EXECUTIVO - UM SUPORTE DE PROCESSAMENTO DISTRIBUIDO PARA GERENCIAMENTO DE REDES DE TELECOMUNICAçÕES

Joaquim Celestino Júnior
jc@masi.ibp.fr

Jean-Pierre Claudé
jpc@masi.ibp.fr

Universidade Pierre et Marie Curie
Laboratoire MASI
45, av. des Etats-Unis
78000-Versailles-France

Sumário

Este trabalho descreve as características e a implementação de um módulo de programa, que foi desenvolvido para dar suporte ao processamento distribuído em um ambiente de administração de redes de telecomunicação. Este módulo está sendo utilizado nos programas europeus RACE e ESPRIT.

I - Introdução

Gerenciar redes de telecomunicações é uma área de pesquisa que tem recebido grande atenção por parte de empresas e da comunidade científica.

Uma rede para gerenciamento de redes de telecomunicações é uma rede separada logicamente que interfaceia redes de telecomunicações em diversos pontos para enviar/receber informação para/da rede e controlá-la.

Esse trânsito de informação é feito através de uma plataforma computacional que abstrai dos usuários todos os problemas concernentes, tais como: distribuição, heterogeneidade, etc..

A proposição de uma plataforma geral para gerenciar redes de computadores tem sido definida por membros de vários projetos europeus[1,2]. Esta plataforma consiste em cinco camadas que abstraem a complexidade e a heterogeneidade dos sistemas de computadores e dos protocolos de comunicação presentes. Estas camadas também possibilitam a transparência de distribuição, de acesso à informação e de replicação. Elas estão divididas em: Núcleo da plataforma de computação, suporte de processamento distribuído, interface para a plataforma de computação, ambiente

de suporte para gerenciar redes de telecomunicações e aplicações de genéricas de gerenciamento.

Este trabalho tem como objetivo o de descrever a camada suporte de processamento distribuído que tem sido implementada, por este laboratório, dentro dos programas europeus RACE e ESPRIT nos dois últimos anos.

II - Suporte Para Processamento Distribuído

Quando se estuda a gerencia de redes de telecomunicações algumas características ou comportamentos da plataforma que são visíveis e importantes ao usuário, são chamadas de requerimentos não funcionais. Eles podem ser enumerados através dos aspectos de segurança, encapsulamento e transparência.

O aspecto de segurança é um problema complexo, pois está ligado a todos os componentes que fazem parte de uma plataforma para gerenciar redes de telecomunicações.

O encapsulamento está ligado ao fato de que deve-se permitir que as aplicações desenvolvidas sobre a plataforma possam co-existir ou mesmo trabalharem juntas com outras aplicações.

A distribuição impõe a necessidade de suporte para vários níveis de transparências que são enumeradas como:

- Localização: permite a interação de um componente de um programa sem o conhecimento de sua localização física;
- Acesso: permite idêntica invocação semântica para componentes locais e remotos;
- Migração: esconde o efeito de um componente do programa ter sido transferido de uma localização para outra;
- Falha: as aplicações devem conhecer ou serem notificadas de falhas nas interações, para que as ações necessárias possam ocorrer.

O suporte para processamento distribuído é o mecanismo que permite a comunicação entre todos os módulos que fazem parte de uma plataforma de computação. Este mecanismo foi desenvolvido no laboratório MASI, com o nome de EXECUTIVO.

III - O EXECUTIVO

O EXECUTIVO é o responsável pelo controle de comunicação e pela distribuição entre todos os componentes da rede. Ele fornece

um ponto de acesso a todos esses componentes, através de uma Interface Envia/Recebe [3].

De maneira a fornecer todos esses requisitos, o EXECUTIVO mantém uma interface com o sistema operacional local. Ele pode ser visto em um primeiro momento como dois blocos independentes, ou seja: a Interface Envia/Recebe e o EXECUTIVO propriamente dito.

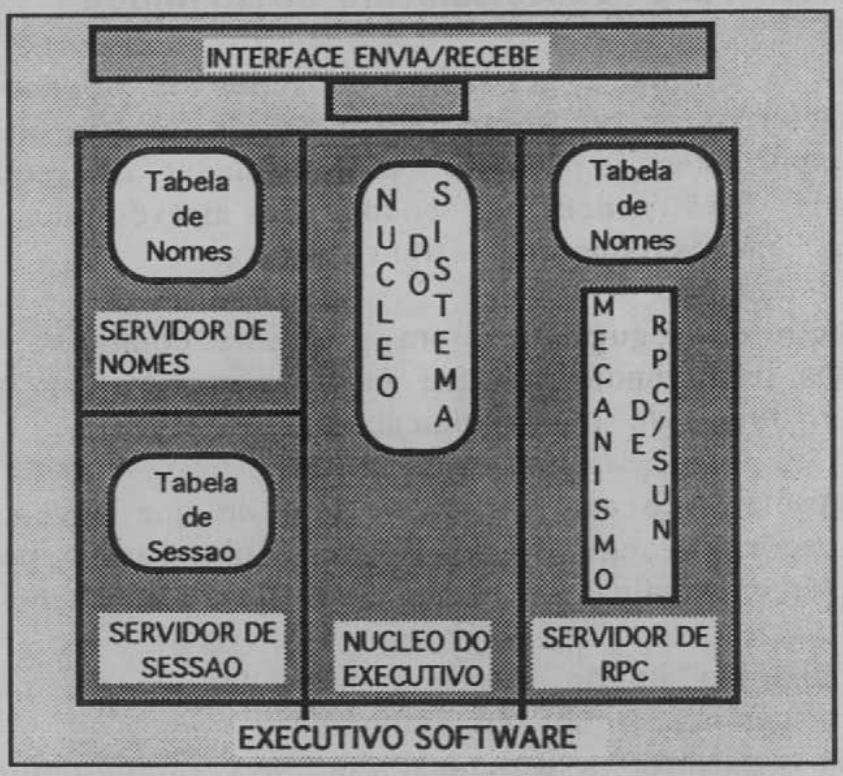

Figura 1: arquitetura lógica do EXECUTIVO

Os componentes têm suas funções descritas abaixo:

III.1 - Interface Envia/Recebe

Esta interface permite que o executivo dialogue com todos os componentes que fazem parte da plataforma de computação.

III.2 - Servidor de Nomes

Possui o endereço físico de todos os processos locais que manifestaram interesse em oferecer algum tipo de serviço à rede.

III.3 - Servidor de Sessão

Possui o estado atual em que o EXECUTIVO e as mensagens se encontram.

III.4 - Servidor de RPC

É a parte responsável por permitir a comunicação entre aplicações situadas em diferentes máquinas.

III.5 - Núcleo

É o gerente de todo o EXECUTIVO, seu principal mecanismo. Sua tarefa é gerir todos o blocos internos e dialogar diretamente com o núcleo do sistema operacional. Ao núcleo atribue-se as responsabilidades de temporização, escalonamento de prioridades, sincronização, etc.

IV - Uma Implementação do EXECUTIVO

A partir da especificação lógica do EXECUTIVO, define-se a comunicação utilizada entre todos os seus blocos internos, e entre este e as aplicações que ele suporta (figura 2).

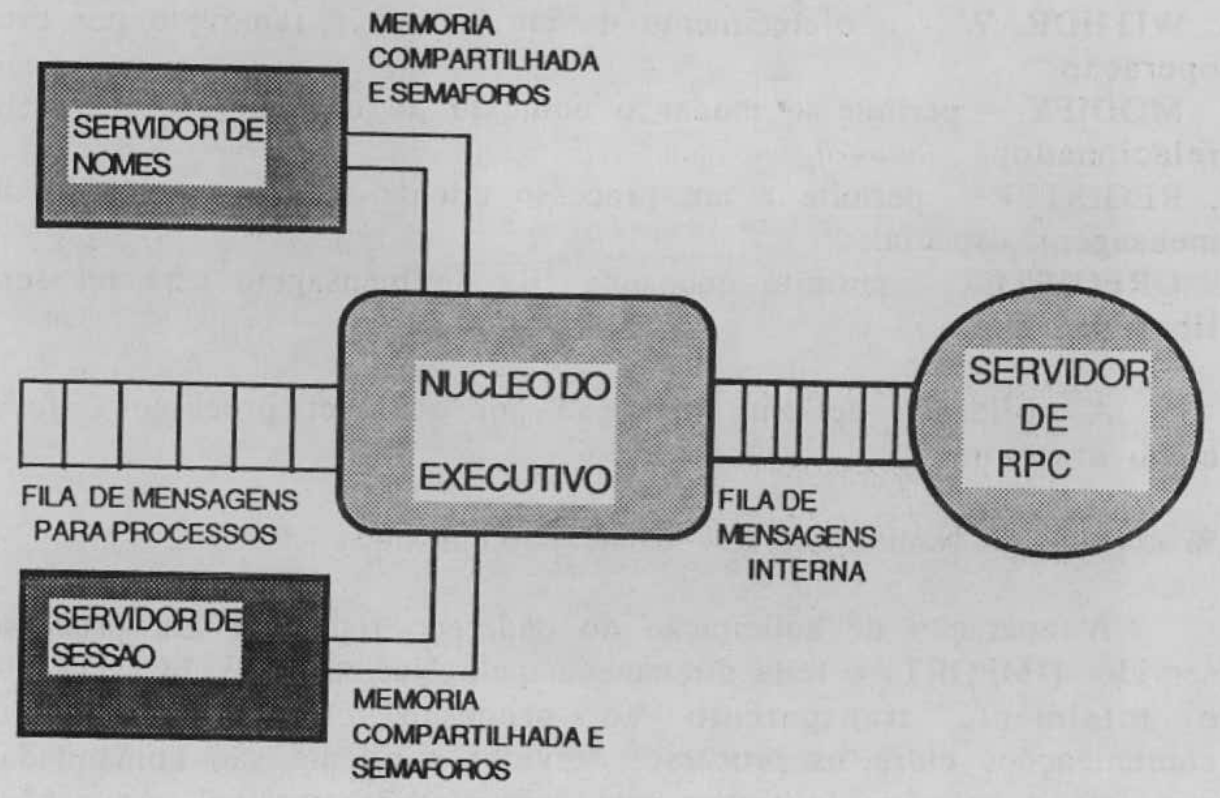

Figura 2: Aspectos da comunicação interna do EXECUTIVO

Uma primeira abordagem para implementar o EXECUTIVO foi fornecer ao mundo exterior um ponto de acesso onde todos os componentes do sistema, a partir de agora chamados processos, pudessem se conectar.

Sendo assim, decidiu-se que seria fornecido quando de sua inicialização, uma fila de mensagens que permitiria o diálogo com todos os processos concernentes ao gerenciamento.

Uma segunda abordagem foi baseada no modelo cliente/servidor. Um processo servidor poderia oferecer seus serviços à rede enquanto que um processo cliente poderia desejar utilizar os serviços disponíveis. Um processo poderia ainda, em um dado instante, ser cliente e servidor.

Era preciso estabelecer regras para que os processos pudessem efetivamente cooperar. Essa cooperação exigia um nível de detalhes no uso de comunicação interprocessos que deveriam ser abstraídas do usuário. Decidiu-se então pela criação de uma pequena gramática baseada em IDL[4]. As seguintes operações foram definidas, com vistas a uniformizar a função desempenhada por um processo:

. EXPORT - um processo oferece seus serviços(servidor) e os parâmetros permitidos são: o nome da entidade e o nome do contexto;

. WITHDRAW - o oferecimento de um serviço é removido por esta operação;

. MODIFY - permite-se mudar o contexto ao qual um serviço está relacionado;

. REGISTER - permite a um processo cliente solicitar uma fila de mensagem especial;

. UREGISTER - permite que uma fila de mensagem especial seja liberada.

A utilização de uma operação por qualquer processo é feita como a seguir:

% <operação> Name=<server> Context=<contexto>

A operação de solicitação do endereço físico de um processo servidor (IMPORT) é feita diretamente pelo Núcleo do EXECUTIVO e é totalmente transparente ao processo cliente. Todas as comunicações entre os processos servidor e cliente são comandadas por este e cabe a ele tornar esta comunicação possível. As seções IV.1 e IV.4 darão detalhes sobre assunto.

IV.1 - Interface Envia/Recebe

Ao usuário do sistema são oferecidos alguns níveis de transparências conforme explicado na sessão II. A Interface E/R encarrega-se de fornecer a transparência de acesso ao sistema. Para

fornecer esta transparência são necessários dois passos importantes: a definição dos modos de comunicação que seriam utilizados pelo usuário e o fornecimento de uma linguagem comum para o envio de qualquer mensagem em qualquer ambiente.

Quatro modos de comunicação(primitivas) foram definidos para uso pelo usuário:
-Síncrona bloqueante - CALL: é ligada a mensagens onde uma resposta é esperada e neste caso, somente a interface ligada ao processo continua bloqueada. Uma primitiva do tipo REPLY será responsável por conduzir a resposta.
-Síncrona Diferenciada - REQUEST: aqui uma resposta é esperada mas o processo não permanece bloqueado. O processo decide, depois de um certo tempo, coletar a resposta através de um outro tipo de primitiva (COLLECT).
-Assíncrona com confirmação - CCAST: é ligada a mensagens em que a confirmação de recebimento pelo processo servidor é necessária. A primitiva REPLY é usada como carona;
-Assíncrona sem confirmação - CAST: é ligada a mensagens que não necessitam de nenhum tipo de retorno.

A especificação desta interface foi feita em "C", como segue:

```
typedef enum i_session {CALL, CAST, CCAST, REQUEST, REPLY, COLLECT, REJECT} I_Session
typedef enum i_operation {EXPORT, WITHDRAW, MODIFY, REGISTER, UREGISTER, ACCEPT, UACCEPT} I_Operation

typedef struct i_pdu
                {
                union
                    {
                I_Session session;
                I_Operation operation;
                } primitive;
                char *Name;
                char *Context;
                char *Message;
                } I_Pdu;
```

Figura 3: estrutura oferecida pela interface ao processo cliente

Assim, a estrutura de mensagem que um processo cliente deve enviar para um processo servidor tem o seguinte formato:

% <sessão> Name=<server> Context="<contexto> Message= <mensagem CMIS>

A estrutura acima é chamada de mensagem do EXECUTIVO e possui um campo chamado "Message", que é a mensagem do usuário que será processada pelo processo servidor. Esta mensagem é do tipo CMIS[5]. Durante todo este trabalho o termo mensagem, salvo contrário, fará referência à mensagem do EXECUTIVO.

É neste nível que a transparência de acesso se faz presente, pois uma mensagem padrão do tipo CMIS pode ser usada para todas as redes e não requer do usuário o conhecimento de cada ambiente. Por exemplo, um usuário pode se servir da mesma mensagem para gerenciar redes SNMP, SNA, etc...

Um pré-processador analisa o programa fornecido pelo usuário e transforma em código "C" o comando após o caracter "%". Uma estrutura para comunicação interprocessos é montada a partir desta linha de comando e é totalmente transparente ao usuário.

Não existe um limite máximo para o tamanho da mensagem. Em nossa implementação o tamanho máximo permitido é 2K bytes. Quando ela ultrapassa esse tamanho ela é automaticamente fragmentada. Esclareça-se que este tamanho pode ser renegociado com o núcleo do sistema operacional.

Alguns processos podem requerer o uso de uma fila particular e são ditos processos especiais. O número de mensagens que são permitidas em cada fila está diretamente ligado ao sistema operacional em curso. No caso de filas especiais é prevista a utilização de níveis de prioridades para estes tipos de processos.

Mais detalhes sobre o gerenciamento da comunicação será visto na seção IV.4 intitulada o Núcleo do EXECUTIVO.

IV.2 - O Servidor de Nomes

Um dos objetivos de um suporte de processamento distribuído é fornecer ao usuário o que chamamos de transparência de localização. Isto consiste em abstrair do usuário a necessidade de ter que localizar o endereço físico de um componente que ele tenha necessidade de usar.

O Servidor de Nomes é o responsável por guardar os endereços físicos de todos os processos servidores que estão presentes numa determinada máquina e que são chamados de processos servidores locais. O Núcleo faz referência ao Servidor de Nomes sempre que ele precisa encontrar o endereço físico, armazenar , suprimir ou modificar um processo servidor local.

Os processos servidores são armazenados como uma árvore de dois níveis, um para o nome do servidor e outro para o processo. A figura 4 mostra esta árvore. Abaixo da raiz, dois níveis são possíveis. No primeiro nível encontramos os nós relacionados aos nomes com o quais cada processo servidor é exportado. No segundo nível encontramos o contexto relacionado a cada servidor.

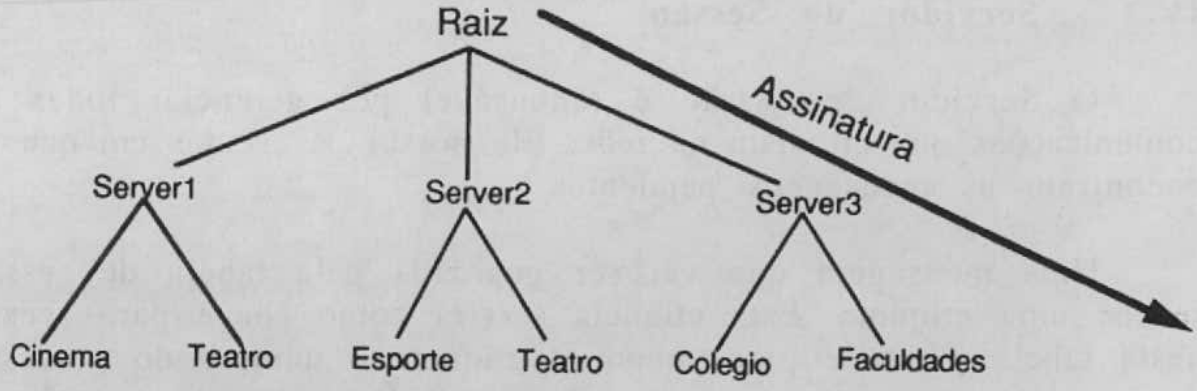

Figura 4: árvore de registro do Servidor de Nomes

O deslocamento na árvore, da raiz até as folhas, serve para encontrar um processo servidor único. É possível se ter um mesmo contexto para nomes diferentes. Sendo assim, denominou-se a assinatura de um processo a associação entre o nome e o contexto.

A solicitação de um processo cliente pode especificar que o contexto não é importante, deste modo mais de uma assinatura pode satisfazer as exigências do processo. Neste caso, o Servidor de Nomes poderá encontrar vários endereços. A escolha de uma assinatura é feita aleatoriamente pelo Servidor de Nomes.

É possível que processos em uma outra máquina possuam o mesmo nome e o mesmo contexto. Neste caso a identificação desse processo que possui também a mesma assinatura, não será possível. O processo local que possui a mesma assinatura será o candidato para estabelecer a comunicação.

Esta abordagem está conforme as especificações definidas no ODP Trader[6] e X500 Diretório[7].

O Servidor de Nomes é implementado como um programa para gerenciar tabelas, usando semáforos e memória compartilhada. É um programa fácil de gerenciar e, devido ao uso de semáforos, assegura uma consistência dos dados armazenados.

A tabela permite que operações de escrita (write), remoção (delete) e modificação (modify) sejam efetuadas. Essas operações estão diretamente relacionadas com as operações EXPORT, WITHDRAW e MODIFY. Outras operações foram implementadas para permitir um monitoramento do EXECUTIVO, como por exemplo:
.Search - pesquisa diretamente uma assinatura;
.List - permite listar todos os contextos de um determinado nome.

IV.3 - Servidor de Sessão

O Servidor de Sessão é reponsável por gerenciar todas as comunicações que circulam na rede. Ele possui os estados em que se encontram as mensagens pendentes.

Uma mensagem que vai ser guardada pela tabela de sessão recebe uma etiqueta. Esta etiqueta servirá como chave para acesso nesta tabela. Usou-se para compor a etiqueta o número do processo que envia a mensagem, seguido do nome da máquina na qual ele for criado. Isto garante que esta chave seja a única.

A mensagem passa a contar com os seguintes campos para transito interno:

```
typedef struct i_stream
                    {
                    char *cle;
                    char *hote;
                    I_Session mode;
                    int address;
                    Message stream;
                    } I_Stream;
```

Figura 5 - Estrutura interna da tabela de sessão

Cabe ao Servidor de Sessão, em comum acordo com o Núcleo, quando este recebe uma mensagem estar atento a fim de saber para quem a resposta deverá ser enviada, se a mensagem foi perdida, ou se o tempo foi expirado (time-out).

O Servidor de Sessão junto com o Núcleo tentam esconder do usuário as possíveis falhas internas. Se em um dado momento o Servidor de Sessão descobre que uma resposta não foi enviada ao destinatário, ele ativa um algorítimo de "back-off"[8].

A implementação consiste em oferecer a cada mensagem uma certa fatia de tempo. Se essa fatia mostrar-se insuficiente, o seu valor é duplicado. Esta operação é repetida, em nosso caso, no

máximo 5 vezes, elevando-se o tempo para 32 vezes do valor original.

Para prover este tipo de serviço é importante que o Servidor de Sessão possa resolver os conflitos de comunicação no meio. Assim, como um exemplo, o tempo de espera à uma mensagem pode expirar e desta maneira o Servidor de Sessão é obrigado a enviar novamente a mensagem. Acontece, que a resposta pode chegar justamente após o "re-envio" dessa mensagem e sendo assim a próxima resposta não terá mais sentido. A implementação consistiu em descartar literalmente a resposta e desativar o temporizador com a remoção da mensagem da tabela de sessão.

IV.4 - Servidor de RPC

Um processo que seja de interesse de um usuário pode encontrar-se residindo na mesma máquina, ao qual chamamos de processo local ou pode residir em outra máquina, neste caso ele é chamado de processo distante ou remoto.

Se um processo é local, ele é resolvido no nível do Servidor de Nomes. A comunicação entre processos distantes é resolvida à nível do Servidor de RPC, utilizando os mecanismos de RPC oferecidos pelo sistema. O Servidor de RPC possui também uma tabela interna de nomes, que guarda o endereço de processos distantes que são adquiridos de acordo com a demanda do usuário.

A obtenção dos endereços de processos distantes é feita através de um algorítmo que interroga as máquinas que compõem a rede, quando um processo cliente solicita a comunicação com um processo servidor que não se encontra na tabela de nomes local.

As técnicas descritas abaixo baseia-se na liteturatura dos serviços de diretório[7]. Essas técnicas levam em conta a interrogação das máquinas, ou seja, Servidores de RPC, a partir de algoritmos precisos.

A primeira técnica corresponde a uma máquina interrogar cada máquina que compõe a rede, para poder estabelecer uma conexão com a máquina que possui o processo servidor procurado. A máquina que o possui responde, estabelece a comunicação e a pesquisa termina. Essa técnica pode não ser muito eficaz, quando a rede possui muitas máquinas e o tempo de resposta é um fator importante.

A segunda técnica consiste em registrar na tabela do Servidor de RPC os nomes do processos distantes que forem encontrados quando da interrogação. Nesta técnica, estes processos são considerados como candidatos em potencial a serem usados novamente. É, também, dada a possibilidade de uma máquina que não possui o processo servidor em sua máquina, mas a tenha já adquirido através da rede (por uma interrogação anterior) a responder a interrogação de uma terceira máquina. O benefício com esta técnica seria uma melhor performance no que diz respeito ao tempo dispensado para interrogar as outras máquinas, pois a tabela oferecida faria o papel de uma memória "cache". Esta é a técnica que está sendo utilizada no EXECUTIVO.

Na terceira técnica, conhecida como "broadcast", a máquina que necessita encontrar um processo servidor distante, difunde em toda a rede a sua solicitação e a máquina distante, que possui o processo servidor correspondente a assinatura desejada, responde ao solicitante.

Uma questão que foi resolvida neste nível diz respeito as assinaturas iguais. É o caso onde dois processos servidores de nomes e contextos idênticos têm sidos oferecidos. Nas duas primeiras técnicas, a primeira máquina que responde a interrogação será aceita. Na técnica Broadcast torna-se um pouco díficil de administrar pois várias possibilidades de conexão podem ser oferecidas. Assim, a escolha é feita aleatoriamente.

A escolha de uma dessas técnicas está ligada a vários fatores, dentre eles: o tamanho da rede, o tempo de resposta de cada máquina, etc. Na técnica Brodcast, a vantagem está em obter-se um tempo de resposta médio menor e uniforme que o obtido pelas outras duas técnicas. Uma possibilidade de utilização desta técnica seria quando as máquinas que fazem parte da rede de gerenciamento levarem um tempo muito grande para responder as interrogações.

Uma questão que foi abordada devido a técnica de utilisar-se uma memória "cache" diz respeito à consistência da mesma (tabela interna de nomes do RPC). Se o processo servidor foi desativado ou migrou para uma outra máquina é importante que a tabela de nomes seja atualizada. Neste caso duas situações podem ocorrer:
1) A própria tabela do Servidor de RPC local pode estar desatualizada. Na implementação do EXECUTIVO a tabela só será atualizada quando um processo cliente tentar manter comunicação diretamente com a máquina distante, depois de tê-la encontrado em sua tabela.

2) O servidor de RPC local pode interrogar outro servidor de RPC que possua o endereço do processo servidor que resida em outra máquina, mas que pode estar também desatualizado.

Nos dois casos o Servidor de RPC local tentará abrir uma conexão com o Servidor de RPC distante que, neste caso, deveria possuir o processo servidor em questão. A conexão será rejeitada, pois devido a consistência do Servidor de Nomes da máquina distante, ele responderá ao requisitante, dizendo que não possui o processo servidor em questão. Sendo assim, o servidor de RPC será obrigado a reativar o algoritmo de RPC para encontrar o processo servidor em outra máquina. No primeiro caso ele atualizará sua tabela de nomes.

Com relação as mensagens recebidas pelo Servidor de RPC dos usuários locais, a sua transferência é feita usando uma fila de mensagens internas e única, conforme mostrado na figura 2, e explicado na seção IV.4.

A parte de comunicação entre máquinas é feita utilizando o mecanismo de RPC fornecido pela SUN rpcgen [9]. Tendo em vista que o modelo adotado por este mecanismo está diretamente relacionado ao modelo Cliente/Servidor, e que, no caso da comunicação presente no EXECUTIVO entre dois servidores de RPC, este modelo não se aplica, algumas modificações foram implementadas no código gerado.

As máquinas que fazem parte da rede são de total conhecimento do Servidor de RPC. Cada cópia do EXECUTIVO disponível em cada máquina possui junto ao Servidor de RPC o conhecimento de toda a rede.

IV. 5 - Núcleo do EXECUTIVO

Este programa é o coração do EXECUTIVO, pois é ele que mantém o controle sobre todos os blocos internos.

O Núcleo é responsável por gerenciar o EXECUTIVO como um todo. Ele é encarregado de lançar todos os processos concernentes a cada módulo interno. Cabe a ele gerenciar todas as filas que se encontram ligadas ao EXECUTIVO.

O EXECUTIVO como visto acima, possui ligado a ele um grande número de filas. Três tipos de filas são considerados:

a) a fila multiplexada, conhecida de todas as aplicações componentes do sistema;
b) a fila existente entre este e o processo encarregado de gerir as mensagens de RPC;
c) as filas ligadas aos processos ditos especiais e aos processos servidores.

O Núcleo é ainda responsável pelo diálogo entre o EXECUTIVO e qualquer aplicação. Esse diálogo se faz utilizando a interface que se encontra ligada ao EXECUTIVO. Esta interface pode ser considerada como um subconjunto da interface E/R. Os serviços oferecidos nesta interface são simplesmente o envio e recebimento de mensagens, através das primitivas de comunicação do sistema.

IV.5.1 - Comunicação entre Núcleo-Fila Multiplexada

Esta é a única fila disponível para todos os processos e existente quando o EXECUTIVO é inicializado. Esta fila deve ser usada no mínimo uma vez pelos processos que tenham interesse em estabelecer uma comunicação no ambiente de redes.

O registro de um processo servidor, a comunicação entre o processo servidor e cliente, ou ainda, a criação de uma fila para um processo especial devem ser feitas através desta fila, usando as primitivas oferecidas pela interface E/R.

Nesta fila todos os processos têm a mesma prioridade e são administrados pelo Núcleo através de um processo dedicado a eles.

O exemplo abaixo tenta esclarecer como um processo envia e recebe uma mensagem do EXECUTIVO, através desta fila.

Ex: Suponhamos que o processo identificador (pid) deseja enviar uma mensagem, que necessita de uma resposta, para um processo servidor. Deste modo, a mensagem será enviada à fila multiplexada conhecida de todas as aplicações.

A interface E/R se encarrega de construir a mensagem em termos de primitivas de comunicação e de enviá-la através desta fila. A interface utiliza o campo chamado *msgtype* para selecionar o receptor da mensagem, no caso o EXECUTIVO, permitindo assim a multiplexagem [10]. Usou-se, ainda, um campo para guardar o *pid*, que será necessário para a recepção da mensagem de retorno.

O EXECUTIVO recebe uma mensagem através da primitiva *msgrcv*, prevendo na recepção um campo idêntico ao número desta fila.

O EXECUTIVO utilisando o pid, que foi enviado pela interface E/R, na primeira mensagem, constroi a mensagem de resposta colocando o pid no campo *msgtype*. O usuário para receber a resposta enviada pelo EXECUTIVO, mantém-se, através da interface, em estado de espera usando como tipo o pid na primitiva de recebimento *msgrcv*.

IV.5.2 - Comunicação entre o Núcleo e demais Filas

Como vimos antes, estas filas são únicas e utilizadas somente pelo EXECUTIVO e por cada processo proprietário em separado. A implementação utilizada é a mesma para todos os processos.

Uma diferença existente entre estes tipos de filas está no fato de que, no caso dos processos especiais, níveis de prioridade são colocados à disposição dos mesmos, enquanto que para os processos servidores esta prioridade não existe.

Os processos servidores servem aos processos clientes de acordo com as mensagens que chegam. A prioridade oferecida aos processos especiais está no fato de poder interromper outro processo de menor prioridade e enviar sua mensagem.

Existem processos dedicados pelo Núcleo para tratar exclusivamente dos processos especiais.

Quanto a comunicação entre o Núcleo e o Servidor de RPC, ela é interna e procede como explicado acima. Um processo é alocado para tratar desta comunicação.

A criação destes processos, através do Núcleo, para tratamento de filas, proporciona o mesmo grau de prioridade para cada tarefa, embora que dentro de cada tarefa a possibilidade do uso de prioridade possa existir.

IV.5.3 - Comunicação Interna

É importante nesta fase um entendimento de tudo que se passa dentro deste bloco, a partir do momento em que o EXECUTIVO recebe uma mensagem proveniente de qualquer das filas.

O EXECUTIVO ao ser carregado no sistema, mantém-se em estado de espera, através de todos os processos que se encontram rodando em paralelo.

Em um primeiro momento todas as tabelas encontram-se vazias. O Servidor de Nomes começa a preencher sua tabela a partir do momento em que processos servidores são declarados. Isto é feito quando o Núcleo, ao decodificar uma mensagem, descobre que se trata de uma mensagem do tipo EXPORT.

O Servidor de Sessão começa a preencher sua tabela quando o Núcleo, ao decodificar uma mensagem, descobre que uma mensagem deve ser retornada ao usuário.

O Núcleo ao mesmo tempo estaria gerando as prioridades fornecidas aos processos especiais, lendo as filas dos processos servidores para saber se existe alguma resposta a uma mensagem pendente, ou ainda, recebendo alguma mensagem que tenha sido colocada na fila destinada a comunicação RPC.

Na realidade, o Núcleo é uma máquina de estados que possui processos paralelos que estão sendo executados.

V - Um Exemplo Completo

Este exemplo, explicitado na figura 5, mostra como uma mensagem enviada por um processo cliente e sua respectiva resposta flui dentro do EXECUTIVO. Tentou-se mostrar um caso complexo, onde um processo cliente envia uma mensagem do tipo CALL, para um processo distante. A resposta é enviada através da primitiva REPLY.

Os seguintes passos explicam como é feita a comunicação entre os dois processos:

1 - a Interface E/R deposita a mensagem na fila;

2 - o Núcleo recebe e repassa ao Servidor de Nomes para saber se o processo servidor é local, obtendo uma negação como resposta;

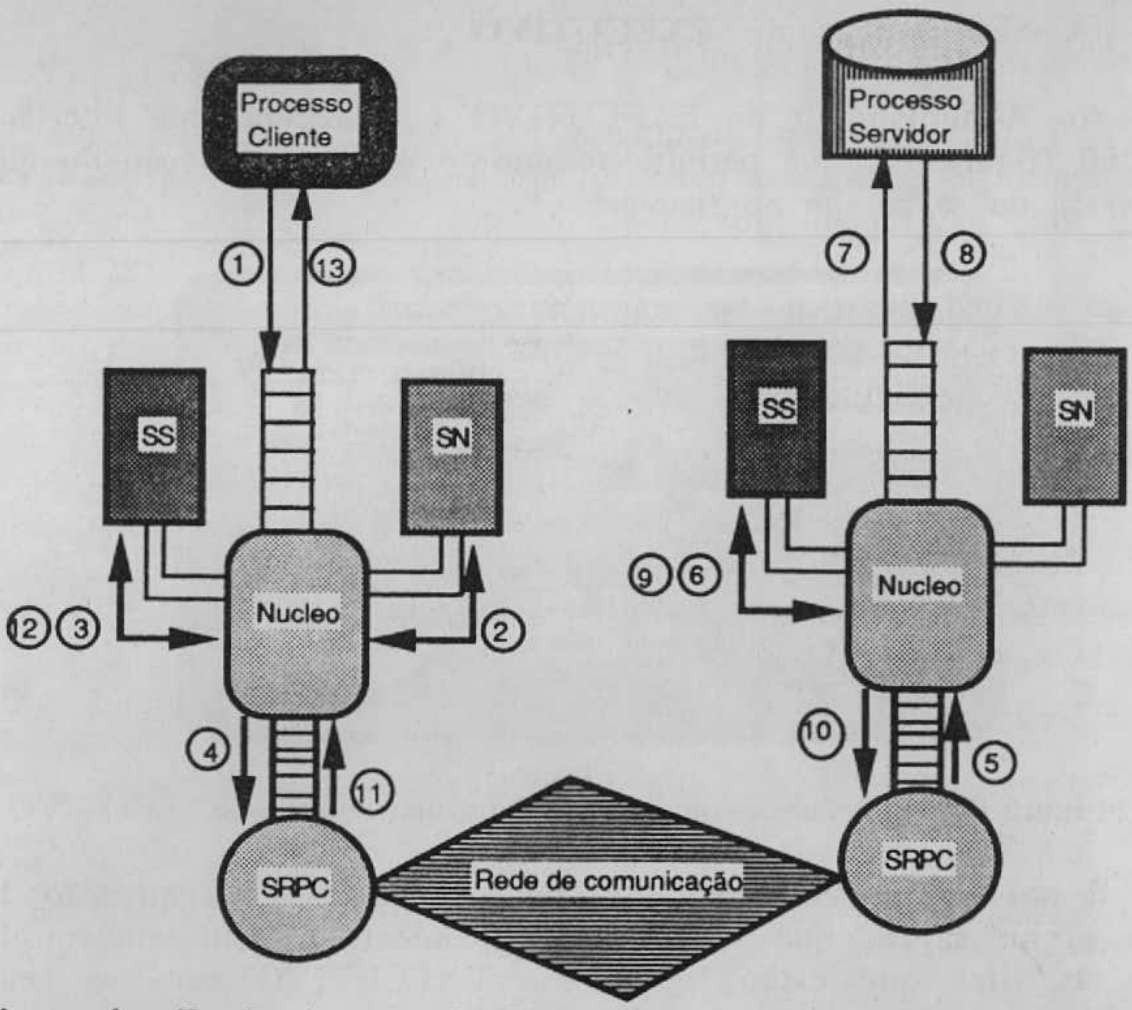

Figura 6 - Envio de uma CALL-mensagem a um processo distante

3 - o Núcleo entrega a mensagem ao Servidor de Sessão para registro em sua tabela (CALL);
4 - o Núcleo entrega a mensagem para o Servidor de RPC tentar encontrar a máquina que possui o processo servidor;
5 - o Núcleo distante recebe a mensagem;
6 - o Núcleo entrega a mensagem para registro pelo Servidor de Sessão distante;
7 - a mensagem é entregue ao processo servidor que a decodifica (campo mensagem);
8 - a Interface E/R distante deposita a mensagem na fila;
9 - o Núcleo repassa a mensagem ao Servidor de Sessão para que esta possa ser removida da tabela de sessão e enviada novamente ao processo cliente;
10 - o Núcleo entrega a mensagem para o Servidor de RPC distante fazer a comunicação diretamente;

O algoritmo se repete até o recebimento da mensagem pelo processo cliente.

VI - A Manutenção do EXECUTIVO

Ao Administrador do EXECUTIVO é fornecida uma interface especial (figura 7) que permite ao mesmo obter uma noção do que se passsa ou se passou no sistema.

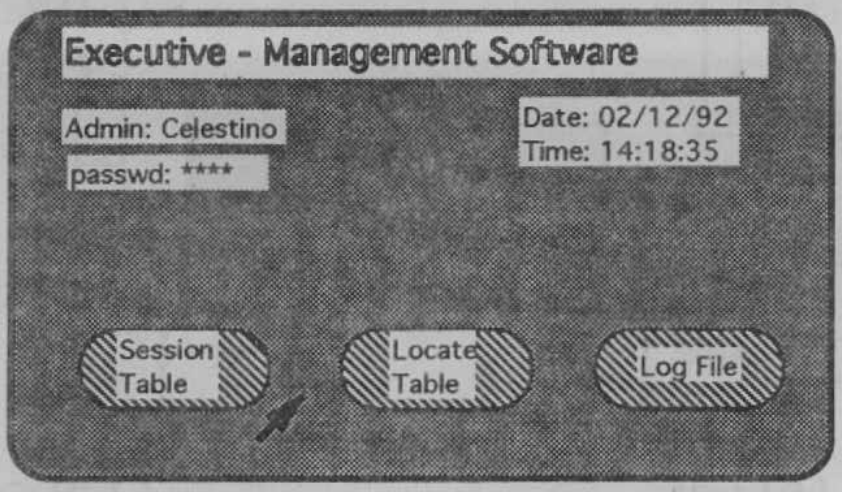

Figura 7 - Interface fornecida ao Administrador do EXECUTIVO

É possível ao Administrador, por exemplo, a visualização de todas as mensagens que se encontram pendentes, como também de todas as filas que estão ligadas ao EXECUTIVO com os seus respectivos donos.

Um arquivo de histórico é fornecido e tem a finalidade de mostrar ao Administrador todas as anomalias que ocorreram no sistema. Por exemplo, uma cópia de uma mensagem que não tenha sido enviada após sucessivos "time-outs" é guardada neste arquivo.

VII - Conclusões

O primeiro protótipo do EXECUTIVO [11] foi baseado no software ANSAware®[12] e foi desenvolvido dentro do projeto ADVANCE. Ele encontra-se rodando em vários países e tem se mostrado robusto. Ele foi desenvolvido com o nome de GMS_DM pelo nosso Laboratório em parceria com uma empresa inglesa.

O segundo protótipo, em uma primeira versão, foi desenvolvido e implementado totalmente em nosso Laboratório e tem sido testado com sucesso. Ele utiliza somente a linguagem "C" e os mecanismos oferecidos pelo UNIX®. Ele está sendo utilizado dentro do projeto PEMMON, como suporte para aplicações de performance, sobre redes TCP/IP e X25.

Uma nova integração permitindo a interconexão através de um módulo CME[13] a um ambiente ISODE foi iniciada.

Uma segunda versão que deverá ser completada nos próximos meses visa a implementação de todas as idéias expostas neste trabalho, conforme descrito abaixo.

Neste momento estamos construindo um pré-compilador para a interface Envia/Recebe, tendo em vista que o envio da mensagem tem sido feito usando campos pré-definidos. A implementação das primitivas REQUEST e COLLECT faz parte desta nova fase.

Uma segunda etapa constará em implementar um algoritmo de prioridade para os processos ditos especiais, já que a leitura destas filas vem sendo feita através de "polling".

Quanto ao Servidor de Sessão a fatia de tempo fornecida as mensagens pendentes está sendo estudada. A idéia é permitir que o próprio Servidor de Sessão possa gerenciar esta tarefa, fornecendo um tempo aleatório baseado em respostas anteriores.

Um ítem, que também está sendo abordado, diz respeito à segurança. A implementação atual do EXECUTIVO não atenta para este problema. Desde de que se conheça as primitivas e a fila multiplexada, qualquer usuário pode ter acesso. Isto é um sério problema tendo em vista que normalmente as máquinas que fazem parte de uma rede de gerenciamento não são dedicadas, pois fazem parte de outras inúmeras redes.

A idéia que será usada nesta segunda implementação será o registro de todos os usuários da rede dentro de um arquivo comum com suas respectivas senhas.

Uma outra etapa será permitir que os processos servidores possam fazer multiprocessamento ("threads").

Uma etapa importante seria a especificação do EXECUTIVO, utilizando técnicas de descrição formal. A idéia que vem sendo amadurecida é a utilização de ESTELLE. Nada nos garante, ainda, que este programa seja robusto. Esta especificação nos ajudaria também a melhor fornecer as transparências a falhas.

Referências

[1]Wade, V., Donnely, W., Celestino, J., Harkness, D., De Souza, J., De Oliveira, M. - "Framework for TMN Computing Platform", 5° RACE TMN Conference, London, Novembro 1991.
[2]Wade, V., Donnely, W., Celestino, J., Harkness, D. - "Experience Designing TMN Computing Platforms for Constrating TMN Management Applications", artigo publicado no livro "The Management of Telecomunications Networks", Editora Ellis Horwood Limited, pags., 225-239, 1992.
[3]Celestino, J., Georges, F., Claudé J-P. - "The EXECUTIVE Delivery Mechanism: A Support of Applications Communications for Network Management", IEEE-ICCT'92 - International Conference on Communication Technology., China, 1992.
[4]Snodgrass, R - "The Interface Description Language: Definition and Use", Editora Computer Science Press, USA, 1989.
[5]ISO 9595 - "Information Processing System - Open Systems Interconnection, Common Management Information Service", 1991.
[6]ISO/IEC JTC1/SC21 N6084 - "ODP Trader", Maio 1991.
[7]EWOS/EGDIR/90/37 - Introduction to OSI Directory Functional Standards - Fifth Draft, Janeiro, 1990.
[8]Mullender, S - "Distributed System" , Editora ACM Press, 1989.
[9]Bloomer, J - "Power Programming with RPC" Editora O'Reilly & Associates, Inc, Fevereiro, 1992.
[10]Stevens, W - "Unix Network Programming", Editora Prentice Hall Software Series, 1990.
[11]Celestino, J - "The GMS Delivery Mechanism", documento interno do projeto ADVACE, ADDN042, 1992.
[12]ANSAware 3.0 Implementation Manual - Documento RM.097.00, Janeiro, 1991.
[13]Agoulmine, N - "Proposition de Mecanismes pour l'interopérabilité des Systèmes de Gestion Ouverts", Tese de Doutorado, Dezembro, 1992.