajnik, D. Liang, J. Shih, **"DOORS: Providing Fault Tolerance for CORBA Applications"**, in poster session of Middleware ' 98, Sept. 1998.