5
Validação

Uma vez implementadas as três abordagens, vários testes foram executados a fim de se poder validar as soluções propostas. Os testes realizados objetivaram avaliar aspectos individuais de cada abordagem (como por exemplo: uso de memória, atraso de invocação, entre outros) bem como compará-las entre si. As próximas seções irão detalhar cada teste executado.

5.1
Gerenciamento de Memória

Todas as soluções propostas por este trabalho incluem um elemento procurador o qual serve potencialmente a vários ORBs que por sua vez dependem dele para o atendimento de suas requisições. Esse fato torna esses elementos peças-chave das abordagens, que por isso devem estar livres de erros tanto de projeto/programação como aqueles provocados por ORBs defeituosos ou mal-intencionados (ORBs que não respeitem o protocolo acordado). Lidou-se com esta última classe de erros através do uso de co-rotinas para isolar o tratamento de cada ORB. Ao ocorrer um erro devido a uma falha no cumprimento dos protocolos descritos neste trabalho, apenas os ORBs envolvidos são afetados, não havendo portanto impacto colateral nos outros ORBs que o procurador em questão está servindo.

No que diz respeito aos erros de programação que pudessem afetar o funcionamento dos procuradores, deu-se especial atenção ao gerenciamento de memória. O motivo dessa preocupação é que estes elementos fazem intenso uso de alocação dinâmica de memória. Portanto, ao serem postos em execução por um longo período, erros de gerenciamento de memória (*memory leak*) podem causar a falha dos procuradores, consequentemente afetando os ORBs por eles atendidos.

O teste para verificação se o gerenciamento de memória estava correto e robusto basicamente consistiu em fazer o procurador de cada abordagem tratar simultaneamente um número elevado de servidores e clientes, com estes últimos realizando um grande número de invocações remotas. Foram postos em execução 150 objetos servidores sendo atendidos por um mesmo procurador com 150 clientes executando um laço infinito e realizando invocações remotas aos objetos servidores. Este teste foi colocado em execução durante o período de 12 horas onde verificou-se (mediante o uso de utilitários de sistema operacional como o programa *top* [17]) uma quantidade relativamente constante de memória alocada por parte dos processos responsáveis pelos procuradores, apesar do elevado número de invocações remotas atendidas.

5.2
Concorrência

Um outro teste realizado objetivou validar a corretude dos procuradores no atendimento concorrente aos ORBs. Para isso um cenário de testes similar ao exposto na seção 5.1 foi usado, com 150 objetos CORBA representados por um mesmo procurador e 150 clientes realizando invocações remotas ininterruptas aos objetos CORBA escolhidos aleatoriamente. A interface implementada pelos objetos consistia de um incrementador extremamente simples. A sua IDL é mostrada a seguir:

```
interface MathOp {
  long inc(in long number);
};
```

A corretude das operações foi avaliada no lado do cliente, onde o resultado recebido devia ser igual ao parâmetro enviado acrescido de uma unidade. Foi desenvolvido um programa que tinha como função iniciar os objetos CORBA, os clientes, o procurador e que coletava os resultados infomando o sucesso da operação.

5.3
Interface com Outros ORBs

As abordagens Procurador TCP e Procurador HTTP não requerem modificações nos ORBs clientes. Assim realizou-se também um teste de conformidade/interoperabilidade utilizando ORBs não modificados diferentes do OiL como clientes de objetos CORBA protegidos por firewalls. Deste modo, visou-se testar a conformidade ao padrão CORBA tanto no código do procurador como no código modificado do ORB servidor.

A mesma interface apresentada na seção 5.2 foi utilizada e como ORBs clientes foram utilizados o ORBacus [18] e o JacORB [19]. Em ambos os casos não foi detectado nenhum erro.

5.4
Atraso de Invocação

Um importante teste realizado teve o objetivo de medir o atraso causado por cada abordagem na invocação remota. A infra-estrutura utilizada nos testes consistiu de duas redes locais interligadas por três roteadores. Em uma dessas redes existe um firewall que previne qualquer conexão iniciada por elementos externos às máquinas da rede interna. É nessa rede protegida que foram postos em execução os objetos CORBA, com seus respectivos clientes executando na outra rede. As máquinas da rede protegida possuiam a seguinte configuração: Processador Intel Pentium IV 1,70 GHz, 256 MB de RAM executando o sistema operacional Linux. As máquinas da rede que abrigam os clientes tinham como configuração: Processador Intel Pentium IV 2,80 GHz, 1 GB de RAM executando o sistema operacional Linux.

Foram criados cinco cenários de testes que estão descritos a seguir:

1. Objeto CORBA configurado com a abordagem OMG tendo como procurador de aplicação um elemento situado na própria rede local;
2. Objeto CORBA configurado com a abordagem Procurador TCP tendo como procurador um elemento situado na rede externa;
3. Objeto CORBA configurado com a abordagem Procurador HTTP tendo como procurador um elemento situado na rede externa. Intervalo de consulta: 0s;

4. Objeto CORBA configurado com a abordagem Procurador HTTP tendo como procurador um elemento situado na rede externa (rede aonde os clientes são executados). Intervalo de consulta: 1s;

5. Objeto CORBA sem nenhuma configuração, sendo o acesso pelos elementos externos permitido através da configuração manual do firewall.

O último cenário de teste foi criado para obter-se uma medida que pudesse ser utilizada para avaliar o atraso inserido por cada uma das três abordagem individualmente, comparando-as com uma situação onde o ORB não é modificado e a travessia é configurada manualmente no firewall.

Foram realizadas aproximadamente 5000 invocações para cada abordagem utilizando a IDL apresentada na seção 5.2. Os resultados obtidos podem ser verificados na tabela 5.1 com os tempos totais de invocação (*round-trip-delay*) medidos em segundos. As duas primeiras medidas representam o menor e o maior valor obtido em cada abordagem enquanto que as medidas seguintes representam a média e o desvio padrão do conjunto de amostras respectivamente.

	Medidas (s)			
Abordagem	Menor	Maior	Média	Desvio Padrão
OMG	0,0083	0,3707	0,0351	0,0518
Procurador TCP	0,0036	0,0605	0,0059	0,0011
Proc. HTTP (0s)	0,0073	0,0384	0,0122	0,0024
Proc. HTTP (1s)	0,8762	1,0235	1,0127	0,0139
Direto	0,0023	0,0697	0,0037	0,0017

Tabela 5.1: Resultados do Teste de Atraso de Invocação

Percebe-se que a abordagem Procurador TCP é aquela em que ocorre o menor atraso na média (0,0022 s $\approx$ 59%) e a que apresenta maior estabilidade com relação atraso imposto (por possuir o menor desvio padrão).

Comparando-se a abordagem Procurador TCP e a abordagem OMG percebe-se uma grande diferença e a razão para isto se deve basicamente ao maior número de conexões feitas e de mensagens trocadas na abordagem OMG, na qual, antes de ser realizada uma invocação remota, ocorre uma negociação de sessão entre os ORBs e os procuradores/firewalls. No caso do teste em questão duas conexões tiveram que ser abertas: uma entre o cliente e o procurador de aplicação e outra entre o procurador de aplicação e o ORB servidor. Quanto ao número de mensagens trocadas, além do envio

da mensagem GIOP de requisição, uma mensagem de negociação de sessão trafega nos dois sentidos entre o cliente e o procurador. No caso da abordagem Procurador TCP a conexão entre o ORB servidor e o procurador já se encontra aberta e não existe a mensagem de negociação de sessão trocada entre o cliente e o procurador. São esses dois pontos (conexão adicional a ser aberta e mensagem de negociação de sessão) os responsáveis por elevar substancialmente o atraso de invocação na abordagem OMG. Um outro ponto negativo em relação à abordagem OMG é o seu grande desvio padrão, indicando que esta solução é mais instável do que as outras no que diz respeito ao atraso imposto.

Conclui-se portanto pelo que foi verificado nos testes que a abordagem Procurador TCP obtém um maior desempenho às custas de uma falta de escalabilidade, visto que a diminuição do atraso de invocação se deve em grande parte à conexão mantida aberta entre o ORB servidor e o procurador, fato este que limita a escalabilidade do procurador ao número de conexões simultâneas permitidas pelo sistema operacional. Em contrapartida, na abordagem OMG este problema de escalabilidade não existe pois a conexão entre o procurador de aplicação e o ORB servidor não é mantida aberta durante toda a existência deste último.

Nota-se que até mesmo a abordagem Procurador HTTP (0 s) causou menor atraso do que a abordagem OMG e apresentou um desvio padrão menor. No entanto, não se pode deixar de mencionar o fato de que esta solução tem um custo maior no que diz respeito à largura de banda exigida, devido ao processo de consulta (*polling*) HTTP.

Apesar da abordagem Procurador TCP ter imposto, na média, um atraso adicional de 59% em relação à conexão direta, o valor absoluto do atraso pode ser considerado baixo. Observa-se inclusive que algumas invocações diretas foram até mesmo mais demoradas do que invocações utilizando a abordagem Procurador TCP. Acreditamos que o uso de mensagens com um número maior de parâmetros do que foi utilizado nestes testes irão provavelmente reduzir a razão entre o atraso causado pela abordagem Procurador TCP e a conexão direta. No entanto, a fim de se comprovar esta hipótese uma gama maior de testes com invocações de métodos mais complexos precisam ser feitos.

5.5
Escalabilidade

O último teste realizado teve como intuito medir a escalabilidade dos procuradores, procurando identificar uma função que relacionasse o atraso de invocação com o número de pares servidores/clientes atendidos por eles. Para isto foram utilizados duas redes interligadas por um roteador CISCO 7204 o qual proíbe a conexão de elementos externos a elementos de uma dessas redes através de uma lista de controle de acesso (ACL). Na rede protegida foram postos em execução 13 objetos servidores CORBA (um objeto para cada estação disponível) enquanto que na outra rede foi utilizado um aglomerado (*cluster*) de 54 nós para executar de 1 a 54 clientes. Estes faziam invocações remotas aos objetos CORBA com carga sobre os objetos servidores uniformemente distribuída. Foram realizados 3 cenários de testes, um para cada abordagem apresentada, e em todos eles usou-se um único procurador para todos os objetos CORBA. Tanto a rede protegida (onde os objetos CORBA foram postos em execução) quanto o aglomerado que hospedava os clientes tinham estações com a seguinte configuração: Processador Intel Pentium IV 1,70 GHz, 256 MB de RAM executando o sistema operacional Linux. A Figura 5.1 apresenta os resultados obtidos nos três testes realizados.

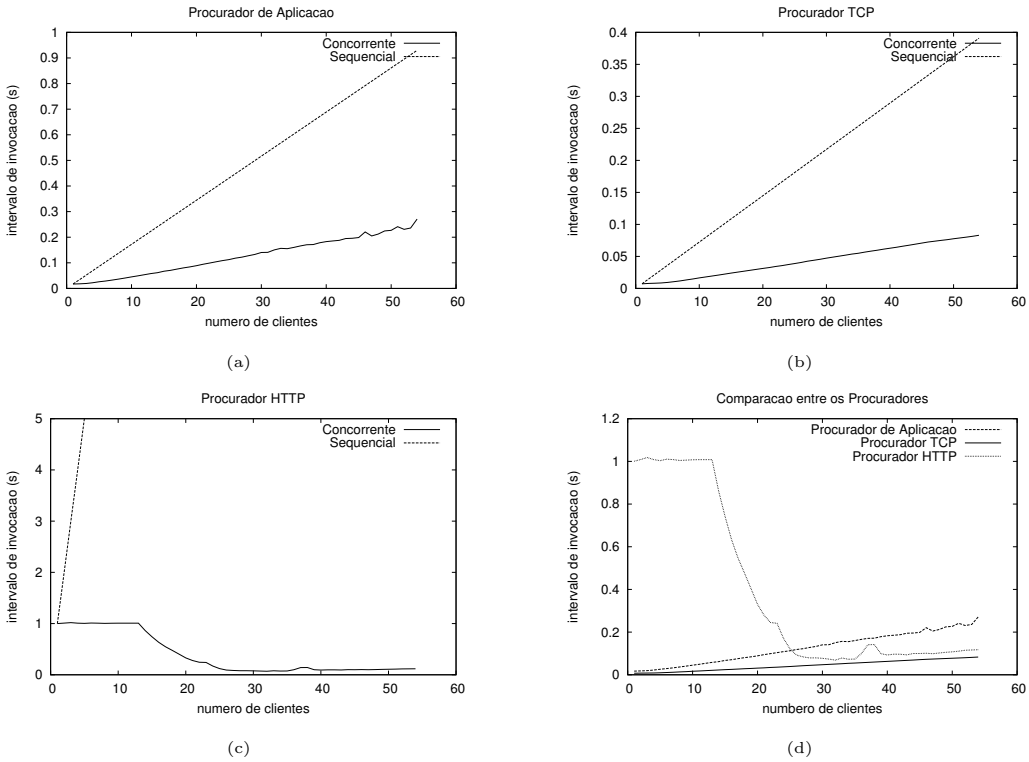

Figura 5.1: Testes de Escalabilidade

As figuras 5.1.a e 5.1.b apresentam o resultado dos testes realizados para a abordagem OMG e Procurador TCP respectivamente. Em ambos os casos é feita também uma comparação com o processamento sequencial (teórico) para identificar o ganho que foi obtido com a versão concorrente utilizando co-rotinas. As curvas do processamento sequencial foram obtidas utilizando o valor do atraso no atendimento de um único cliente e extrapolando estes valores de maneira linear para o atendimento de mais de um cliente. Como pode ser verificado, as versões concorrentes apresentaram um comportamento linear com coeficientes angulares bem inferiores àqueles das versões sequenciais (Abordagem OMG: redução do coeficiente angular de 0,017 para 0,005; Abordagem Procurador TCP: redução do coeficiente angular de 0,007 para 0,002). À despeito da limitação existente na abordagem Procurador TCP com relação ao número máximo de conexões permitidas simultaneamente (ver seção 3.2.3), os procuradores apresentaram um ótimo comportamento no que se refere à escalabilidade.

Um aspecto interessante pode ser identificado no gráfico da figura 5.1.c que apresenta os resultados para a abordagem Procurador HTTP com intervalo de consulta de 1 s. Neste gráfico da abscissa 1 até 13 clientes o comportamento é o esperado com o atraso sendo ligeiramente superior a 1 s visto ser este o intervalo de consulta HTTP. No entanto a partir do décimo quarto cliente, apesar do aumento no número de clientes atendidos, o atraso na invocação remota cai rapidamente. A explicação para este fato vem da característica de "transporte oportunista de mensagens" - TOM (ver seção 3.3.1) apresentada por esta abordagem. Como existem na rede protegida 13 objetos servidores, a partir do valor 14 os clientes adicionais são distribuídos para objetos CORBA que já estão atendendo a outros clientes. O resultado é que ao enviar uma resposta GIOP através de uma requisição HTTP, o objeto CORBA recebe no corpo do pacote HTTP de resposta as requisições GIOP enviadas por outros clientes que se encontravam armazenadas no procurador. Dessa forma, o objeto CORBA passa a enviar cada vez menos mensagens de consulta HTTP visto que as requisições chegam "de carona" (TOM) no envio de uma mensagem de resposta GIOP. O decaimento na curva se extende até um determindo ponto onde o excesso de clientes passa a sobrecarregar o procurador mais do que o ganho obtido com o recebimento de mensagens via TOM. Nos testes executados, o menor atraso obtido foi quando havia 32 clientes para 13 servidores sugerindo que o menor atraso obtido na abordagem Procurador HTTP ocorre quando cada servidor atende em média 2,5 clientes. Esse fato evidenciou que em um ambiente com vários clientes e objetos servidores,

um alto número de procuradores não significa necessariamente um melhor serviço. No entanto mais testes são necessários a fim de se obter um melhor entendimento no comportamento do procurador visto que a razão ótima encontrada deve depender de outras variáveis específicas do experimento como por exemplo o valor do atraso de invocação.

A figura 5.1.d apenas compara diretamente os resultados obtidos com as três abordagens (já mostradas nas figuras 5.1.a, 5.1.b e 5.1.c) em um mesmo gráfico. Além de se confirmar mais uma vez ser a abordagem Procurador TCP a mais rápida, evidenciou-se que a abordagem Procurador HTTP representa uma boa alternativa para situações de forte restrição (impossibilidade de configurar o firewall e de realizar uma conexão TCP para a rede externa), uma vez que a partir de determinados valores da razão clientes/servidor o aumento no atraso em relação a abordagem Procurador TCP é pequeno. Outro fato interessante apresentado neste gráfico é o menor atraso da abordagem Procurador HTTP em relação à abordagem OMG quando a razão clientes/servidor é maior do que 1,9.

5.6 Conclusão dos Testes

Os testes realizados demostraram que as 3 abordagens implementadas apresentaram o têm funcionamento esperado, estão em conformidade com o padrão CORBA e que a abordagem Procurador TCP e Procurador HTTP obtiveram os melhores desempenhos.