3
Propostas de Travessias de Firewalls/NAT

Este capítulo irá apresentar as propostas deste trabalho para que aplicações que utilizem CORBA como plataforma de comunicação possam atravessar firewalls/NAT.

Em um ambiente de rede protegido por firewalls/NAT, aplicações distribuídas podem encontrar diferentes situações no que diz respeito a permissões de acesso definidas pelo administrador da rede local. Geralmente essas situações vão desde a possibilidade de um elemento interno poder se conectar a algum elemento externo de maneira trivial até a impossibilidade de realizar qualquer conexão que não seja utilizando o protocolo de aplicação HTTP através de um procurador (*proxy*).

Da mesma forma, o acesso ao administrador da rede interna para solicitação de configurações nem sempre é uniforme. Existem situações que vão desde a possibilidade da realização de uma configuração mínima (abertura de uma única porta) até situações onde o acesso ao administrador é extremamente difícil e complexo.

Com o objetivo de atender da melhor forma a cada possível situação e explorar aspectos positivos de cada uma, são propostas por este trabalho três abordagens diferentes de travessia de firewalls/NAT. O desenvolvedor da aplicação CORBA é responsável por escolher a abordagem mais adequada e realizar a configuração da mesma. Essa escolha/configuração é feita através de um arquivo XML que será detalhado na seção 4.1 e todo o trabalho do desenvolvedor necessário para habilitar a aplicação a realizar a travessia é basicamente escrever este arquivo.

Este capítulo irá detalhar cada uma das três propostas nas seções seguintes, deixando aspectos relevantes de implementação para serem descritos no capítulo 4.

3.1
Abordagem OMG

A primeira proposta se baseia na especificação da OMG para travessia de firewalls/NAT [5] descrita na seção 2.1. Esta solução atende tanto a situações onde o objeto servidor está impossibilitado de receber conexões provenientes de fora da rede interna como situações nas quais o cliente não pode iniciar conexões para a rede externa. Ela exige a abertura de uma única porta no firewall da rede interna.

A solução apresentada é composta de duas partes. A primeira é habilitar tanto o ORB cliente quanto o ORB servidor a trabalharem com as estruturas e protocolos definidos pela OMG na especificação citada anteriormente. A segunda é a construção de um procurador de aplicação capaz de negociar sessões e repassar as mensagens GIOP entre os ORBs cliente e servidor. A arquitetura da solução está exposta na figura 3.1

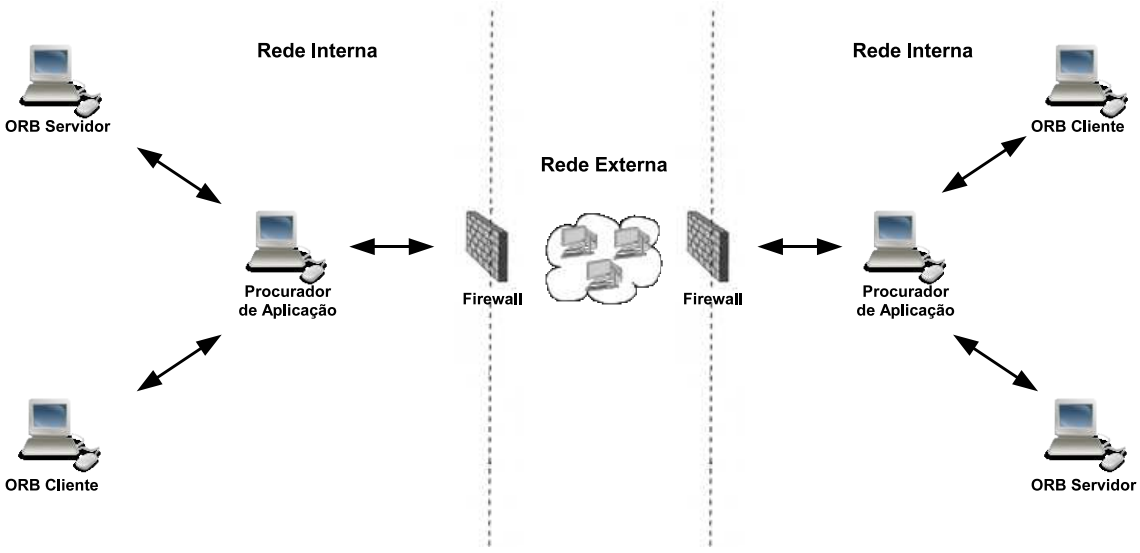

Figura 3.1: Abordagem OMG

De acordo com a figura 3.1, a idéia básica é que todo o ORB (cliente ou servidor) que desejar transpor firewalls/NAT deve instanciar um procurador de aplicação capaz de processar o protocolo GIOP/IIOP dentro da rede interna. A existência desse procurador permite que seja aberta uma única porta no firewall por onde todo o tráfego GIOP/IIOP irá passar para dentro e fora da rede. Todas as requisições provenientes de elementos externos e destinadas a objetos CORBA situados na rede interna passam primeiro pelo procurador de aplicação dessa rede que, por saber tratar o protocolo GIOP/IIOP, é

responsável por distribuir as requisições para os objetos correspondentes. Da mesma forma para clientes que não possam iniciar conexões com elementos fora da rede interna, é o procurador de aplicação que inicia essa conexão, transpondo o firewall/NAT pela porta aberta para ele e enviando a requisição para o objeto correspondente.

Tanto o ORB cliente quanto o ORB servidor devem ser informados/configurados pela sua respectiva aplicação sobre eventuais procuradores de aplicação existentes entre eles e a rede externa (e.g. Internet). Essa configuração é feita através do arquivo de configuração descrito na seção 4.1.

Conforme visto na seção 2.1, o IOR gerado pelo ORB do objeto servidor deve conter estruturas que indiquem todo o caminho (firewalls e procuradores de aplicação) entre a rede externa e o objeto em questão. De posse desse IOR, o ORB cliente deve enviar uma mensagem de negociação de sessão ao primeiro elemento (firewall) entre ele e o objeto servidor, contendo nesta mensagem toda a lista de elementos entre os dois ORBs. Essa mensagem é propagada e, em caso de sucesso, uma mensagem de resposta indicando que a sessão foi aberta é retornada ao ORB cliente. A partir de então mensagens GIOP são enviadas por esta conexão normalmente.

Esta solução tem a vantagem de implementar a especificação definida pela OMG e assim permitir a interoperabilidade com ORBs de outros fabricantes que também a implementem. O ponto negativo dessa abordagem é que objetos servidores CORBA que usem essa solução exigem inteligência dos ORBs clientes, visto que estes últimos devem também dar suporte à especificação (por inteligência denota-se a capacidade do ORB de tratar as estruturas definidas pela especificação de travessia de firewalls/NAT da OMG). As outras propostas mostradas nas seções 3.2 e 3.3 não requerem modificações no ORB cliente.

3.2
Abordagem Procurador TCP

Esta abordagem soluciona o problema de um objeto servidor estar impossibilitado de receber conexões provenientes da rede externa. Ela é indicada para cenários onde existe a possibilidade de uma aplicação, executando em uma máquina interna à rede protegida, iniciar uma conexão TCP com um elemento externo à rede e manter essa conexão aberta por tempo indefinido.

A idéia básica dessa abordagem é que o ORB da aplicação CORBA

servidora abra uma conexão com um procurador situado na rede externa, cujo endereço é especificado no arquivo de configuração (ver seção 4.1). O ORB mantém essa conexão aberta enquanto haver necessidade de objetos da rede interna serem acessados por elementos da rede externa e é por ela que serão encaminhadas tanto as requisições dos clientes externos como as respostas correspondentes.

A figura 3.2 mostra um cenário de uso desta abordagem. Nela dois ORBs servidores abrem conexões (indicadas pelo número 1) com um procurador por onde irão receber requisições provenientes da rede externa. Um cliente recebe um IOR de um objeto de um desses ORBs contendo o endereço do procurador no perfil IIOP e contacta o procurador (conexão 2) como se fosse ele o hospedeiro do objeto CORBA servidor. A requisição é repassada para o ORB do objeto e a resposta por sua vez é repassada para o ORB cliente.

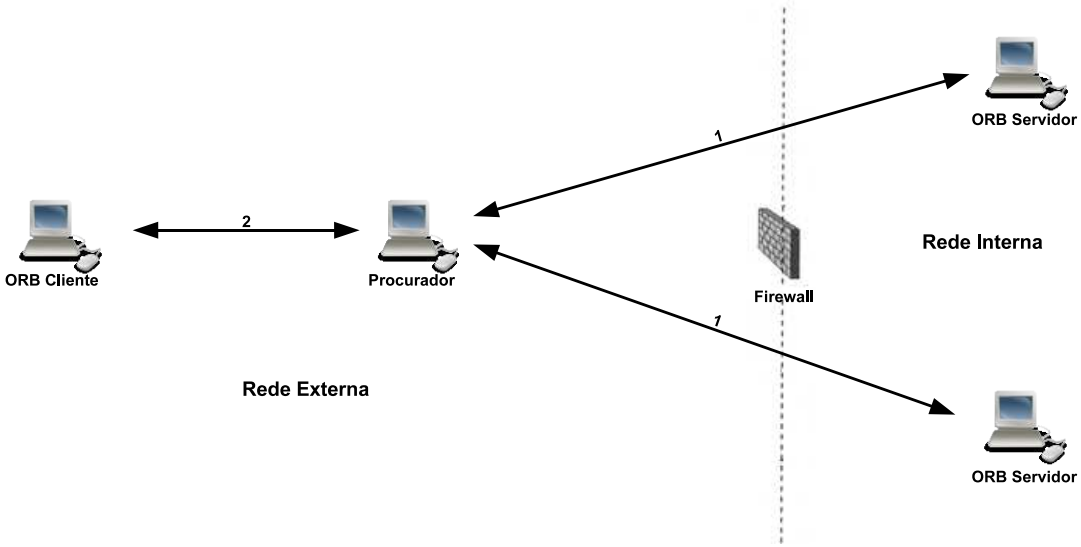

Figura 3.2: Abordagem Procurador TCP

Existem dois tipos de conexões feitas pelo ORB servidor ao procurador, conexões de registro e conexão de dados. O primeiro tipo diz respeito à conexões de registro de objetos CORBA servidores no procurador. Nesta conexão é enviada uma mensagem de registro e recebida uma resposta correspondente antes da conexão ser finalizada. Durante a existência do ORB vão existir várias dessas conexões, uma para cada registro de objeto servidor. O segundo tipo, a conexão de dados, diz respeito à conexão que deve ser iniciada pelo ORB servidor e por onde tanto as requisições de clientes externos chegarão, como as respostas serão enviadas. Esta conexão deve ser mantida aberta enquanto houver interesse do ORB servidor em receber requisições provenientes da rede externa. Durante a existência deste ORB deve haver apenas uma conexão desse

tipo, por onde chegarão as requisições de todos os objetos que foram registrados por ele no procurador.

A figura 3.3 mostra um exemplo de comunicação entre um ORB servidor e um procurador. Nela o ORB abre uma conexão de registro (indicada pelo número 1) e envia a mensagem correspondente. Após o recebimento da resposta ele fecha a conexão e abre uma conexão de dados (indicada pelo número 2) e a mantém aberta. Novos objetos serventes que porventura venham a ser criados implicarão em aberturas de novas conexões de registro (indicadas pelos números 3 e 4) e as requisições destinadas a eles serão recebidas pela conexão de dados aberta anteriormente. O motivo de se enviar as mensagens de registro em conexões separadas é simplificar o tratamento da conexão de dados: uma vez realizado um protocolo simples de abertura desta conexão, mensagens de requisição e resposta GIOP podem trafegar e esta conexão pode ser tratada da mesma forma que as outras conexões existentes com outros ORBs clientes.

Figura 3.3: Tipos de Conexões - Procurador TCP

Nesta solução, para o ORB servidor é o procurador quem faz as requisições e do mesmo modo para os clientes situados na rede externa é o procurador o hospedeiro do objeto CORBA servidor, ficando para ambos (ORBs cliente e servidor) transparente a existência de um elemento intermediário. A transparência só não é total porque o ORB servidor tem que inicialmente registrar os objetos no procurador e manter a conexão de dados aberta. No entanto, o processo de recebimento de requisições e envio de respostas funciona para este ORB como se fosse o procurador o responsável pelas requisições. No caso do ORB cliente a transparência é total e nenhuma modificação no mesmo precisa ser feita.

Nas próximas seções serão detalhadas as mensagens trocadas entre o ORB servidor e o procurador bem como os protocolos de registro de objetos e de tratamento de requisições.

3.2.1
Cabeçalho de Mensagens

Todas as mensagens enviadas pelo ORB servidor ao procurador e vice-versa, com exceção das mensagens GIOP, devem incluir o cabeçalho descrito abaixo em IDL (*Interface Definition Language* [1]):

```
module TCPProxy {
  struct HeaderMsg {
    boolean byte_order;
    octet message_type;
    long message_size;
  }
}
```

Assim como as mensagens GIOP, as mensagens trocadas entre o ORB servidor e o procurador são codificadoas em CDR (*Commom Data Representation* [1]). O campo `byte_order` tem a função de indicar a ordenação de bytes utilizada (*little* ou *big endian*) na codificação. O campo `message_type` e `message_size` indicam o tipo de mensagem trocada e o tamanho da mesma (excluindo este cabeçalho) respectivamente. Este cabeçalho é seguido pela mensagem correspondente codificada em CDR.

3.2.2
Protocolo de Registro de Objetos Servidores

A cada pedido de criação de um objeto CORBA servidor feito pela aplicação CORBA ao ORB, este último deve abrir uma conexão TCP com o procurador, enviar uma mensagem de registro, esperar a resposta e fechar a conexão. Depois desses passos o ORB pode retornar o controle à aplicação, uma vez que o objeto já está registrado no procurador e já pode ser alcançado por elementos situados na rede externa. A sintaxe da mensagem de registro, bem como o seu identificador (a ser incluído no cabeçalho), estão descritos abaixo em IDL:

```
module TCPProxy {
  const long REGISTER_MSG_ID = 0
```

```
    struct RegisterMsg {
      IOP.IOR ior;
      long orb_id;
    }
  }
```

Nesta mensagem o campo `ior` contém o IOR do objeto gerado pelo ORB. O procurador irá modificá-lo para que ele agora faça referência ao procurador. Essa modificação é feita nos campos `host`, `port` e `object_key` do primeiro perfil IIOP encontrado no IOR. Os campos `host` e `port` devem agora indicar um endereço do procurador enquanto que o campo `object_key` deve ser um identificador ainda não utilizado para o par `host/port` inserido. É de responsabilidade do procurador manter o mapeamento entre o IOR gerado por ele e o IOR original, repassando ao ORB servidor qualquer requisição recebida através do IOR gerado. É importante salientar que o procurador deve atualizar os campos `object_key` e `request_id` da mensagem GIOP `Request` (o motivo de se atualizar o `request_id` será detalhado na seção 3.2.3).

O campo `orb_id` é um identificador gerado na primeira mensagem de registro enviada pelo ORB. Este número identifica o ORB perante o procurador. Nesta primeira mensagem, o valor desse campo deve ser 0 para indicar que ainda não foi feito nenhum registro anterior. Neste caso, além de registrar o objeto, o procurador gera esse identificador (que não pode ser o valor 0 para não gerar ambiguidades) e o retorna na mensagem `RegisterReply` descrita em IDL a seguir:

```
module TCPProxy {
  const long REPLY_MSG_ID = 1
  struct RegisterReplyMsg {
    IOP.IOR ior_exported;
    long orb_id;
  }
}
```

A partir de então todas as mensagens seguintes de registro devem conter o identificador de ORB retornado para que o procurador saiba a que ORB associar os objetos representados por ele. A mensagem de resposta contém

também o campo `ior_exported` que contém o IOR gerado pelo procurador. A função desse campo é permitir que a aplicação CORBA possa ter acesso a este IOR e assim flexibilizar a publicação do mesmo. Do ponto de vista da aplicação servidora, logo após a criação do objeto CORBA servidor este IOR já deve estar disponível para a mesma, caso a operação tenha tido sucesso (na seção 4.3.1 é detalhado como essa disponibilização é feita).

3.2.3
Protocolo de Tratamento de Requisições

Após o primeiro registro de objeto, o ORB deve abrir com o procurador uma conexão de dados por onde serão encaminhadas tanto as requisições como as respostas GIOP. A mensagem de abertura de conexão de dados deve ser codificada em CDR e está descrita a seguir em IDL:

```
module TCPProxy {
  const long INIT_CHANNEL_MSG_ID = 2
  struct InitializeChannel {
    long orb_id;
  }
}
```

Como pode ser visto, o único campo da mensagem é o identificador do ORB recebido na resposta à mensagem de registro. Uma vez enviada essa mensagem, o procurador identifica essa conexão como sendo a conexão de dados por onde as requisições destinadas a objetos deste ORB serão repassadas. Após essa mensagem o ORB fica aguardando a chegada de requisições por esta conexão e todo o tráfego será apenas de mensagens GIOP.

Um aspecto importante a ser ressaltado é quanto ao identificador de requisições incluído nas mensagens GIOP de requisição e de resposta. A partir do envio da mensagem `InitializeChannel` pelo ORB servidor, do ponto de vista deste último todas as requisições recebidas por esta conexão são requisições feitas pelo procurador. Segundo a especificação CORBA [1] o campo `request_id` contido no cabeçalho da mensagem GIOP `Request` serve para associar as respostas com as requisições e é responsabilidade do ORB cliente (neste caso o procurador) gerar valores que não criem ambiguidades. Mais

especificamente, durante uma conexão não deve ser reutilizado um valor de identificador de requisição relativo a uma requisição anterior que não teve ainda a sua resposta obtida, por estar pendente ou porque foi cancelada e nenhuma resposta foi obtida. Isso significa que o procurador não pode confiar nos identificadores recebidos dos ORBs clientes, pois as requisições poderão ser encaminhadas pela mesma conexão e no ORB servidor isso pode gerar ambiguidades e resultar no repasse equivocado de respostas. Por exemplo: caso o procurador repasse para o mesmo ORB servidor duas requisições GIOP provenientes de clientes diferentes e com o mesmo identificador, ao receber as respostas do ORB servidor o procurador não saberá para qual ORB cliente enviá-las. Ainda há a possibilidade de o ORB servidor achar que a segunda requisição é um reenvio e ignorá-la. É necessário portanto, que o procurador gere novos identificadores para cada requisição recebida e guarde a associação entre esses identificadores e os originais enviados pelos ORBs clientes.

A conexão de dados é finalizada quando o procurador ou o ORB servidor enviarem uma mensagem GIOP `CloseConnection`. Neste momento o procurador deve liberar os recursos locais utilizados pelo ORB, como conexões de rede, lista de requisições pendentes e os identificadores de ORB e de objetos.

Um último aspecto a ser tratado com relação a esta abordagem é quanto à escalabilidade. A existência de uma conexão TCP aberta durante um longo tempo reduz sensivelmente a escalabilidade do procurador, uma vez que este recurso (número de conexões TCP abertas simultaneamente) é geralmente limitado pelo sistema operacional. Conforme será verificado na seção 5.5, a conexão de dados ao mesmo tempo que permite a travessia de firewalls e diminui o retardo, reduz a escalabilidade do procurador.

3.3
Abordagem Procurador HTTP

Esta abordagem é indicada para situações em que o objeto CORBA servidor se encontra em uma rede protegida por firewall/NAT com as seguintes restrições:

- não é permitido receber conexões provenientes da rede externa;
- não há a possibilidade de se configurar o firewall/NAT;
- a única forma de se conectar com algum elemento da rede externa é utilizando o protocolo HTTP.

A idéia básica dessa abordagem é utilizar um procurador na rede externa que represente os objetos CORBA protegidos e realizar toda a comunicação entre o procurador e o ORB servidor utilizando o protocolo HTTP[1]. Essa abordagem é similar à usada pela plataforma JXTA (ver seção 2.2) para travessia de firewalls/NAT. Inicialmente foi considerado utilizar esta plataforma como camada de comunicação dentro do ORB, no entanto, devido ao elevado número de falhas e excessivo atraso gerado no momento dos testes, foi decidido implementar uma abordagem similar sem a utilização de toda a infra-estrutura JXTA.

Na presente abordagem, ao se criar um objeto CORBA, o ORB servidor deve enviar uma mensagem de registro encapsulada em um pacote HTTP ao procurador. Esta mensagem contém o IOR do objeto gerado pelo ORB servidor que será modificado pelo procurador para a inserção de seu endereço/porta no perfil IIOP. Este IOR modificado é então devolvido ao ORB no pacote HTTP de resposta e pode então ser usado pela aplicação para publicação na rede externa. Feito isso, ao utilizar este novo IOR um ORB cliente irá proceder com o protocolo GIOP enviando ao procurador a mensagem de requisição como se este último fosse o criador do objeto CORBA. Esta requisição será armazenada no procurador até que o ORB servidor faça algum contato via uma requisição HTTP. Ao ser feito esse contato, as requisições GIOP armazenadas são enviadas ao ORB servidor no corpo do pacote HTTP de resposta. O ORB servidor as processa e envia as respostas GIOP em outra requisição HTTP ao procurador que irá reencaminhá-las aos respectivos ORBs clientes. Para o cliente, a transparência dessa abordagem é total visto que todo o processamento se dá como se fosse o procurador o responsável pelo objeto CORBA, pois o primeiro desconhece a existência do mecanismo para a travessia de firewalls/NAT.

A figura 3.4 exemplifica o processo. As setas 1 e 2 representam um registro de objeto no procurador. O cliente de posse do IOR gerado pelo procurador invoca um método remoto indicado pela seta 3. As setas 4 e 5 mostram o ORB servidor enviando uma consulta ao procurador e recebendo a requisição armazenada. Após tratar a requisição, o ORB envia a resposta GIOP encapsulada em uma mensagem HTTP ao procurador (seta 6 com a respectiva resposta HTTP representada pela seta 7) que por sua vez encaminha esta resposta ao ORB cliente (seta 8).

[1] O motivo do nome Procurador HTTP se deve ao protocolo de comunicação usado entre o ORB servidor e o procurador

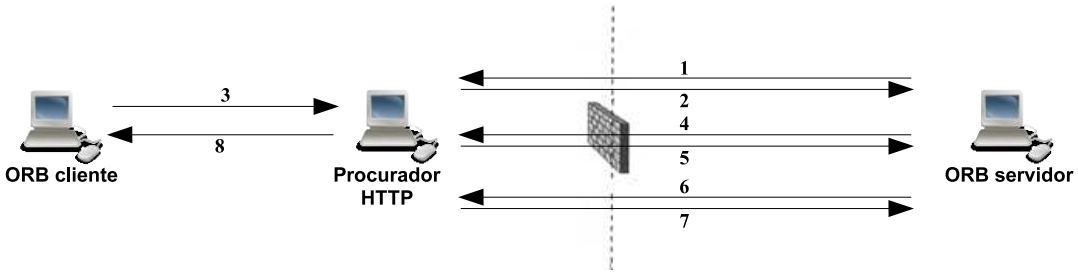

Figura 3.4: Procurador HTTP - Exemplo

3.3.1

Protocolo Procurador-ORB Servidor

Assim como na abordagem Procurador TCP, cada mensagem trocada entre o o ORB servidor e o procurador deve conter o seguinte cabeçalho definido em IDL:

```
module HTTPProxy {
  struct HeaderMsg {
    boolean byte_order;
    octet message_type;
    long message_size;
  }
}
```

Este cabeçalho deve estar codificado em CDR e os campos `byte_order`, `message_type` e `message_size` indicam respectivamente a ordenação de bytes utilizada, o tipo da mensagem e o seu tamanho.

Existem 4 tipos de mensagens diferentes enviadas pelo ORB servidor ao procurador. A seguir tem-se uma breve descrição de cada uma delas:

Register : Requisição de registro de objeto no procurador;

Remove : Remoção de objeto registrado no procurador;

Polling : Mensagem de consulta por novas requisições;

Reply : Envio de mensagem GIOP Reply para ser encaminhada ao ORB cliente.

Dois tipos de mensagens são enviadas pelo procurador ao ORB servidor que são:

RegisterReply : Resposta a uma requisição de registro de objeto no procurador;

Request : Envio de uma mensagem GIOP Request para ser tratada pelo ORB.

Todas essas mensagens são codificadas em CDR e devem estar contidas dentro do corpo de um pacote HTTP, visto ser este o único protocolo permitido pelo firewall. Das mensagens descritas acima, a única que merece uma explicação mais detalhada é a mensagem de `Polling`. Ela tem a função de garantir que dentro de um determinado intervalo de tempo (especificado no arquivo de configuração XML - seção 4.1) alguma mensagem HTTP será enviada ao procurador e assim eventuais requisições GIOP armazenadas poderão ser reencaminhadas no corpo da mensagem HTTP de resposta. Conforme será visto mais adiante as requisições GIOP podem chegar ao ORB servidor de outras formas além de uma resposta à mensagem de `Polling`, no entanto essa mensagem periódica não só garante o eventual recebimento das mensagens como limita o atraso na aplicação causado pela abordagem.

Assim como na abordagem Procurador TCP, as mensagens enviadas pelo ORB devem conter um identificador do mesmo perante o procurador. Este identificador é enviado ao ORB na resposta ao primeiro registro de objeto CORBA (nesta mensagem de registro o ORB envia como identificador o valor 0 para indicar a solicitação de geração de identificador).

A fim de se reduzir o número de mensagens HTTP trocadas e diminuir o atraso nas aplicações, dentro de um mesmo pacote HTTP pode haver uma combinação qualquer das mensagens descritas anteriormente (respeitando-se, é claro, a orientação de cada uma). Por exemplo o ORB servidor pode enviar de uma só vez um conjunto de mensagens de registro de objeto e de respostas GIOP em um mesmo pacote HTTP. O procurador por sua vez pode incluir mensagens de resposta de registro de objetos e eventuais requisiçoes armazenadas no pacote HTTP de resposta.

Um último aspecto a ser tratado nesta abordagem é com relação ao cabeçalho HTTP. O cabeçalho utilizado deve conter o seguinte formato:

```
POST <url> HTTP/1.0
```

```
Content-Length: <size>
```

A URL indicada por `<url>` pode ser qualquer valor, não sendo interpretada pelo procurador ou ORB. O campo `Content-Length` deve indicar no valor `<size>` o número de bytes do corpo da mensagem HTTP, que deve conter apenas as mensagens descritas anteriormente. Esse campo possibilita ao procurador/ORB obter precisamente as mensagens enviadas e proceder com a decodificação das mesmas. Qualquer campo adicional no cabeçalho HTTP será ignorado.

O apêndice B apresenta o formato das mensagens trocadas por esta abordagem além de oferecer maiores detalhes sobre as suas semânticas.

Quanto aos identificadores das requisições GIOP recebidos, o procurador não pode confiar nos valores enviados pelos ORBs clientes. Assim como na abordagem Procurador TCP, ele também deve gerar novos identificadores e guardar o mapeamento entre estes e os originais pelo mesmo motivo: o procurador não saberia para que ORB cliente encaminhar respostas GIOP recebidas com o mesmo identificador.

A abordagem Procurador HTTP tem a vantagem de não requerer nenhuma configuração no firewall e ainda assim ser viável com as condições mais restritivas até agora vistas. Em compensação, tanto o atraso imposto pela consulta HTTP (*polling*) como a sobrecarga gerada pelo encapsulamento das mensagens em pacotes HTTP são pontos negativos que não podem ser ignorados. Apesar destes aspectos negativos, será visto na seção 5.5 que a característica de "transporte oportunista de mensagens" - TOM[2] utilizado nesta abordagem apresentou resultados interessantes e animadores. Quanto à escalabilidade, esta abordagem é melhor do que a abordagem Procurador TCP, visto que aqui o ORB servidor não exige tantos recursos do procurador.

[2]Tentativa de tradução para o termo em inglês *piggyback* para indicar a característica desta abordagem de receber requisições GIOP na resposta a qualquer tipo de mensagem enviada.