

# APLICAÇÕES EM SISTEMAS DISTRIBUÍDOS USANDO COMUNICAÇÃO EM GRUPO

Ademir Goulart  
[ademir@univali.br](mailto:ademir@univali.br)

Luis Fernando Friedrich  
[friedrich@inf.ufsc.br](mailto:friedrich@inf.ufsc.br)

ademir@univali.br, UNIVALI - Universidade do Vale do Itajaí, Rua Uruguai 458, CEP 88.302-202 Fax 55 47 3417544 Itajaí (SC) Brasil

friedrich@inf.ufsc.br, UFSC – Universidade Federal de Santa Catarina – INE – Departamento de Informática e Estatística, Campus Trindade – Florianópolis (SC) Brasil

## RESUMO

O presente trabalho visa mostrar o desenvolvimento de aplicações em sistemas distribuídos usando comunicação em grupo. Além de uma aplicação de gerenciamento de recursos em sistemas distribuídos usando um mecanismo de comunicação em grupo outra aplicação de medida de performance é apresentada. O software de comunicação em grupo adotado foi o SPREAD. A aplicação de gerenciamento de recursos controla o uso dos equipamentos, envio de mensagens em broadcast, disponibilidade de micros para uso público, quantidade de disco usado por determinados aplicativos, existência de programas em execução e consulta se determinado usuário está com seu *login* ativo mostrando seu endereço. A aplicação de medida de performance envia mensagens de diferentes tipos e tamanhos a um grupo de clientes, fazendo a coleta automática dos tempos medidos para posterior análise. Como conclusão identificamos que a programação usando mecanismos de comunicação em grupo facilita em muito o desenvolvimento de sistemas pois permite que a comunicação de um processo para muitos seja considerada de forma simples como comunicação de um para um.

**Palavras-chave:** Comunicação em Grupo, Gerenciamento em Sistemas Distribuídos, Performance em Comunicação em Grupo.

## ABSTRACT

*The objective of this paper is to show developing application in distributed systems using group communication. A resource management application in distributed systems and a performance evaluation application are presented. SPREAD was the communication software used to support application. Some of the resources developed enable to manage the length of time the equipment was used, broadcasting of messages, availability of computer for public use, disk space used by some application, if some application is running and also search for one specific user in the network, show its address. In performance evaluation application, the variations in messages size and the type of ordinance generate transmission times that are recorded for later analyze. As conclusion it was identified that programming using group communication mechanism helps application development because process communication from one to many can be considered as communication form one to one.*

Key words: Group communication, Distributed System Management, Resource Management

## 1 Introdução

Redes de computadores estão por toda parte. A internet é uma, como são as muitas redes que a compõem. Redes de telefonia celular, redes corporativas, redes fabris, redes em campus, redes domésticas, redes embarcadas, todas estas, tanto separadamente como em conjunto, compartilham as características essenciais que fazem delas um modelo relevante para o estudo sob o título “**Sistemas Distribuídos**”. Uma definição de sistemas distribuídos é aquela no qual os componentes de hardware e software localizados em computadores interligados por rede, se comunicam e coordenam suas ações somente através da troca de mensagens (COULOURIS, 2001). O presente trabalho tem na seção 2 a conceituação de comunicação em grupo. Na seção 3 apresenta-se a aplicação de gerenciamento de recursos em sistemas distribuídos. O gerenciamento dos recursos em um ambiente de sistemas distribuídos se torna complexo devido a escalabilidade e dispersão geográfica dos componentes que formam este sistema. Saber se determinado usuário está ativo ou seja, está usando o seu *login*, se tem equipamento livre para uso em um ambiente público tal como em laboratórios para estudantes, quanto tempo um micro foi usado ou quanto de recursos em disco está alocado para determinado tipo de arquivo, são gerenciamento de recursos necessários a qualquer instante em ambientes de sistemas distribuídos compostos de um ou de N micros. Na seção 4 apresenta-se o aplicativo para avaliação de performance em um ambiente de sistema distribuído usando comunicação em grupo. Na seção 5 apresentam-se as conclusões.

## 2 Comunicação em Grupo

A comunicação entre os computadores interligados, pode ocorrer de diferentes formas, usando diversos protocolos tanto em ambiente de rede local como ambiente de redes de longa distância. Um caso particular de comunicação ocorre quando uma mesma mensagem tem que ser enviada para diversos computadores na rede, em especial mensagens de solicitação de status para todos os componentes gerenciados. Alguns ambientes físicos de rede permitem o broadcast, onde um único comando de envio manda a mensagem para todos os endereços da rede porém a confiabilidade deste método não é garantida, podendo ocorrer alguma perda sem que o emissor da mensagem seja notificado desta perda.

Visando atender esta necessidade de comunicação de um para muitos, com confiabilidade, escalabilidade, e de forma transparente para quem desenvolve aplicações como estas apresentadas neste trabalho, foi criado um novo paradigma chamado comunicação em grupo. Um protocolo de comunicação em grupo separa a complexidade de controle e gerenciamento das mensagens, da complexidade da aplicação, tratando os diversos computadores como sendo pertencentes a um grupo. Assim podemos nos preocupar apenas com a aplicação, suas funcionalidades, seus procedimentos, seus algoritmos particulares, sem nos envolvermos com os problemas da comunicação. Passa a ser responsabilidade do protocolo de comunicação em grupo o controle de fluxo, a confiabilidade, o sequenciamento e a garantia de entrega das mensagens ao aplicativo de gerência de recursos, bem como o controle de quais computadores estão fazendo parte deste grupo, controlando a entrada de novos membros no grupo, saída voluntária ou saídas involuntárias devido a quebras no computador ou particionamento da rede.

Diversos mecanismos de comunicação em grupo existem tais como: ISIS (BIRMAN, 1994); PHOENIX (MALLOTH, 1996); RMP (WHETTEN, 1994); TOTEM (MOSER, 1996); HORUS (RENESE, 1996); ENSEMBLE (HAYDEN, 1998); TRANSIS (DOLEV, 1996);

NEWTOP (EZHILCHELVAN, 1995); ATOMIC (KOCH, 2000); INTERGROUP (BERKET, 2000); JGROUP (MONTRESOR, 2000) e SPREAD (AMIR, 1998).

Alguns destes mecanismos foram desenvolvidos, tiveram alguma evolução e depois permaneceram sem atualizações. Outros como INTERGROUP e JGROUP são bem recentes e totalmente baseado em ambiente Java/RMI. Já o SPREAD é um dos mecanismos de comunicação em grupo que vem evoluindo ao longo do tempo, com boa documentação, disponibilidade de fontes na modalidade *Open Source*, portado para diversos sistemas operacionais como UNIXes (AIX, BSDI, LINUX, FreeBSD, SGI, MAC OSX, SGI, SOLARIS, SUNos), Windows e Mac OS. Conta com bibliotecas que podem ser usadas em linguagens tipo C, PERL e JAVA.

Para o presente trabalho foi adotado o mecanismo de comunicação em grupo SPREAD (AMIR, 1998) por ser disponível em diversas plataformas ter uma boa documentação e atender tanto comunicação em grupo para redes locais (LAN) como redes de longa distância (WAN). Possui uma interface de programação de aplicação (API) com as seguintes funções:

- SP\_connect – Para estabelecer a comunicação com o *daemon* SPREAD.
- SP\_disconnet – Para terminar a conexão com o *daemon* SPREAD.
- SP\_join – Para iniciar a conexão a um determinado grupo, passando a fazer parte deste grupo.
- SP\_leave – Para deixar de fazer parte de um determinado grupo.
- SP\_multicast – Para enviar uma mensagem a todos os membros de um grupo.
- SP\_receive – Para receber mensagens, tanto mensagens de aplicação como mensagens de transições informando alterações dos membros do grupo.

As aplicações aqui apresentadas neste artigo não são extensão do SPREAD nem acrescentam novas funcionalidades ao mesmo. São ferramentas independentes que usam todas as funcionalidades relativas a comunicação em grupo que são disponibilizadas pelo SPREAD através do uso de suas API anteriormente descritas.

### 3 Aplicativo de Gerência de Recursos

Em um ambiente de sistemas distribuídos encontramos diversos tipos de equipamentos, com diferentes arquiteturas (Intel, PowerPC, SUN, etc) e diferentes sistemas operacionais (Windows, Unix, Mac Os, etc). Uma gerência integrada destes recursos, implica no uso de um software padrão que possa ser executado em todos os ambientes de hardware e software disponíveis no sistema distribuído a ser gerenciado.

Diversos recursos podem ser gerenciados e na sequência vamos detalhar alguns que são objeto desta aplicação:

- Tempo de uso – É o tempo total que um determinado equipamento ficou ligado.
- Equipamento disponível – Quando um equipamento está disponível, sem usuário com *login* ativo.
- Mensagem genérica – Quando por motivos de gerência é necessário enviar uma mensagem única a todos os usuários da rede, ou seletivamente a um determinado grupo de usuários.
- Quantidade de área em disco – Para um determinado tipo de aplicação podemos necessitar consultar o total de área em disco, ocupada em cada um dos micros gerenciados, baseado no nome padrão usado para identificar a extensão. Exemplo, o total de arquivos do tipo doc, exe, zip, etc.
- Localizar um usuário – A partir do nome de *login* do usuário, saber se ele se encontra usando um equipamento, identificado pelo seu *login* ativo e informando em qual equipamento, em qual a localização física do usuário.

- Software em uso – Permite gerenciar quantas cópias de um determinado software estão em uso em neste momento da consulta em todo o ambiente do sistema distribuído.

O gerenciamento dos recursos aqui listados ocorre em tempo real. As consultas quando realizadas fornecem resultados com base em respostas recebidas de todos os componentes que formam o sistema distribuído. Apenas no caso de tempo de uso de um equipamento, a análise dos dados é pertinente a um determinado período de tempo que o equipamento selecionado tenha ficado ativo.

### 3.1 Implementação do Gerenciamento

A aplicação de gerenciamento de recursos em ambiente distribuído é composta de dois ambientes. Um ambiente de coleta de dados para o tratamento exclusivo da consulta de tempo de uso de um recurso no sistema distribuído, na forma de um arquivo *log* registrando quando um micro é ligado ou desligado, que será descrito no item 3.4. Outro ambiente de gerenciamento em tempo real com programas, na forma de mestre escravo. No atual contexto o programa escravo é tratado como um conjunto de escravos que fazem parte do mesmo grupo. Assim um programa mestre, de consulta, que será descrito em 3.3, enviará mensagens aos programas clientes que identificarão o tipo de consulta e fornecerão a resposta adequada, conforme descrito em 3.2.

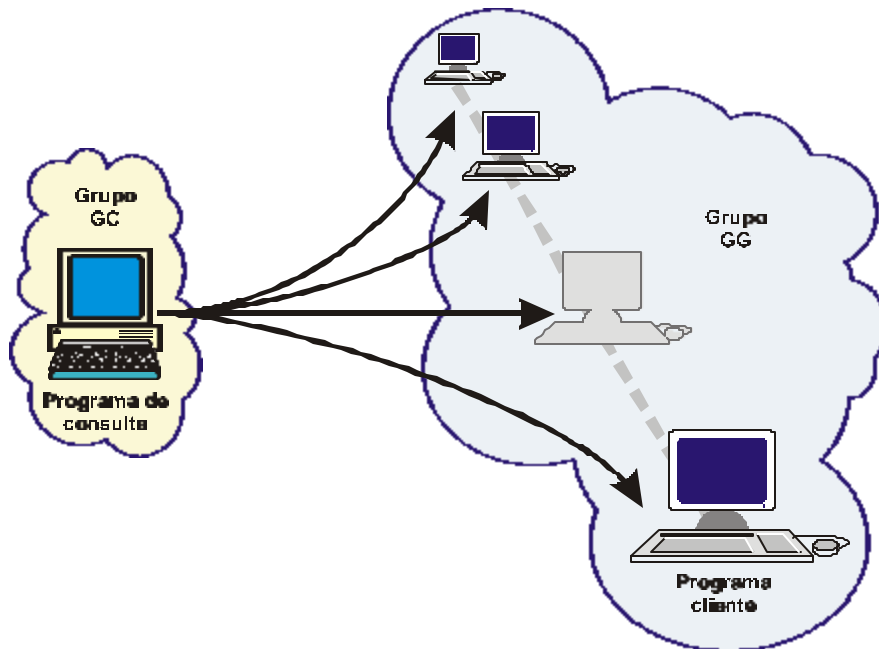


Figura 1 Programa de Consulta e Programas Cliente

### 3.2 Programa Cliente

Implementado para trabalhar em ambiente WINDOWS, como um aplicativo (*daemon*) que fica ativo sempre e sem ícone na barra de tarefas. Quando este programa for iniciado, vai se conectar ao SPREAD com uma identificação única (nome da máquina). Este programa só será

desativado via mensagem enviada pelo programa de consulta, mostrado na Figura 1. Todos os clientes fazem parte de um grupo chamado GG, grupo geral. Este programa cliente é carregado mesmo antes do processo de *login* na estação. Fica sempre aguardando alguma mensagem que vem para o grupo GG ao qual pertence.

O programa cliente terá as seguintes funções:

- Função 1 – Mostra mensagem recebida em uma janela tipo *pop-up*. Exemplo na figura 2.
- Função 2 – Informa se o micro (cliente) está livre, sem usuário com *login* ativo.
- Função 3 – Verifica se o software XYZ está em uso neste cliente.
- Função 4 – Verifica se existem arquivos com a terminação XYZ neste cliente. Se existe, conta quantos arquivos e o total de k bytes destes arquivos. Responde com o número de arquivos e com o total de k bytes.
- Função 5 – Verifica se um determinado usuário está com seu *login* ativo neste cliente. Confere se o *login* enviado na consulta é o corrente neste cliente e se for responde com a identificação da maquina.
- Função Z – Desativa o cliente. Quando recebe este código Z o cliente encerra a sua execução..

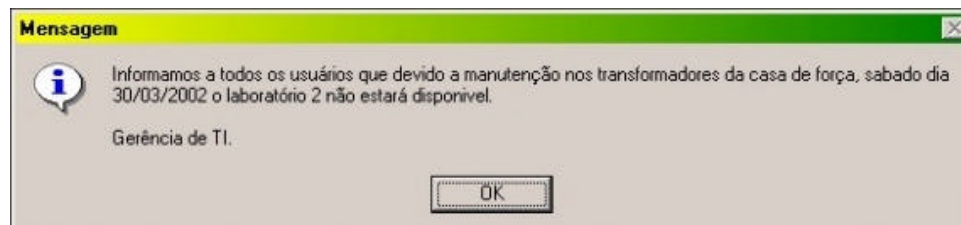


Figura 2 Exemplo de mensagem recebida pelo programa cliente

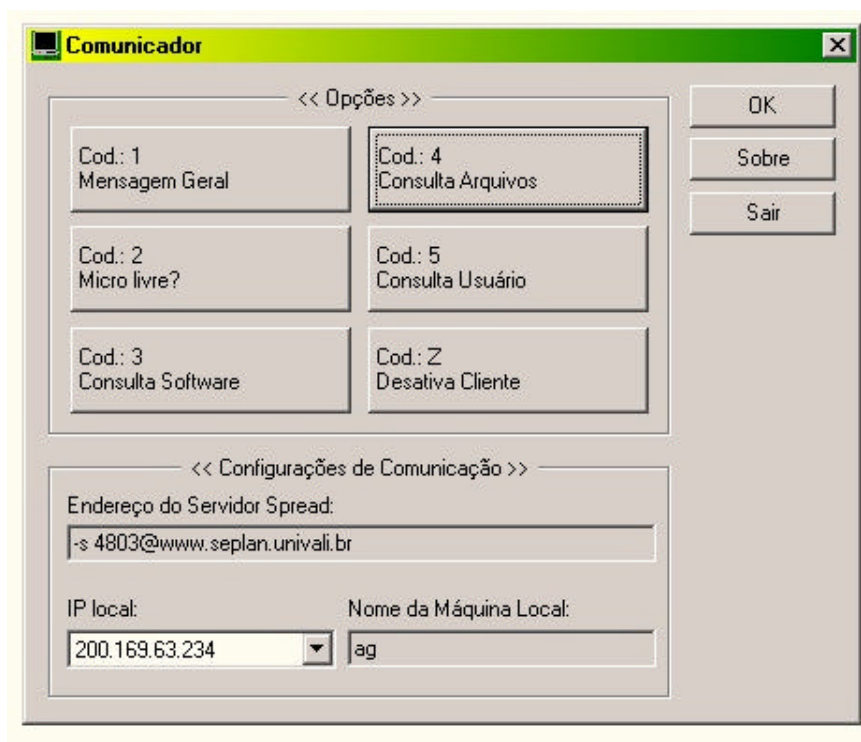


Figura 3 Tela de comandos do Programa de Consulta

### 3.3 Programa de Consulta

Este é um programa que foi implementado em ambiente WINDOWS e que vai interagir com os clientes que estarão fazendo parte do grupo GG. Cada uma das funções pode ser selecionada conforme exemplo da tela mostrada na figura 3. Para cada função selecionada, abre uma nova janela para receber os dados complementares que sejam necessários. Implementa as seguintes funções:

- Função 1 – Mensagem Geral. Esta é uma situação de broadcast geral onde uma mensagem será mostrada em todos os clientes. Recebe a identificação do grupo e a mensagem que é enviada a todos deste grupo.
- Função 2 – Consulta de micro livre. Manda uma mensagem de consulta a micro livre e aguarda respostas. Poderá receber N respostas que serão mostradas em uma janela com barra de rolagem. Assume um *time-out* de 1 minuto para receber todas as respostas. Este tempo de *time-out* é um parâmetro na tela já inicializado com o valor de 1 minuto e que o usuário pode alterar.
- Função 3 – Consulta se o software XYZ está em uso. Exemplo na Figura 4. Recebe do usuário um campo com o nome do software a ser consultado. Manda a mensagem consultando os clientes e aguardar resposta em um determinado tempo conforme descrito na função 2. Uma janela mostrara todas as maquinas que estão executando este software.

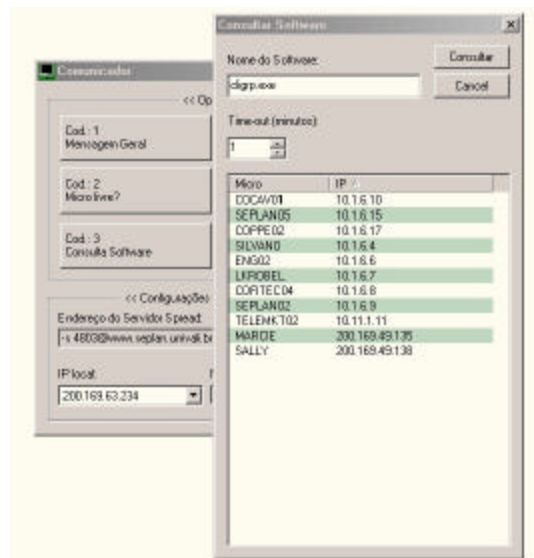


Figura 4 Consulta se determinado software esta em uso

- Função 4 – Consulta se tem arquivos do tipo XYZ na maquina cliente. Exemplo na Figura 5. Recebe do usuário um campo com o tipo de arquivo. Manda a mensagem de consulta a todos os clientes e mostra o resultado em uma janela onde cada linha tem o nome da maquina, número de arquivos e tamanho total dos arquivos. Permite classificar por maquina, número de arquivos, tamanho total. Tem o mesmo controle de *time-out* da função 2.

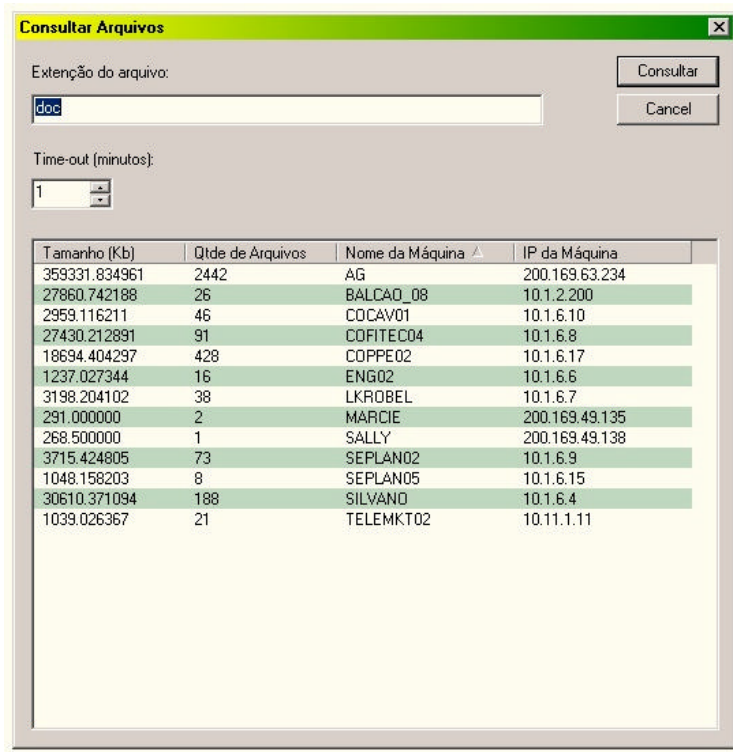


Figura 5 Consulta tipo de arquivos em todas os micros

- Função 5 – Consulta se um determinado usuário está com seu *login* ativo. Recebe o *login* de um usuário e envia uma mensagem com esta consulta.. Se vier resposta informa a localização onde está o usuário, nome da maquina e endereço IP. Usa o mesmo controle de time-out da função 2.
- Função Z – Desativa o programa cliente. Se esta solicitação for efetivada, após validar uma senha de acesso a este recurso, será enviada uma mensagem a todos os clientes e os mesmos serão encerrados. Normalmente um cliente é ativado quando se liga o micro, fica sempre executando, até que a maquina seja desligada.

### 3.4 Programa Gerador de Log de Conexão

No mecanismo de Comunicação em Grupo que foi usado neste trabalho, o SPREAD, descrito anteriormente, todas as informações relativas às transições de estado estão disponíveis automaticamente. Assim para cada cliente que é ativado ou desativado, esta mudança nos membros do grupo se reflete em uma mensagem de transição de estado que é enviada pelo SPREAD para todos os membros do grupo.

Aproveitando esta facilidade foi desenvolvido um programa para executar no mesmo equipamento onde está instalado o SPREAD. Este programa foi implementado no ambiente LINUX e tem como objetivo gravar um arquivo *log* com registros onde são informadas a data e hora em que cada cliente foi ativado e desativado. Com esta facilidade passamos a ter um *log* que nos mostra a hora que o micro foi ligado e a hora que o micro foi desligado. Como as maquinas que estão sendo monitoradas estão na mesma rede local, onde está o *daemon* do SPREAD não temos problemas com particionamento de rede.

Desta forma um programa de visualização seleciona e tabula os registros de *logs* de conexão de um determinado micro, apresentando um relatório de uso (maquina ligada) do mesmo em determinado período de tempo, conforme Figura 6.

Este programa gerador de *log* é ativado como um serviço na partida do sistema e somente será desativado quando o sistema for desligado. O arquivo de *logs* é incremental e é rotacionado na rotina padrão de tratamento dos demais *logs* do sistema.

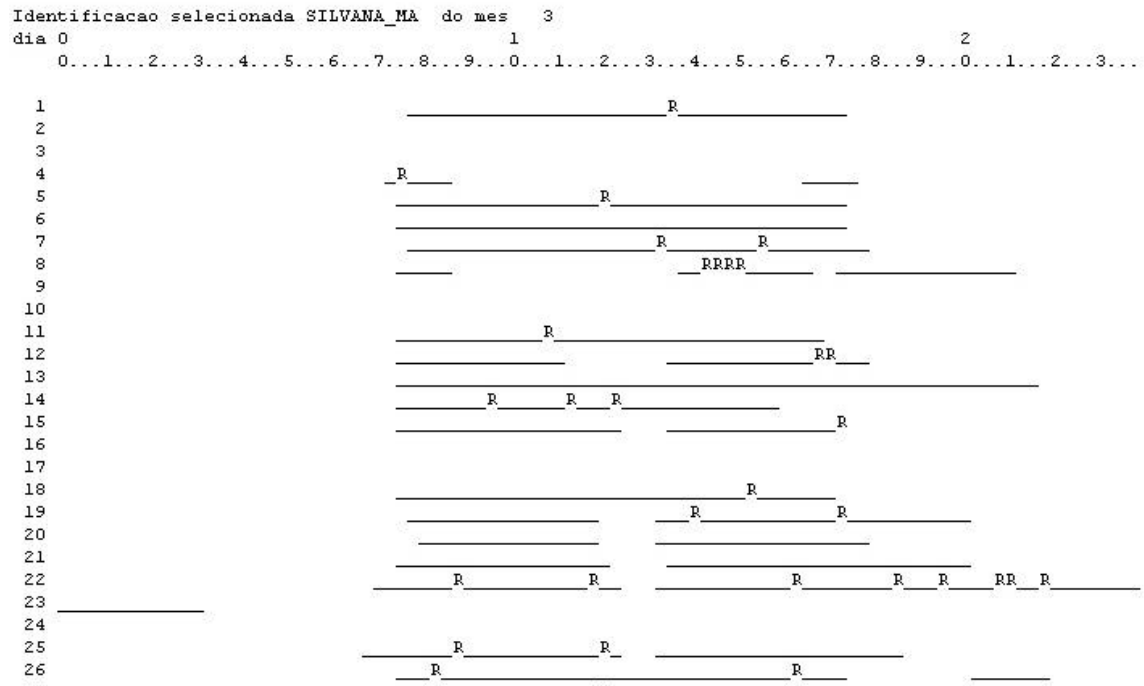


Figura 6 Relatório de uso de um equipamento

#### 4 Aplicativo de Avaliação de Performance

Este segundo aplicativo a seguir apresentado, tem como objetivo medir o tempo de transmissão de um conjunto de mensagens enviadas a diversos aplicativos em um ambiente de sistemas distribuídos. A coleta dos dados também foi implementada usando os recursos do mecanismo de comunicação em grupo. Esta aplicação é composta de três programas. Um programa para o envio de lotes de mensagens a todos os participantes do grupo, o registro do tempo que este lote levou para ser completamente recebido e o cálculo do tempo total para o recebimento do lote de mensagens, são efetuados pelos programas FLOOENV, LOGBM e FLOOREC respectivamente, os quais serão descritos a seguir:

- FLOOENV – Programa que gera as mensagens para envio a todos os membros do grupo chamado “flooder”. Recebe como parâmetros, o número de mensagens a enviar, o tamanho da mensagem e o tipo de ordenação que estas mensagens devem ter.



- LOGBM – Este programa está associado ao grupo “log” e recebe mensagens de início e de fim de lote, provenientes de todos clientes (FLOOREC) que fazem parte do grupo “flooder”. Os registros recebidos são gravados em um arquivo de log em disco para posterior análise.
- FLOOREC – Uma vez ativado, este programa entra em um laço de recebimento das mensagens enviadas ao grupo “flooder” ao qual pertence. No início de cada lote um registro cabeçalho é recebido, onde são informados as características do lote e o número de mensagens deste lote. Inicializa o contador de tempo, informa ao grupo “log” uma mensagem com todos os parâmetros desta execução além da data e hora de início do recebimento do lote. Ao final do lote envia outro registro de controle para o grupo “log” confirmando o numero de mensagens recebidas, data e hora de final além do tempo total de duração deste recebimento.

Na Figura 7 é mostrado um esquema de interligação destes programas, sendo que o numero de clientes FLOOREC é genérico variando de 1 a N clientes.

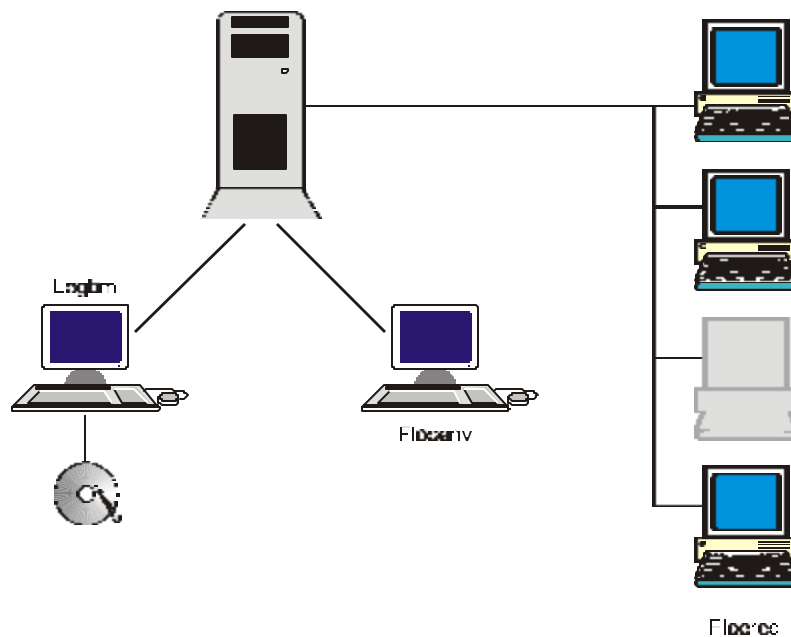


Figura 7. Interligação dos programas FLOOENV, LOGBM e FLOOREC.

#### 4.1 Arquitetura da Rede

A rede empregada para os testes é composta de diversos laboratórios, em diferentes blocos e em dois diferentes campi. Os *links* internos a cada campus estão conectados com *fast ethernet* e *gigabit ethernet*. Todos os micros clientes estão ligados a portas de *switch fast ethernet*. Entre os dois campi o link WAN é de 2 *Megabits*. A Figura 8 mostra a arquitetura da rede em sua versão completa com 167 micros clientes, usados na avaliação aqui descrita. Uma segunda rede usada para comparação, apenas com 9 equipamentos, mantinha a mesma topologia apenas com 8

equipamentos no campus I e 1 no campus II. Todos os testes foram efetuados em um final de semana quando o tráfego normal da rede se apresentava nulo ou insignificante para interferir nas medidas encontradas.

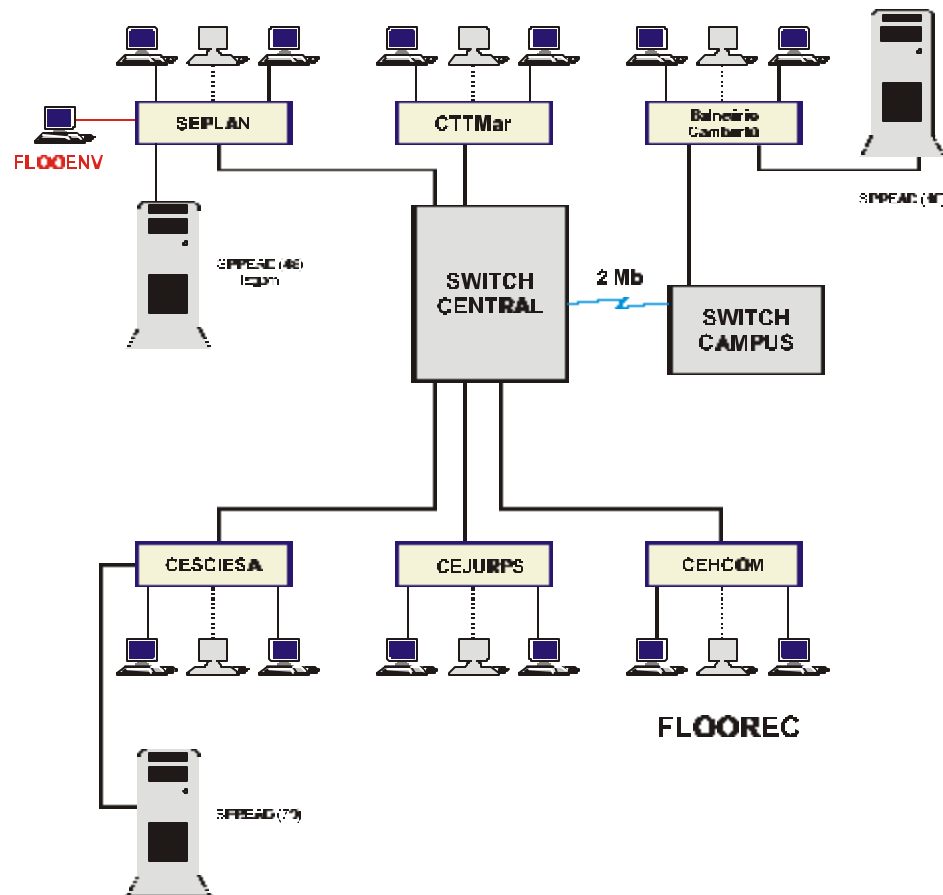


Figura 8. Arquitetura da rede durante os testes de performance.

## 4.2 Resultados Obtidos

Para os testes de performance e escalabilidade realizaram-se diversas rodadas de execução do programa FLOOENV, variando o tipo de ordenação, o tamanho da mensagem e o número de mensagens enviadas. Os resultados foram tabulados para cada um dos cenários de teste conforme mostra a Figura 9.

Hora Início	Mensagens do tipo					Reliable
	Tamanho da mensagem					100 bytes
	Ambiente de rede com 167 micros					QR3F
	numero mensagens	quantidade amostras	tempo em segundos			
			menor	maior	Médio	
13:42:40	100	152	0	2	0,35	
13:43:08	200	152	1	2	1,19	
13:43:50	300	152	1	36	8,12	
13:45:15	400	152	2	3	2,42	
13:45:52	500	152	2	4	2,83	
13:47:49	1000	152	4	5	4,72	
13:48:26	2000	152	7	9	7,95	
19:49:06	3000	152	12	13	12,15	
13:49:53	4000	152	15	17	15,44	
13:50:34	5000	152	26	29	28,05	
13:51:29	10000	73	97	98	97,39	
	Mensagens do tipo					Safe
	Tamanho da mensagem					100 bytes
	Ambiente de rede com 167 micros					QR4F
	numero mensagens	quantidade amostras	tempo em segundos			
			menor	maior	Médio	
13:54:48	100	73	0	1	0,63	
13:55:18	200	73	1	2	1,19	
13:55:49	300	73	1	2	1,35	
13:56:18	400	152	2	4	2,43	
13:56:50	500	152	2	4	2,67	
13:58:21	1000	152	4	7	5,07	
13:58:55	2000	152	7	9	8,14	
13:59:33	3000	133	13	14	13,84	
14:01:51	4000	133	19	21	19,93	
14:02:59	5000	133	16	18	16,90	
14:03:52	10000	133	33	35	33,95	
14:05:24	100000	132	335	337	336,28	

Figura 9. Tabulação dos resultados medidos.

Como pode ser observado para cada medida tem-se uma amostragem de 152 máquinas respondendo, sendo que as 13:51 horas muda para 73 máquinas respondendo e as 13:56 horas voltam a responder novamente 152 máquinas. Isto mostra um particionamento ocorrido na rede onde um grupo de 79 máquinas fica isolado do teste durante alguns minutos, retornando automaticamente ao estado inicial de 152 máquinas no grupo, após restabelecer a conexão na rede.

A comparação de tempos para o ambiente de rede com 9 máquinas versus o ambiente com 167 máquinas é mostrada no gráfico da Figura 10.

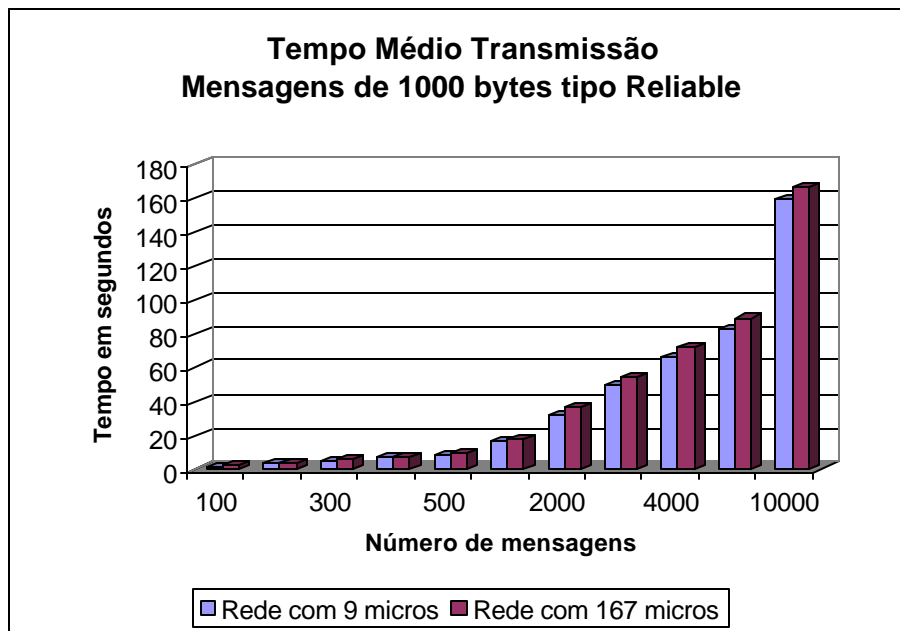


Figura 10. Tempo médio para mensagens em rede com 9 micros X rede com 167 micros.

Outra avaliação de forma comparativa leva em consideração o tamanho das mensagens. Assim para diferentes volumes foi observado o tempo médio de entrega de mensagens com 100 bytes versus mensagens com 1000 bytes. Na Figura 11 tem-se o gráfico destas medidas.

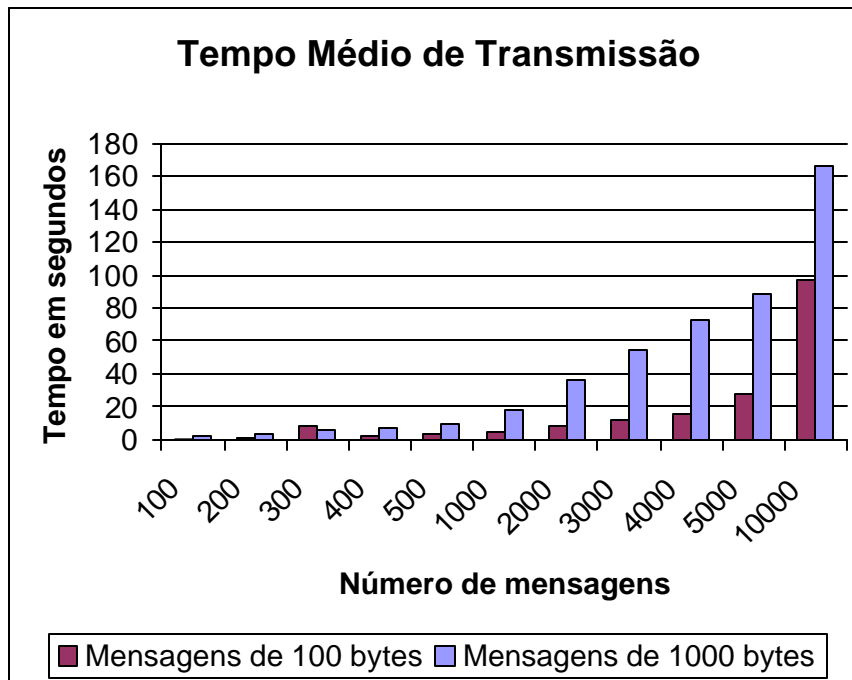


Figura 11. Tempo médio para mensagens de 100 bytes X 1000 bytes.

É importante considerar que devido ao grande número de amostras para cada tomada de tempo, o processo de coleta dos dados foi totalmente automatizado pois cada micro cliente na

ponta final, envia uma mensagem de início e outra de fim com o tempo decorrido para o recebimento do lote de mensagens em sua estação. Estas mensagens enviadas ao grupo “log” são todas gravadas em um arquivo texto para posterior análise, dando origem aos gráficos aqui apresentados.

## 5 Conclusões

O uso de um mecanismo de comunicação em grupo permite total abstração quanto a escalabilidade no contexto de gerenciamento dos recursos em um sistema distribuído. Todos os recursos estejam eles na rede local ou em redes de longa distância podem ser gerenciados simultaneamente pelo mesmo programa de consulta. Para o aplicativo de avaliação de performance podemos ter qualquer número de máquinas clientes. Em termos práticos é recomendado um gerenciador SPREAD em cada segmento de rede, controlando os equipamentos desta rede local. Estes gerenciadores se integram formando um sistema único de controle de grupo permitindo que uma consulta feita em qualquer ponto da rede seja respondida por todos os outros pontos que compõem este grupo mesmo estando em segmentos de redes distintos. Uma eventual queda de conexão entre dois segmentos vai particionar o ambiente em dois grupos distintos e quando houver a reconexão automaticamente é feita uma reorganização voltando a ter um grupo único com todos os micros do sistema distribuído, idêntico à situação anterior à queda da rede. Outra grande vantagem é podermos integrar num único gerenciamento diferentes arquiteturas de hardware e sistemas operacionais.

Esta ferramenta de gerenciamento de sistemas distribuídos, usando comunicação em grupo não foi comparada com produtos similares pois não encontramos produtos semelhantes usando comunicação em grupo. Este artigo contribui para divulgar o uso de mecanismos de comunicação em grupo mostrando o seu uso prático em um software de gerenciamento de recursos em sistemas distribuídos.

Quanto ao uso em aplicações, como os programas de medida aqui empregados, este estudo mostra a importância dos mecanismos de comunicação em grupo pois com o uso deste recurso pode-se abstrair de quantas máquinas clientes tem-se no ambiente, o qual estamos querendo avaliar.

Quanto à performance e escalabilidade, pode-se observar que o ambiente SPREAD tem ótima escalabilidade, produzindo tempos de resposta plenamente adequados durante os testes com a variação do número de componentes envolvidos. Tanto no volume de pequena rede com 9 micros ou no teste chamado de rede completa com 167 micros tem-se um tempo de resposta muito semelhante o que nos permite concluir que o SPREAD funciona de forma eficiente quando se aumenta de dezenas para centenas de máquinas. Naturalmente deve-se considerar que no teste de pequena rede foram usados dois *daemon* SPREAD e no teste de rede completa foram usados três *daemon* SPREAD. Este mecanismo de comunicação em grupo escala muito bem, bastando para isso apenas ir acrescentando novos *daemon* SPREAD na rede, a medida que o número de clientes for incrementado em algumas dezenas.

Outra consideração a ser feita sobre performance está relacionada com o tipo de ordenação das mensagens. Embora o SPREAD trabalhe com diferentes tipos de ordenação de mensagens tais como sem ordem, FIFO, CAUSAL e SAFE, para o tipo de teste efetuado, basicamente não se tem diferença de tempo. O processo de ordenação faz sentido quando temos diversas fontes de envio simultâneo para o grupo. No teste descrito, como é única a fonte de envio para o grupo, deixa de existir a importância desta característica de ordenação.

## Referências Bibliográficas

AMIR, Y.; STANTON, J. **The SPREAD Wide Area Group Communication System**. Technical report, Center of Networking and Distributed Systems, Johns Hopkins University, Baltimore, Mariland, 1998. Disponível na internet URL: <http://www.cnds.jhu.edu/publications/> capturado em agosto de 2001.

BERKET, K., **The InterGroup Protocols: Scalable Group Communication for the Internet**. Dissertation, University of California – USA 2000. Disponível na internet URL <http://www-itg.lbl.gov/InterGroup> capturado em dezembro de 2001.

BIRMAN, K; RENESSE, R. van, **Reliable Distributed Computing with the ISIS Toolkit**, IEEE Computer Society Press, 1994.

COULOURIS G.; DOLLIMORE J.; KINDBERG T., **Distributed Systems Concepts and Design**, Addison Wesley, Third Edition, 2001

DOLEV, D.; MALKI, D.; **The Transis Approach to High Availability Cluster Communication.**, Communications of the ACM, 39(4), April 1996

EZHILCHELVAN, P.; MACEDO, R.; SHRIVASTAVA, A. **NEWTOP: A Fault-tolerante Group Communication Protocol**. Proceedings of the 15<sup>th</sup> IEEE International Conference on Distributed Computing Systems, pag 296-306, Vancouver, Canada, maio 1995.

HAYDEN, M. **The Ensenble System**, PhD thesis, Department of Computer Science, Cornell University, January 1998

KOCH, R.R. **The Atomic Group Protocols: Reliable Ordered message delivery for ATM networks** PhD Dissertation, Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA, December 2000

MALLOTH, C; **Conception and Implementation of a Toolkit for Building Fault-Tolerant Distributed Applications in Large-Scale Networks**, PhD thesis , Ecole Polytechnique Federale de Lausanne, 1996.

MONTRESOR, A., **System Support for Programming Object-Oriented Dependable Applications in Partitionable Systems**, Technical Report UBLCS-2000-10 Department of Computer Science, University of Bologna, Bologna – Italy, 2000. Disponível na internet URL <http://www.cs.unibo.it> capturado em dezembro de 2001.

MOSER, L. E.; MELLIAR-SMITH, P.M.; AGARWAL, D.A.; BUDHIA, R.; LINGLEY-PAPADOULOS, C. **Totem: A Fault-Tolerant Group Communication System**. Communications of the ACM, 39(4) , April 1996.

RENESSE, R. van; BIRMAN, K.P.; MAFFEIS, S. **Horus: A Flexible Group Communication System**, Communications of ACM, 39(4):76-83, April 1996.

WHETTEN, B.; MONTGOMER, Y., T.; KAPLAN, S., **A High Performance Totally Ordered Multicast Protocol**. Proceedings of the International Workshop on Theory and Practice in Distributed Systems, pag 33-57, Dagstuhl Castle, Alemanha, Setembro 1994.