GERENCIAMENTO DE RECURSOS EM SISTEMAS DISTRIBUÍDOS USANDO COMUNICAÇÃO EM GRUPO

Ademir Goulart[1]
ademir@univali.br

Luis Fernando Friedrich[2]
friedrich@inf.ufsc.br

UNIVALI - Universidade do Vale do Itajaí
Rua Uruguai 458, CEP 88.302-202 Itajaí (SC)

UFSC – Universidade Federal de Santa Catarina
INE - Departamento de Informática e Estatística,
Campus Trindade – Florianópolis (SC)

RESUMO

O presente trabalho visa mostrar uma aplicação de gerenciamento de recursos em sistemas distribuídos usando um mecanismo de comunicação em grupo. O software de comunicação em grupo adotado foi o SPREAD e algumas aplicações desenvolvidas permitem gerenciar recursos tais como tempo de uso dos equipamentos, envio de mensagens em broadcast, disponibilidade de micros para uso público, quantidade de disco usado por determinados aplicativos, existência de aplicativos em execução e consulta se determinado usuário está com seu *login* ativo mostrando seu endereço.

Palavras-chave: Comunicação em Grupo, Gerenciamento em Sistemas Distribuídos, Gerenciamento de recursos.

ABSTRACT

The objective of this paper is to show a resource management application in distributed systems using group communication. SPREAD was the communication software used to support the management 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.

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 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 aplicativo, são gerenciamento de recursos necessários a qualquer instante em ambientes de sistemas distribuídos compostos de um ou de N micros. O presente trabalho descreve uma ferramenta desenvolvida para gerenciamento de recursos, usando a forma de comunicação conhecida como comunicação em grupo. No capítulo 2 temos uma descrição dos recursos que são gerenciados por esta ferramenta. No capítulo 3 as características de comunicação em grupo são mostradas bem como detalhes do mecanismo de comunicação em grupo SPREAD o qual foi usado como suporte para esta aplicação de gerenciamento. No capítulo 4 é detalhado como foi feita a implementação do software de gerenciamento e no capítulo 5 apresentamos a conclusão.

2 RECURSOS A GERENCIAR

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 seqüência vamos detalhar alguns que são objeto deste trabalho:

- 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, a análise dos dados é pertinente a um determinado período de um determinado equipamento.

3 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 uma aplicação como esta de gerenciamento, 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 (RENESSE, 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.

A aplicação aqui apresentada neste artigo não é uma extensão do SPREAD nem acrescenta novas funcionalidades ao mesmo. É uma ferramenta independente apenas usando todas as funcionalidades relativas a comunicação em grupo que são disponibilizadas pelo SPREAD através do uso de suas API anteriormente descritas.

4 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 4.3. 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 4.2, enviará mensagens aos programas clientes que identificarão o tipo de consulta e fornecerão a resposta adequada, conforme descrito em 4.1.

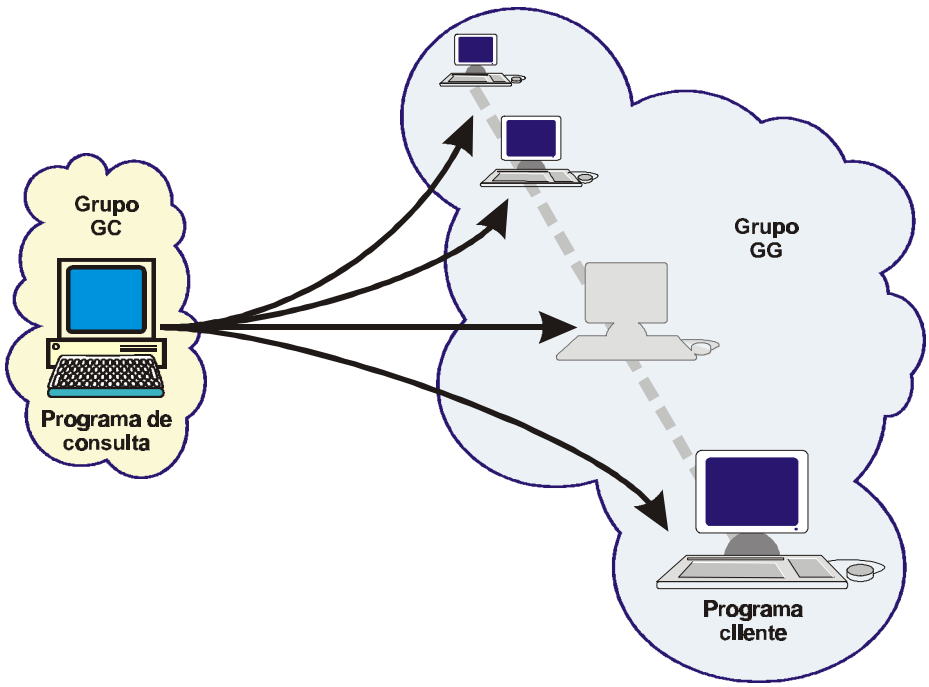

Figura 1: Programa de Consulta e Programas Cliente

4.1 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 máquina.
- Função Z – Desativa o cliente. Quando recebe este código Z o cliente encerra a sua execução.

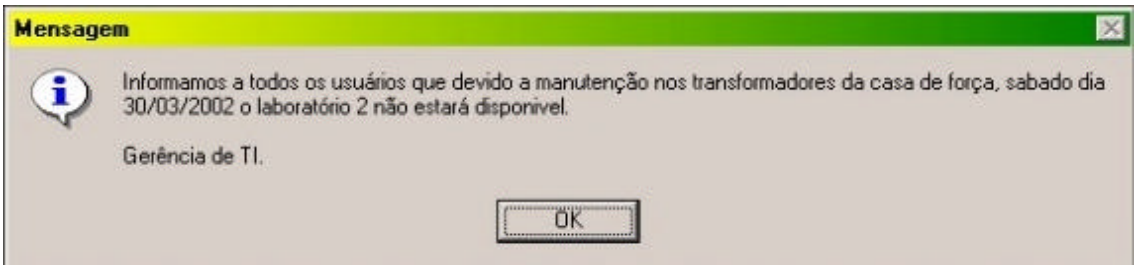

Figura 2: Exemplo de mensagem recebida pelo programa cliente

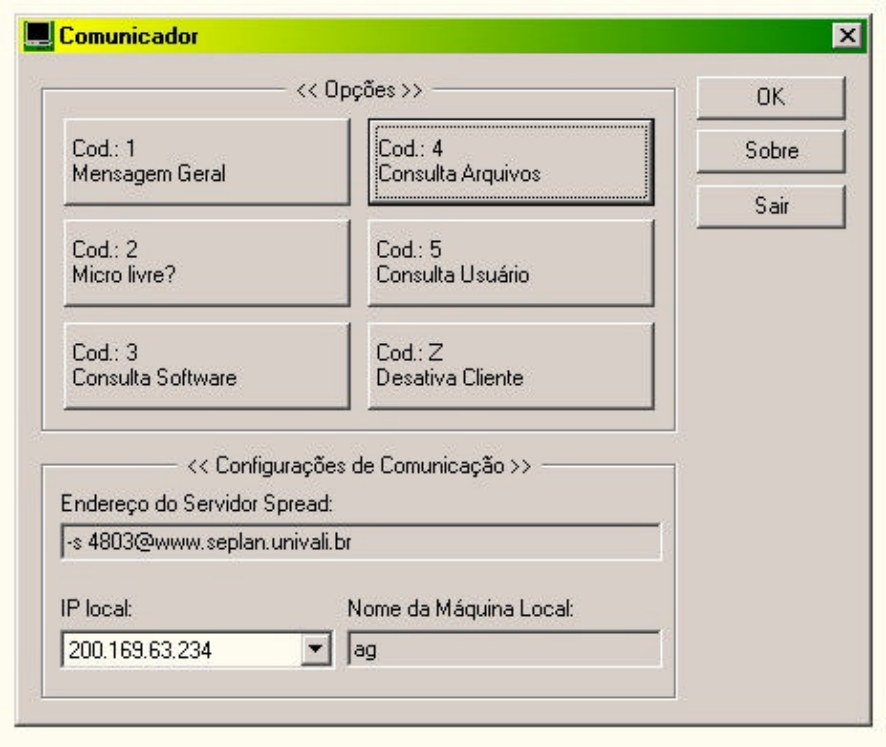

Figura 3: Tela de comandos do Programa de Consulta

4.2 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. Temos um botão para selecionar cada uma das funções 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. Exemplo conforme Figura 4.

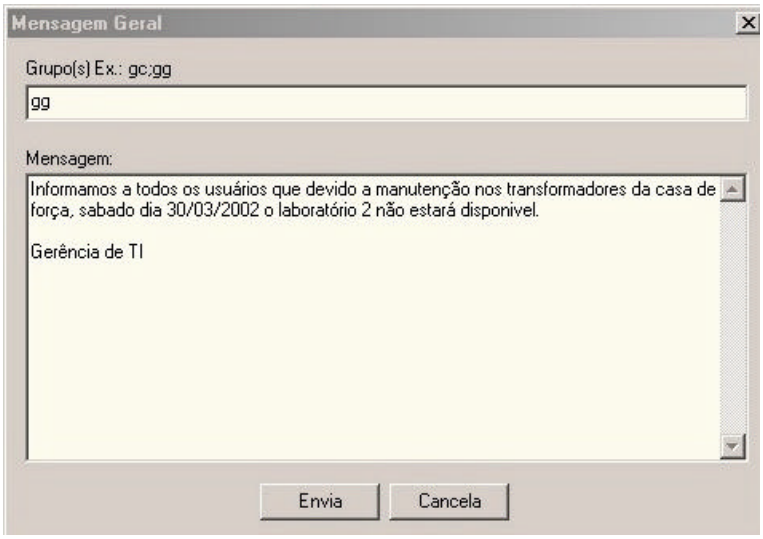

Figura 4: Envio de mensagem genérica

- Função 2 – Consulta de micro livre. Exemplo na Figura 5. 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 incializado com o valor de 1 minuto e que o usuário pode alterar.

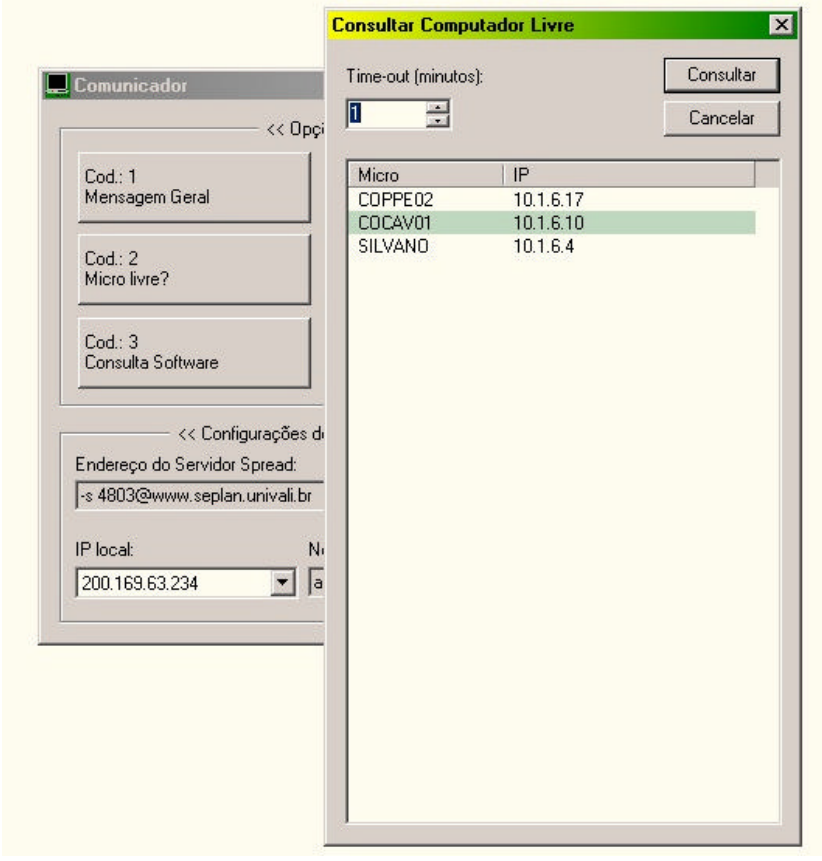

Figura 5: Consulta de micro livre

- Função 3 – Consulta se o software XYZ está em uso. Exemplo na Figura 6. 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.

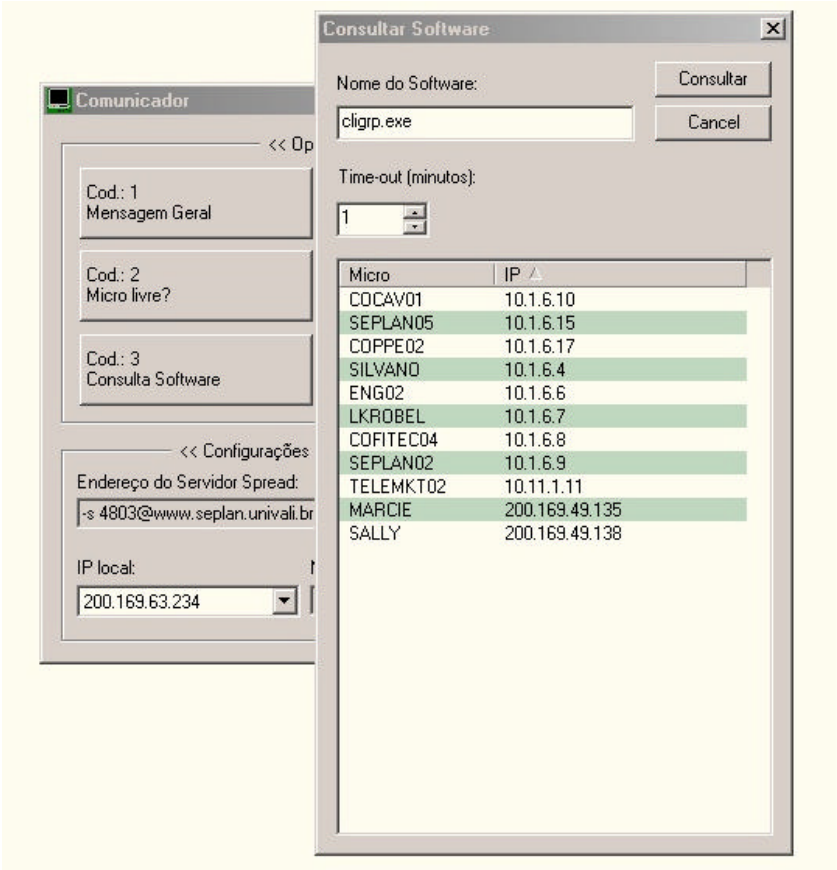

Figura 6: Consulta se determinado software esta em uso

- Função 4 – Consulta se tem arquivos do tipo XYZ na maquina cliente. Exemplo na Figura 7. 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.

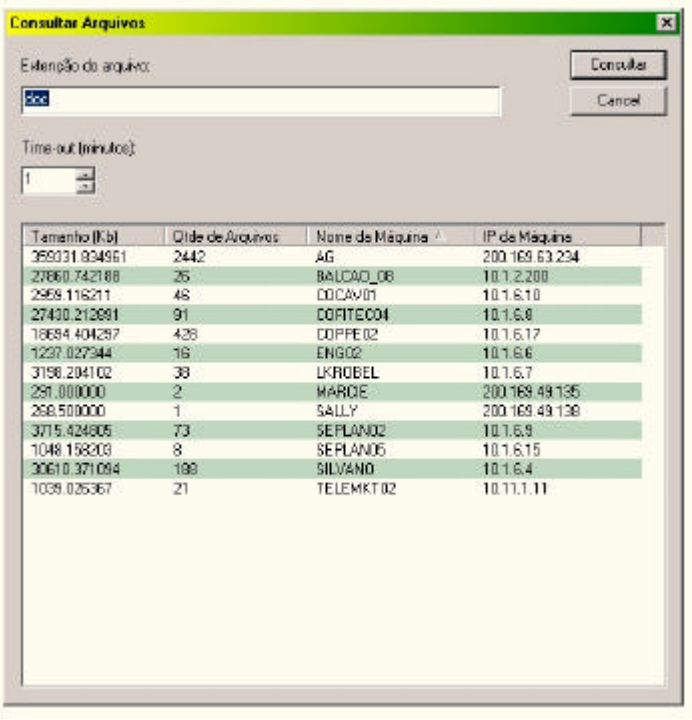

Figura 7: Consulta tipo de arquivos em todas os micros

- Função 5 – Consulta se um determinado usuário está com seu *login* ativo. Exemplo na Figura 8. 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.

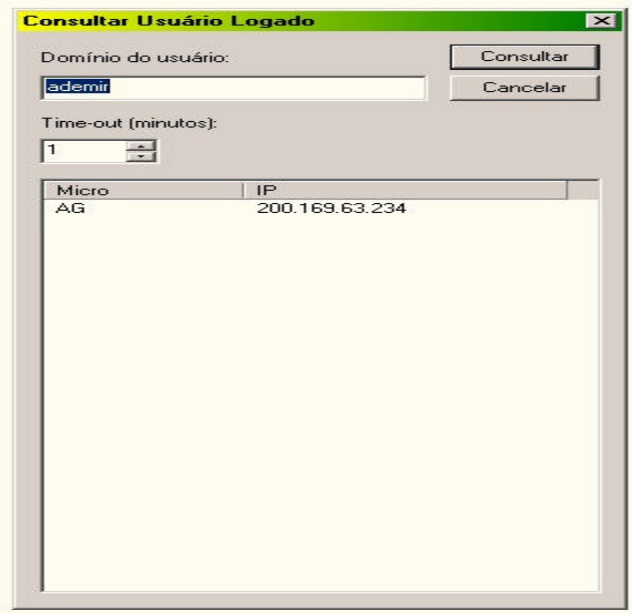

Figura 8: Consulta usuário

- 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.

4.3 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 ficar executando 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 9.

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.

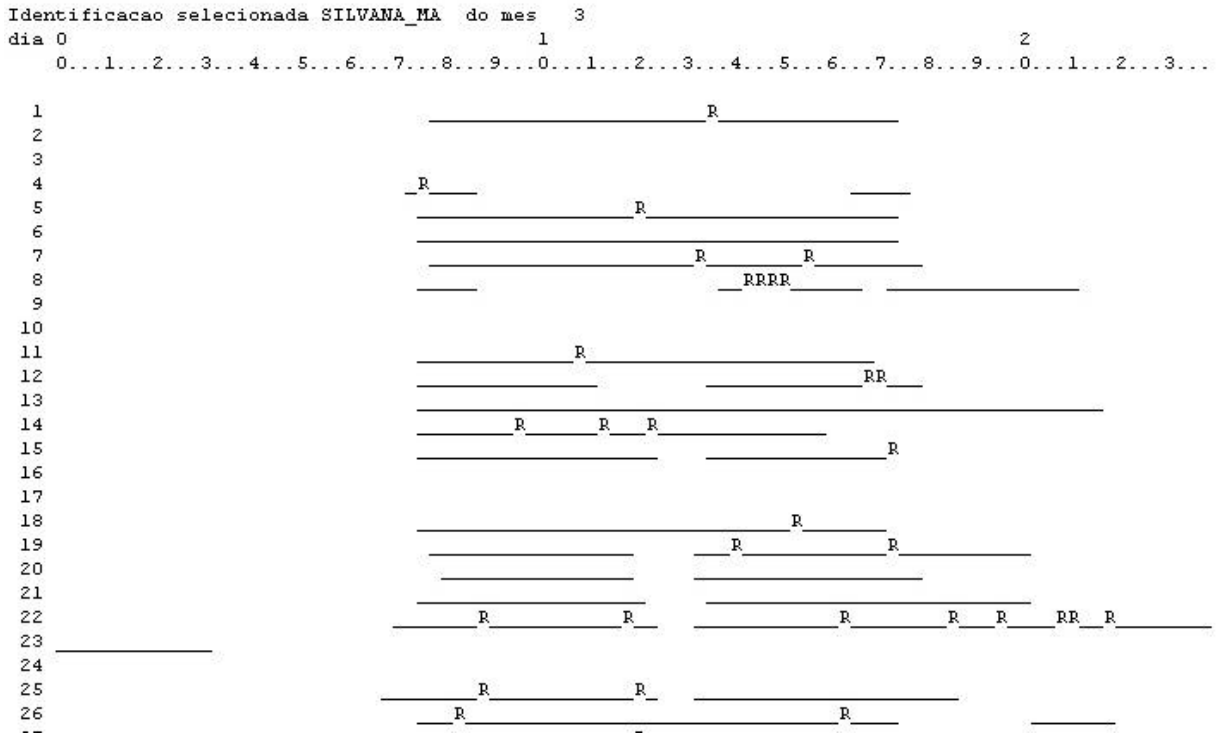

Figura 9: Relatório de uso de um equipamento

5 CONCLUSÃO

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. 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.

Como trabalhos futuros fica a sugestão de portar o aplicativo cliente para outras plataformas como MAC e LINUX. Também uma versão do programa de consulta em JAVA seria muito interessante pois permitiria o uso do mesmo em ambiente WEB.

6 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 15th 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.