GroupPac 3: Estendendo o FT-CORBA para Gerenciamento e Replicação Ativa*

Alysson Neves Bessani[1 †], Lau Cheuk Lung[2] ,
Eduardo Adílio Pelinson Alchieri[1 ‡], Joni da Silva Fraga[1 §]

[1]DAS - Departamento de Automação e Sistemas
UFSC - Universidade Federal de Santa Catarina
Campus Universitário, Caixa Postal 476 - CEP 88040-900 - Trindade - Florianópolis - SC

[2]PPGIA - Programa de Pós-Graduação em Informática Aplicada
PUC-PR - Pontifícia Universidade Católica do Paraná
R. Imaculada Conceição, 1155 - Prado Velho - CEP 80215-901 - Curitiba -PR

{neves,alchieri,fraga}@das.ufsc.br, lau@ppgia.pucpr.br

Resumo. *O FT-CORBA da OMG adota nas suas especificações quatro técnicas de replicação para tolerância a faltas de objetos: Sem Estado; Passiva Fria; Passiva Morna; e, por fim, a Replicação Ativa. Apesar da replicação ativa estar prevista no FT-CORBA, seus mecanismos não foram devidamente especificados devido à falta de definições para comunicação de grupo. Neste artigo, propomos um conjunto de estensões a estas especificações para melhorar o suporte da arquitetura à replicação ativa e ao gerenciamento de replicação.*

1. Introdução

O padrão FT-CORBA surgiu de um esforço da OMG (*Object Management Group*) no sentido de especificar uma arquitetura para objetos distribuídos tolerantes a falhas [Object Management Group, 2002]. Este padrão introduz suporte a tolerância a faltas na arquitetura CORBA fazendo uso de técnicas de replicação de objetos. Este suporte é construído a partir de um conjunto de objetos de serviço usados no gerenciamento de objetos de aplicação replicados. A tolerância a faltas fornecida usando os serviços definidos pelo padrão está fundamentada em premissas de falhas de parada (*crash*).

A especificação FT-CORBA define quatro tipos de replicação para os grupos de objetos: **Sem Estado** (estilo de replicação em que o estado dos objetos não é alterado pelas requisições); **Passiva Fria** (replicação com apenas um objeto ou réplica primária que grava periodicamente informações de estado em meio persistente); **Passiva Morna** (funciona de maneira semelhante ao anterior com a diferença de que os *checkpoints* da réplica primária são difundidos entre as réplicas secundárias); e, por fim, a **Replicação Ativa** (onde todas as réplicas executam as requisições do cliente).

O único tipo de replicação que é previsto e não tem seus mecanismos necessários especificados é a replicação ativa. A especificação atual do FT-CORBA permite que as abstrações definidas para a gestão de replicações possam ser também usadas com réplicas ativas, desde que suportadas em suas necessidades de comunicação por ferramentas proprietárias[1]. Assim, as requisições são difundidas, através de difusão atômica

*Realizado com recursos do CNPq (projeto número 401802/2003-5).
†Bolsista PGI/CNPq.
‡Bolsista PIBIC/CNPq.
§Bolsista de Produtividade em Pesquisa do CNPq - Nível 2.
[1]Proprietárias no sentido de não serem baseadas em padrões abertos.

[Hadzilacos and Toueg, 1994, Défago et al., 2000], a todos os objetos membros do grupo (réplicas ativas) usando meios externos ao ORB. Neste caso, se considerarmos as necessidades de suporte de uma replicação ativa, do ponto de vista conceitual, diríamos que o FT-CORBA fornece os mecanismos para controle de *membership* do grupo formado pelas réplicas ativas e a ferramenta proprietária fornece o suporte para comunicação de grupo. Alguns pontos que a especificação da OMG não deixa claro são como integrar essa ferramenta proprietária na arquitetura CORBA e qual a arquitetura de implantação dos serviços definidos, entre outros.

Este trabalho apresenta um conjunto de estensões propostas à arquitetura FT-CORBA para um melhor suporte a replicação ativa e gerenciamento de grupos de réplicas. São ao todo três novos mecanismos que visam melhorar a arquitetura de tolerância a faltas definida para o padrão CORBA, sem, no entanto, ferir qualquer aspecto já definido nas especificações atuais. O primeiro mecanismo a ser apresentado é a definição, gravação e recuperação de grupos em arquivos XML [Bray et al., 2004]. Este mecanismo visa definir um modelo para a representação de grupos de réplicas FT-CORBA e dar suporte a gravação do estado de um grupo (propriedades e estados das réplicas) para posterior recuperação. O segundo mecanismo definido, chamado célula de tolerância a faltas, é um pequeno servidor que visa dar suporte às réplicas. Grande parte dos serviços da especificação atual são ativados dentro deste servidor. A última (e mais importante) contribuição deste trabalho é um *framework* para a integração de ferramentas de comunicação de grupo na arquitetura FT-CORBA visando dar suporte à replicação ativa. Através deste *framework* é possível o uso de qualquer sistema de comunicação de grupo na arquitetura FT-CORBA.

As estensões propostas foram implementadas no sistema GROUPPAC [Lung et al., 2001]. Este sistema é uma implementação completa do padrão FT-CORBA desenvolvida pelo grupo de sistemas distribuídos do LCMI/DAS/UFSC. Inicialmente desenvolvido para dar suporte a sistemas de larga escala, o GROUPPAC foi uma das primeiras implementações do padrão FT-CORBA desenvolvidas, e ainda hoje é uma das poucas disponibilizadas livremente (*http://grouppac.sf.net*).

O texto está organizado da seguinte forma: na seção 2 temos uma breve descrição dos principais aspectos das especificações FT-CORBA. A seção 3 apresenta nossas estensões para gerenciamento de grupos, seguida pela seção 4, onde a célula de tolerância a faltas tem seu conceito e arquitetura apresentados. Na seção 5 é apresentado detalhadamente o *framework* de integração de sistemas de comunicação de grupo ao FT-CORBA. Nesta seção são descritos ainda os *plugins* de integração disponíveis atualmente para o GROUPPAC. Finalmente, a seção 6 relata alguns experimentos similares da literatura e a seção 7 apresenta as conclusões do trabalho.

2. As Especificações FT-CORBA

A arquitetura **FT-CORBA** [Object Management Group, 2002] define os serviços, em nível de *middleware*, responsáveis por prover as funcionalidades básicas para aplicações tolerantes a faltas através da replicação de objetos CORBA. Esses serviços estão divididos em três módulos básicos:

- **Gerenciamento de Replicação (SGR):** Este é o serviço responsável pelo ciclo de vida dos grupos. Duas funcionalidades são oferecidas no mesmo: Gerenciamento de Propriedades, onde as características funcionais da replicação (tipo de replicação usada, número mínimo de réplicas, etc.) são definidas, e o Gerenciamento de Grupos, que oferece mecanismos para a criação de membros (através de Fábricas de Objetos) e para o controle de *membership* dos grupos definidos;

- **Gerenciamento de Falhas (SGF):** É o serviço responsável pela detecção, notificação, análise e diagnóstico de falhas. Este serviço trabalha em conjunto com o SGR para que este último mantenha um *membership* sempre atualizado dos grupos;
- **Gerenciamento de Recuperação e Logging (SRL):** O SRL é responsável pela consistência de estados das réplicas. Este serviço define mecanismos para a recuperação de réplicas e do próprio grupo. Entre estes mecanismos está um suporte para construção de *logs* usados no armazenamento de requisições enviadas ao grupo. Este serviço também fornece mecanismos para atualização de membros através de *checkpoints*. Além disso também é responsabilidade do SRL atuar na recuperação de réplicas faltosas através da atualização de seus estados.

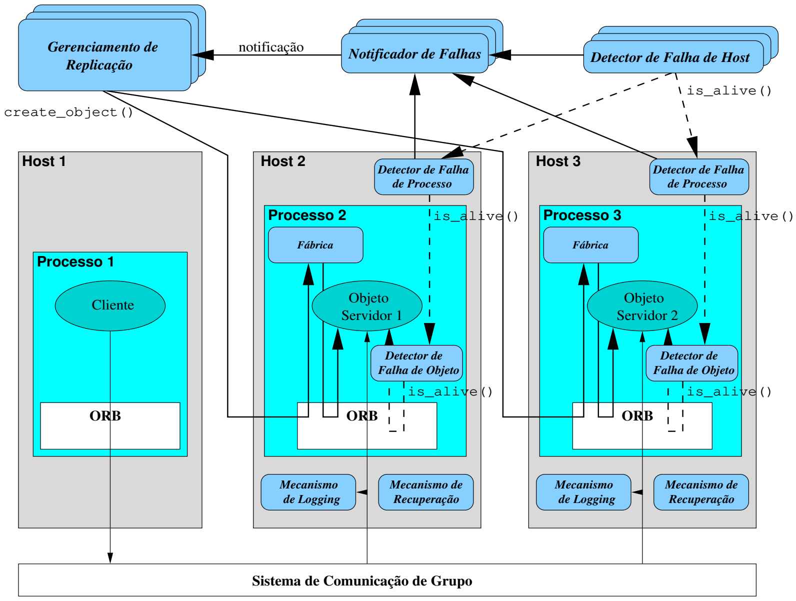

Figura 1: Arquitetura FT-CORBA.

Na figura 1, que ilustra a arquitetura FT-CORBA, não é explicitado suporte dos serviços FT-CORBA no *host* 1. Isto evidencia que, para o cliente, a replicação do servidor de aplicação é completamente transparente, a não ser pelo tratamento da referência de grupo (ver adiante). A criação de membros do grupo é feita através do gerenciador de replicação (SGR), que delega esta funcionalidade chamando, no *host* correspondente, o método remoto `create_object()` do objeto fábrica que é o responsável local pela criação e ativação de novos membros do grupo. Ainda nesta figura, os *hosts* 2 e 3 contêm membros do grupo de réplicas criados a partir das fábricas ativas nos mesmos.

A detecção de falhas no FT-CORBA (SGF) segue uma hierarquia, de acordo com três níveis de abstrações: objeto, processo e *host*. Em cada processo que mantém membros de um grupo existe um detector de falhas de objetos. Este detector verifica periodicamente, através da chamada de método `is_alive()`, se os objetos em seu processo ainda estão em atividade. Para nível de *host* é definido o detector de falhas de processos que faz o monitoramento dos processos do *host*. Por fim, em nível de sistema, completando a hierarquia, ocorre a detecção de falhas em *hosts*. Os detectores de falhas de *hosts* enviam, periodicamente, requisições aos detectores de falhas de processos (ativando também o método `is_alive()` destes últimos), para verificar se os mesmos e, por conseqüência, seus *hosts* ainda estão ativos.

Quando qualquer um dos detectores presentes no sistema observam uma parada em um objeto, processo ou *host*, esta é sinalizada ao notificador de falhas (ver figura 1), que a repassa ao SGR para que este atualize o *membership* do grupo. Se a falha ocorre no membro primário de uma replicação passiva, cabe ao mecanismo de recuperação escolher um novo primário do grupo de objetos e atualizar seu estado de tal forma que este possa continuar processando requisições do ponto onde o objeto primário faltoso parou. A fim de obter uma melhor confiabilidade nos sistemas que usam suporte FT-CORBA, os objetos de serviço (gerenciador de replicação, notificador de falhas e detector de falhas de *host*) devem ser replicados.

Conforme já citado, a especificação FT-CORBA não define um suporte de comunicação de grupo, indispensável na replicação ativa: não existem interfaces e nem tampouco um protocolo padronizado para tal. As especificações apenas sugerem que se utilize qualquer suporte proprietário para este fim. Na figura 1, este suporte de comunicação aparece como camada subjacente, abaixo da camada de *middleware* e, portanto, dos objetos de serviço FT-CORBA.

Em relação à interoperabilidade com grupos onde objetos são mantidos por diferentes ORBs que implementam o FT-CORBA, a OMG criou um formato padronizado para a referência de grupo de objetos: IOGR (*Interoperable Object Group Reference*). A referência IOGR consiste basicamente em um conjunto de perfis IIOP (concretização do GIOP sobre TCP/IP) que identificam cada membro do grupo sendo o membro primário definido a partir da presença da *tag* `TAG_PRIMARY` em seu perfil.

3. Gerenciamento: Definição e Persistência de Grupos em XML

O gerenciador de replicação definido nas especificações FT-CORBA tem um componente para o gerenciamento de propriedades dos grupos. Estas propriedades são sempre definidas durante a criação de um grupo, e algumas delas podem ser alteradas durante o ciclo de vida do mesmo. A gestão destas propriedades é feita através da interface `PropertyManager`, e serve como um bom ponto de entrada para mecanismos de gerência mais amigáveis como interfaces gráficas e programas que carregam arquivos que definem grupos.

O GROUPPAC têm uma ferramenta de administração que serve para a criação de grupos de réplicas e sua gestão. A interface desta ferramenta é apresentada na figura 2.

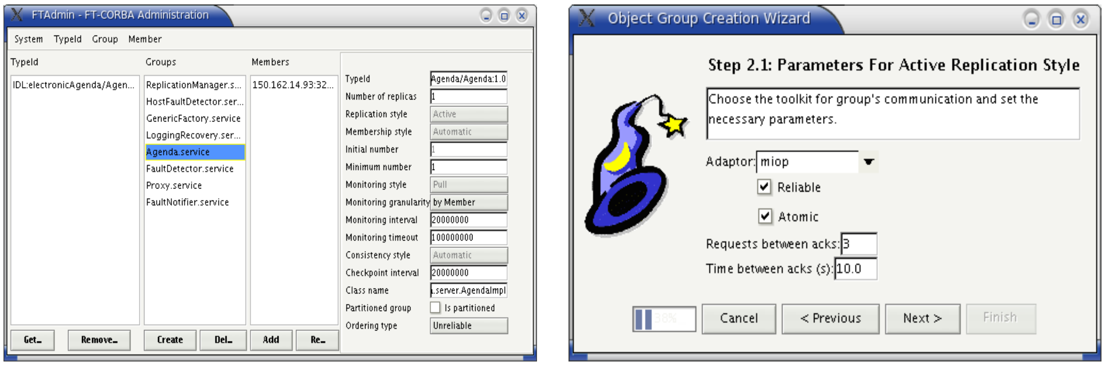

Figura 2: Ferramenta de administração e *wizard* de criação de grupo.

A criação de grupos via esta ferramenta é feita passo a passo através do preenchimento de um conjunto de formulários, o que nem sempre é agil. Portanto definiu-se um mecanismo para a especificação de grupos em XML que, além de permitir a criação de grupos dos mais diferentes tipos (permitindo a especificação de todas as propriedades definidas pela OMG mais algumas estensões), permita a gravação de um grupo em arquivo para sua posterior recuperação. Nesta gravação temos o armazenamento dos membros (localização, estado e se é o primário) e das propriedades do grupo.

O mecanismo de gerência de grupos implementado é definido em três módulos básicos:

- **XML Schema:** Um XML Schema [Fallside, 2001] define um conjunto de regras para a definição de um documento XML. Estas regras descrevem a estrutura esperada do documento, sendo portanto semelhante a uma gramática. Em nossa implementação optou-se pela utilização de XML Schema ao invés de DTD (*Document Type Definition*) [Bray et al., 2004] devido ao maior poder de expressão do primeiro;
- **Interpretador:** Este módulo define duas classes usadas para a leitura, validação e interpretação do arquivo XML que descreve um grupo. Uma vez as informações interpretadas, a criação do grupo é feita através da interação com o gerenciador de replicação;
- **Capturador de Estado:** Este componente é responsável pela captura do estado do sistema (número de membros, suas localizações e seus estados). Por ser executado apenas em tarefas administrativas, espera-se que a aplicação seja parada antes de se realizar uma gravação do estado do grupo.

A figura 3 apresenta o XML para a criação de um grupo.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <ft-service name="Agenda" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4                           xsi:noNamespaceSchemaLocation="ft-schema.xsd">
5
6    <description>Simple replicated service for ft-corba</description>
7
8    <type-id>IDL:electronicAgenda/Agenda:1.0</type-id>
9
10   <replication-style type="Active">
11      <property name="adaptor-factory"
12                value="br.ufsc.das.grouppac.scg.impl.umiop.UMIOPAdaptorsFactory"/>
13      <property name="reliable" value="true"/>
14      <property name="atomic" value="true"/>
15      <property name="n" value="3"/>
16      <property name="t" value="10"/>
17   </replication-style>
18
19   <membership-style type="Automatic">
20     <initial-number>1</initial-number>
21      <minimum-number>1</minimum-number>
22      <class-name>electronicAgenda.server.AgendaImpl</class-name>
23   </membership-style>
24
25   <monitoring-style type="Pull">
26      <granularity>by member</granularity>
27      <interval>2</interval>
28      <timeout>10</timeout>
29   </monitoring-style>
30
31   <consistency-style type="Automatic">
32      <interval>2</interval>
33   </consistency-style>
34</ft-service>
```

Figura 3: Especificação em XML de um grupo de réplicas para o FT-CORBA.

Nesta figura temos alguns pontos que devem ser destacados. Em primeiro lugar, o grupo é descrito por um elemento raiz `ft-service` identificado por um nome (neste caso “Agenda”). Dentro deste elemento temos vários outros elementos importantes, como por exemplo o `replication-style` (linhas 10-17). Este elemento define o tipo de replicação (no exemplo é ativa) e pode conter propriedades a serem usadas como critério na criação do grupo. Em nosso exemplo, definimos a fábrica de adaptadores usada pelo SCG (ver seção 5) e uma série de propriedades a serem passadas para este adaptador.

Os demais elementos da figura 3 dizem respeito a outras propriedades definidas no padrão FT-CORBA, apresentadas no capítulo 23 das especificações [Object Management Group, 2002].

4. CTF: Célula de Tolerância a Faltas

A fim de agrupar todos os elementos necessários para a ativação de um objeto tolerante a faltas foi criado um pequeno servidor capaz de oferecer todo o suporte para uma réplica funcionar corretamente. Este servidor, chamado **célula de tolerância a faltas** (CTF), é uma espécie de *container* onde as réplicas convivem. A figura 4 apresenta sua arquitetura.

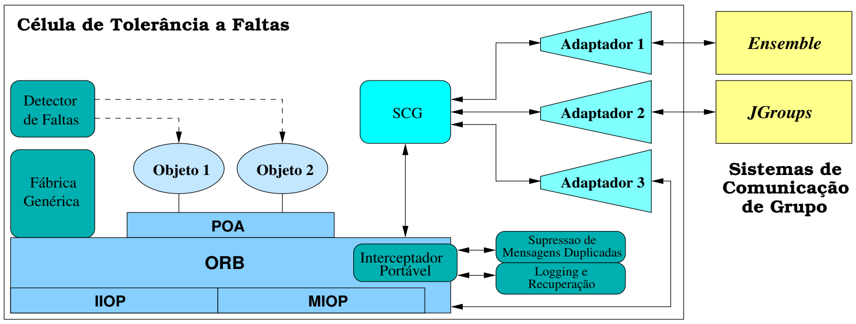

Figura 4: Arquitetura da Célula de Tolerância a Faltas.

Nesta figura estão representados os vários elementos necessários em um processo que mantenha réplicas de grupos. O elemento central da CTF é o ORB, e sobre ele são ativados todos os demais elementos. Cada célula CTF suporta no máximo uma réplica de cada tipo[2]. Estas réplicas são sempre criadas pela fábrica genérica, que utiliza as capacidades de reflexão computacional do Java para a execução desta tarefa. Uma vez criadas, as réplicas são monitoradas pelo detector de falhas (de objeto) que verifica o funcionamento das mesmas, reportando erros ao notificador de faltas.

A comunicação com a réplica pode se dar de duas formas, dependendo do tipo de replicação usado no grupo. Se usado o modelo de replicação ativa ou ativa com votação, a comunicação é feita através do *framework* SCG, que permite a integração da arquitetura FT-CORBA a sistemas de comunicação de grupo[3] que possam prover serviços de difusão atômica [Hadzilacos and Toueg, 1994], um requisito básico para a replicação ativa [Schneider, 1990]. Se o estilo de replicação do grupo for sem estado, passiva ou passiva morna, as mensagens são trocadas via IIOP, usando apenas mecanismos padrão do ORB. Note-se que em todos os tipos de replicação a mensagem passa pelo interceptador portável, que executa duas tarefas: *logging* das mensagens e supressão de mensagens duplicadas, evitando que um objeto execute a mesma requisição duas vezes.

Vale ressaltar que nem todos os elementos apresentados na figura 4 são ativados de uma única vez. Quando o SCG é necessário (replicação ativa e ativa com votação) os serviços de *logging* e recuperação não são (já que eles só são úteis em replicação passiva), e vice versa.

[2]O tipo corresponde ao tipo da interface, definido em IDL, chamado nas especificações de *type_id*.

[3]Podem ser construídos adaptadores que se baseiam apenas em serviços de comunicação do ORB, como o apresentado em [Bessani et al., 2004]

5. SCG: Integração de Ferramentas de Comunicação de Grupo ao FT-CORBA

O modelo de replicação ativa, proposto para o FT-CORBA, baseia-se na idéia de um grupo fechado de réplicas acessado por clientes leves através do mecanismo de comunicação usual do CORBA (chamada de métodos remotos via IIOP). A especificação FT-CORBA também prevê a utilização de ferramentas de comunicação de grupo proprietárias para difusão de mensagens com diferentes níveis de qualidade de serviço [Object Management Group, 2002]. Entretanto, estas mesmas especificações não definem como estas ferramentas de comunicação de grupo serão integradas na arquitetura FT-CORBA.

Diante disto foi criado o *framework* de **suporte de comunicação de grupo** (SCG). O SCG é parte integrante do GROUPPAC e define um mecanismo padrão através do qual a infra-estrutura FT-CORBA interage com diferentes ferramentas de comunicação de grupo visando prover serviços de comunicação com as propriedades necessárias para a replicação ativa. Através deste mecanismo é possível a integração de qualquer suporte de comunicação de grupo à arquitetura do FT-CORBA através da construção de adaptadores que são integrados ao GROUPPAC através de *plugins*. A figura 5 apresenta este modelo.

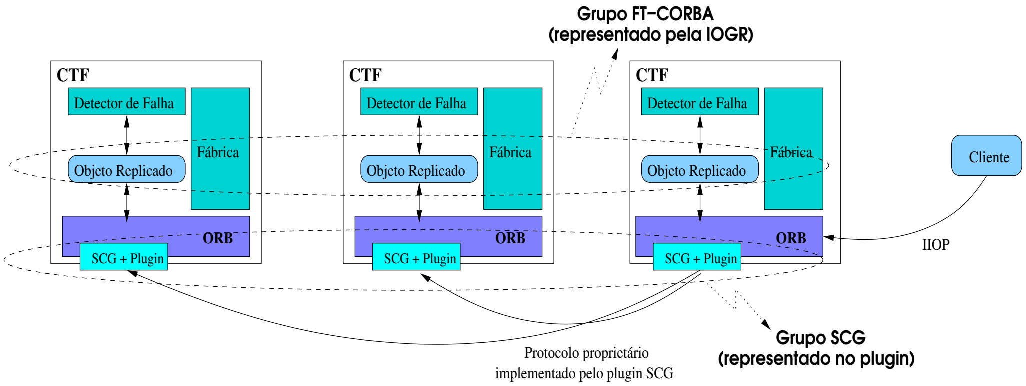

Figura 5: Replicação Ativa no GROUPPAC.

Nesta figura, o cliente, de posse da referência de grupo (IOGR), envia uma requisição ponto a ponto (através do IIOP) a um dos objetos listados na IOGR. O receptor desta requisição serve de ponte para acesso aos protocolos de comunicação de grupo (encapsulados nos *plugins* e ferramentas de comunicação de grupo). Estes protocolos são executados atendendo ao requisito de que todas as réplicas do serviço executem a mesma sequência de requisições [Schneider, 1990]. As respostas, se houverem, são enviadas de volta à "ponte", que retorna o resultado ao cliente[4]. Portanto, é através destes elementos "ponte" e de *plugins* que o grupo fechado definido pelo FT-CORBA fica acessível a clientes IIOP. Caso a réplica "ponte" falhe em algum ponto desse processamento, um *timeout* ocorre no cliente que reenvia a requisição a outro objeto cuja referência está contida na IOGR.

O SCG foi concebido através da instanciação de vários padrões de projeto [Gamma et al., 1995] utilizados em três componentes:

- ***Proxy* de Requisições:** Este componente recebe todas as requisições enviadas ao grupo as repassa ao *plugin* de comunicação de grupo. Sua implementação é baseada no conceito de servidores genéricos, utilizando os mecanismos

[4]Se o tipo de replicação for ativa, a primeira resposta devolvida por uma réplica é retornada ao cliente. Já no caso da replicação ativa com votação, todas as respostas de réplicas não faltosas são recebidas para então o resultado ser computado e devolvido.

de DSI (*Dynamic Skeleton Interface*), previsto nas especificações CORBA [Object Management Group, 2002];

- **Invocador de Requisições:** Este é o componente responsável por passar as requisições recebidas do *plugin* de comunicação de grupo para a réplica do grupo invocado. A implementação deste componente emula um *stub* para a invocação dos métodos da réplica sem o *overhead* do ORB;
- **Suporte ao *Plugin* de Comunicação de Grupo:** Este componente compreende as classes e interfaces de suporte a *plugins* de ferramentas de comunicação de grupo. Basicamente, temos um mecanismo para a carga dinâmica de *plugins* e algumas interfaces usadas como base na implementação destes.

A figura 6 apresenta um pequeno diagrama em UML com as diversas classes e interfaces que compõem o *framework*. Nesta figura estão representadas as três interfaces que devem ser implementadas para a concretização de um *plugin* SCG: `AdaptorsFactory` (criação de adaptadores), `SenderAdaptor` (adaptador *proxy*/*plugin*) e `ReceiverAdaptor` (adaptador *plugin*/invocador).

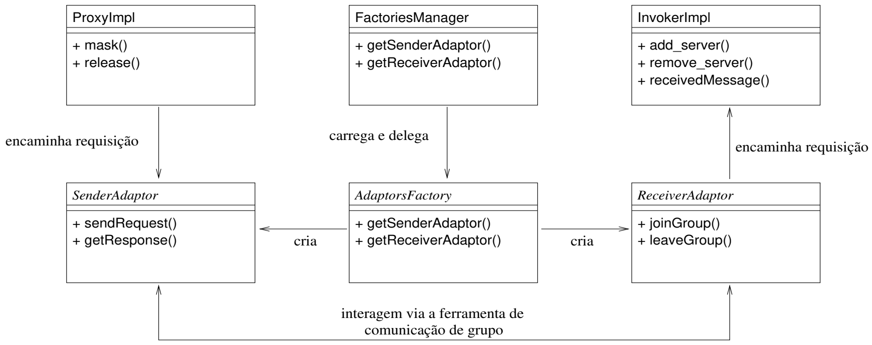

Figura 6: *Framework* SCG.

O grande "truque" empregado no *framework* SCG para conseguir a utilização da ferramenta de comunicação de grupo é a troca das referências dos membros presentes na IOGR por referências a *proxies* para o grupo[5]. A figura 7 apresenta um diagrama de seqüência em UML que apresenta como ocorre a criação de um grupo para replicação ativa.

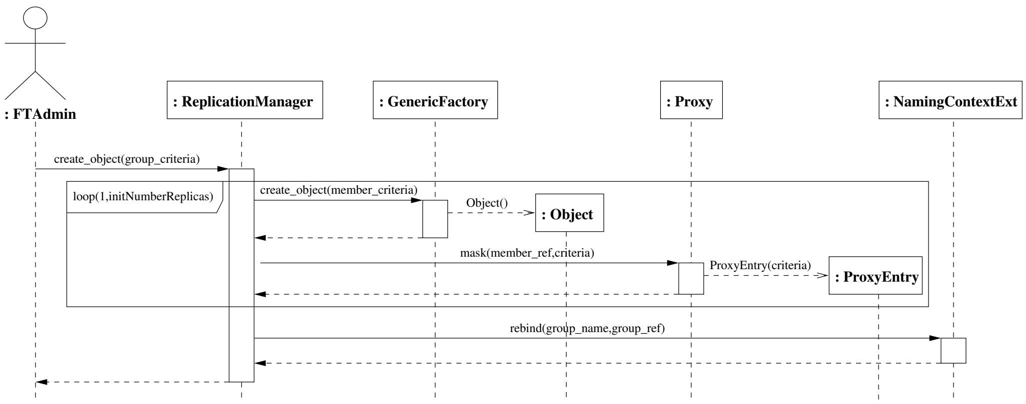

Figura 7: Seqüência de operações para a criação de um grupo.

Nesta figura temos um administrador criando um grupo passando como parâmetro uma série de critérios para o gerenciador de replicação (classe `ReplicationManager`). Este, por sua vez, executa, para cada réplica a ser criada, uma chamada à `GenericFactory` que cria a réplica (observando os critérios de criação) em sua localização

[5]As especificações permitem este tipo de artifício [Object Management Group, 2002].

e uma chamada ao `Proxy` para que este crie uma entrada (`ProxyEntry`) que possa mascarar esta réplica[6]. Dai em diante o gerenciador monta a IOGR de grupo (contendo as referências para os *proxies* criados) e a disponibiliza no serviço de nomes tolerante a faltas. O processamento se encerra com a IOGR sendo devolvida ao administrador.

Uma vez que o cliente possui a IOGR, este pode realizar invocação de operações nas réplicas através dos *proxies*. Conforme já apresentado na figura 5, esta invocação é feita em dois passos de comunicação: cliente-*proxy* via IIOP e *proxy*-grupos de réplicas via o sistema de comunicação de grupo. O diagrama de seqüência da figura 8 apresenta esse processamento.

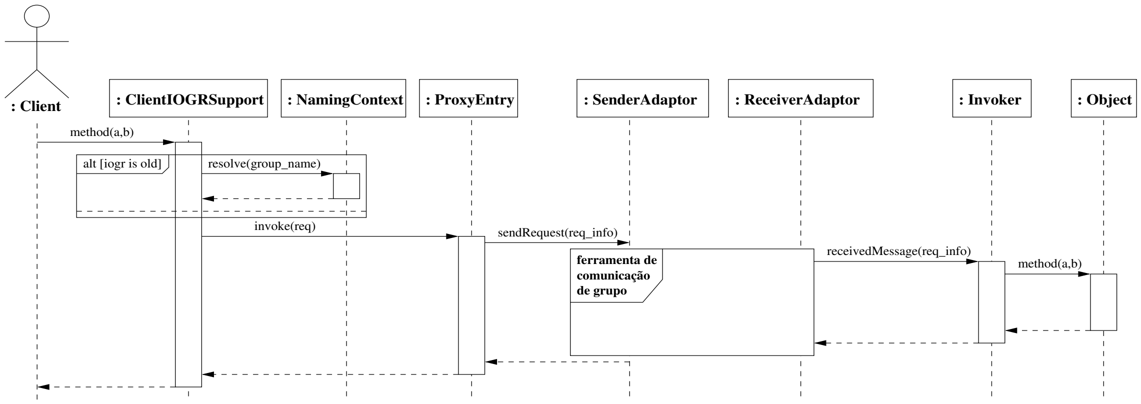

Figura 8: Seqüência de operações para a execução de uma operação.

Na figura 8 um cliente invoca a operação `method(a,b)` em um grupo que implementa a IDL onde a mesma está definida. O primeiro passo a ser executado é a verificação por parte do interceptador portável `ClientIOGRSupport` instalado no cliente. Se a IOGR usada para a invocação não é a versão mais atual para esse grupo o cliente faz esta atualização através do serviço de nomes. A seguir, uma das referências presentes na IOGR é escolhida, e a requisição é enviada ao `ProxyEntry` correspondente (que mascara uma das réplicas). O `ProxyEntry` é um servidor dinâmico que pode atender a qualquer tipo de método (via DSI). Após receber a requisição ele verifica qual ferramenta de comunicação de grupo deve ser usada para esta IOGR e usa o adaptador cliente apropriado (classe que implementa `SenderAdaptor`) para invocar tal ferramenta. A ferramenta difunde a requisição invocada usando um protocolo de difusão atômica implementado pela ferramenta de comunicação de grupo. Esta difusão acaba por ser recebida (na mesma ordem) por todos os adaptadores receptores (implementação da interface `ReceiverAdaptor`). Estes a entregam ao `Invoker` que realiza a invocação do método na réplica. A resposta à requisição (se houver) segue pelo caminho inverso, conforme mostrado na figura.

5.1. *Plugins* Disponíveis para o SCG

Atualmente estão implementados *plugins* SCG para as ferramentas ENSEMBLE [van Renesse et al., 1998] e JGROUPS [JGroups, 2004] e para o protocolo baseado nos serviços de comunicação do ORB MJACO [Bessani et al., 2002] apresentado em [Bessani et al., 2004]. Esta subseção apresenta uma breve descrição destes *plugins* e seus adaptadores.

Os *plugins* ENSEMBLE e JGROUPS são muito parecidos entre si. Eles implementam apenas uma “casca” bastante simples que transforma requisições CORBA em men-

[6]Por questões de simplicidade, o diagrama apresenta apenas uma `GenericFactory` e um `Proxy` que são chamados uma vez para cada réplica criada. Na verdade, estas chamadas ocorrem em fábricas e *proxies* diferentes que executam em diferentes localizações.

sagens e as envia através da API desses sistemas de comunicação de grupo (adaptador emissor). A recepção destas mensagens é implementada através dos adaptadores receptores e classes auxiliares que implementam interfaces de *callback*[7]. Tanto o ENSEMBLE quanto o JGROUPS trabalham com microprotocolos que são agrupados em pilhas para oferecer a funcionalidade desejada, que, no nosso caso, é a replicação ativa.

Já o *plugin* MJACO utiliza os serviços de comunicação providos por este ORB para a construção de um protocolo de difusão confiável com ordem total. Este protocolo explora o modelo de objetos definido no padrão UMIOP [Object Management Group, 2001], que permite o acesso a vários objetos a partir de uma única referência de grupo via *multicast* IP e o serviço de *membership* do FT-CORBA, dando suporte assim a um serviço de replicação ativa inteiramente baseado em padrões da OMG.

A idéia básica para a construção deste *plugin* foi refletir o grupo FT-CORBA, representado pela IOGR, em um grupo UMIOP, usado no contexto do SCG, através de uma referência de grupo UMIOP (para difusão não confiável). Para tanto foi definido um algoritmo que implementa difusão com ordem total baseado nas premissas do FT-CORBA: comunicação ponto a ponto confiável (pilha IIOP/TCP/IP), comunicação multiponto não confiável (pilha MIOP/UDP/*multicast* IP) e serviço de *membership* com propriedades fortes como as definidas em [Ricciardi and Birman, 1991] (implementado pelo `ReplicationManager`). Este protocolo, construído em camadas, usa um algoritmo de difusão confiável extremamente leve sob um outro de difusão atômica baseado no paradigma do ordenador fixo, onde um processo do grupo é o responsável por definir a ordem total de todas as mensagens [Défago et al., 2000].

Uma análise do desempenho destes três *plugins* (em termos de tempos de resposta) pode ser encontrada em um artigo complementar a este [Bessani et al., 2004].

6. Trabalhos Relacionados

Na literatura, são encontrados diversos trabalhos envolvendo a disponibilidade de serviço de comunicação de grupo e tolerância a faltas na arquitetura CORBA. Estas experiências (classificadas em três abordagens: integração [Maffeis, 1995, ISIS, 1995], serviço [Felber, 1998] e interceptação [da Silva Fraga et al., 1997, Moser et al., 1999, Lung et al., 2000]) se preocupam basicamente em manter características de portabilidade e de desempenho. Considerações sobre interoperabilidade eram limitadas, uma vez que estes trabalhos foram realizados antes da publicação das especificações FT-CORBA e UMIOP pela OMG.

A abordagem de integração consiste na construção ou na modificação de um *middleware* CORBA existente, adicionando mecanismos de comunicação de grupo. Nesta abordagem, o ORB é modificado para que as aplicações não possam distinguir objetos simples de grupos de réplicas, oferecendo, desta forma, um alto grau de transparência. A idéia principal nesta abordagem é que o processamento de grupo seja suportado por uma ferramenta de comunicação de grupo abaixo do núcleo do ORB. Todas as chamadas que envolvam comunicação ou gestão de grupo são repassadas pelo núcleo do ORB a este suporte de mais baixo nível.

A idéia básica na abordagem de serviço é prover a comunicação de grupo como um objeto de serviço sobre o ORB, e não como parte do próprio ORB. Dessa forma, o suporte de grupo deve ser formado por uma coleção de objetos de serviço, que podem estar distribuídos em diferentes estações da rede. Estes objetos são responsáveis pela

[7] Chamadas pelas ferramentas quando da chegada de uma mensagem.

gestão de grupos e pela entrega de mensagens aos objetos membros de grupo, mantendo as propriedades definidas pelo tipo de comunicação de grupo utilizado.

A abordagem de interceptação prevê que as requisições enviadas aos objetos servidores devem ser capturadas e desviadas para um sistema de comunicação de grupo, de maneira transparente para a aplicação. Para tal, podem ser utilizados estruturas das linguagens de programação [Lisbôa, 1997], interfaces do sistema operacional [Moser et al., 1999] ou mesmo mecanismos do próprio ORB [da Silva Fraga et al., 1997, Lung et al., 2000], conhecidos como interceptadores.

A abordagem apresentada neste trabalho pode ser entendida como híbrida, já que combina características das abordagens de interceptação (usando interceptadores do CORBA) e serviço (objeto de serviço SCG). O uso dessa combinação permite uma melhor transparência dos mecanismos de comunicação de grupo: os clientes não precisam acessar o SCG diretamente como ocorre na abordagem de serviço.

7. Conclusão

O presente artigo apresenta um conjunto de estensões ao padrão FT-CORBA que visam melhorá-lo sem no entanto ferir as especificações correntes. Três pontos básicos são estendidos: gerenciamento de grupo, onde um mecanismo de gravação e recuperação do estado de grupos é proposto a partir de uma estrutura de representação de grupos de réplicas em XML; arquitetura do servidor tolerante a faltas, onde os serviços definidos na especificação são agrupados em um pequeno servidor de aplicações que suporta a execução de réplicas; e, finalmente, o suporte a replicação ativa, onde criamos o SCG para a integração de qualquer ferramenta de comunicação de grupo à infra-estrutura de tolerância a faltas. Estas estensões tornam o GROUPPAC um dos suportes de tolerância a faltas mais avançados (senão o mais) em termos de objetos distribuídos.

8. Agradecimentos

Agradecimentos especiais à Ricardo Sangoi Padilha e Luciana de Sá Moreira (pesquisadores do DAS/UFSC) pela implementação original do GROUPPAC e do SCG.

Referências

Bessani, A. N., da Silva Fraga, J., and Lung, L. C. (2002). MJaco: Integração do multicast IP na arquitetura CORBA. In *Anais do XX Simpósio Brasileiro de Redes de Computadores - SBRC'2002*, Buzios, RJ.

Bessani, A. N., da Silva Fraga, J., Lung, L. C., and Alchieri, E. A. (2004). Replicação ativa no CORBA: Padrões, protocolos e framework de implementação. In *Anais do XXII Simpósio Brasileiro de Redes de Computadores - SBRC'2004*, Gramado, RS.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., and Yergeau, F. (2004). Extensible markup language (XML) 1.0 (third edition). Recomendation, World Wide Web Consortium, http://www.w3.org/TR/REC-xml.

da Silva Fraga, J., Maziero, C. A., Lung, L. C., and Filho, O. G. L. (1997). Implementing replicated services in open systems. In *Proceedings of the 3rd IEEE International Symposium on Autonomous Descentralized Systems - ISADS'97*, pages 273–280, Lausanne, Suiça.

Défago, X., Schiper, A., and Urban, P. (2000). Totally ordered broadcast and multicast algorithms: a comprehensive survey. Technical Report TR DSC/2000/036, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland.

Fallside, D. C. (2001). XML Schema part 0: Primer. Recomendation, World Wide Web Consortium, http://www.w3.org/TR/xmlschema-0/.

Felber, P. (1998). *The CORBA Object Group Service - A Service Approach to Object Groups in CORBA*. Tese de doutorado, École Polytechnique Fédérale de Lausanne, Lausanne, Suiça.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading.

Hadzilacos, V. and Toueg, S. (1994). A modular approach to the specification and implementation of fault-tolerant broadcasts. Technical report, Department of Computer Science, Cornell University, New York - USA.

ISIS (1995). Isis distributed systems inc, iona technologies, ltd. - orbix+isis programmer's guide. Document D070-00.

JGroups (2004). Jgroups: A toolkit for reliable multicast communication. Disponível em http://www.jgroups.org.

Lisbôa, M. L. B. (1997). Arquiteturas de meta-nível. In *Anais do Simpósio Brasileiro de Engenharia de Software - SBES'97*, Fortaleza, CE.

Lung, L. C., da Silva Fraga, J., Farines, J. M., and Oliveira, J. R. (2000). Experiências com comunicação de grupo nas especificações fault tolerant corba. In *Anais do 18o. Simpósio Brasileiro de Redes de Computadores*, Belo Horizonte - MG - Brasil.

Lung, L. C., da Silva Fraga, J., Padilha, R., and Souza, L. (2001). Adaptando as especificações ft-corba para redes de larga escala. In *Anais do XIX Simpósio Brasileiro de Redes de Computadores - SBRC'2001*, Florianópolis, SC.

Maffeis, S. (1995). Adding group communication and fault-tolerance to CORBA. In *Proceedings of the USENIX Conference on Object Oriented Technologies*, pages 135–146, Monterey, Canada.

Moser, L. E., Melliar-Smith, P. M., Narasimhan, P., Tewksbury, L. A., and Kalogeraki, V. (1999). The eternal system: an architecture for enterprise applications. In *Proceedings of the 3rd International Enterprise Distributed Object Computing Conference (EDOC'99)*, Mannhein - Alemanha.

Object Management Group (2001). Unreliable multicast inter-orb protocol specification v1.0. OMG Standart ptc/03-01-11.

Object Management Group (2002). The common object request broker architecture: Core specification v3.0. OMG Standart formal/02-12-06.

Ricciardi, A. M. and Birman, K. (1991). Using process groups to implement failure detection in asynchronous environments. In *ACM Symposium on Principles of Distributed Computing*, pages 341–353, Montreal - Quebec - Canada.

Schneider, F. B. (1990). Implementing fault-tolerant service using the state machine aproach: A tutorial. *ACM Computing Surveys*, 22(4):299–319.

van Renesse, R., Birman, K. P., Mark Hayden, A. V., and Karr, D. (1998). Building adaptative systems using ensemble. *Software - Pratice and Experience*, 28(9):963–979.