Desenvolvimento de Sistemas de Gerência de Agregados para Plataforma GNU/Linux

Marco Aurélio Stelmar Netto, Alex de Vargas Barcelos,

César Augusto Fonticielha De Rose

Faculdade de Informática – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Av. Ipiranga, 6681 – 90619-900 – Porto Alegre – RS – Brazil
stelmar@cpad.pucrs.br,alexb@pucrs.br,derose@inf.pucrs.br

Abstract. *The area of high performance processing has seen an increase in the development of tools and systems based on the idea of free software. Cluster computing is one of the architectures that supplies the needs of such area. In this context a Cluster Management System, called Crono, has been developed. Its main characteristics is to keep resources with high availability during the maximum period of time so that users' applications can have higher performance. This paper presents the Crono system and its development process on a free software environment. More specifically, Crono has been developed for the GNU/Linux platform.*

Resumo. *Uma forte tendência verificada nos últimos anos remete ao fato de que a área de processamento de alto desempenho está voltada a utilização de ferramentas e sistemas baseados em software livre. Uma das arquiteturas criadas para suprir as necessidades desta área é baseada em máquinas agregadas. Neste contexto foi desenvolvido um sistema de gerência, chamado Crono, cujo objetivo é gerenciar arquiteturas deste tipo, mantendo os recursos com alta disponibilidade, durante o maior tempo possível, para que os usuários obtenham o maior desempenho de suas aplicações. Este artigo apresenta o sistema Crono e seu processo de desenvolvimento sob um ambiente de software livre, baseado na plataforma GNU/Linux.*

1. Introdução

Ambientes de pesquisa e até mesmo grandes corporações vêm necessitando de alto poder computacional a baixo custo, para as mais variadas tarefas como renderizar gráficos em alta velocidade, prever o clima, fazer simulações de experimentos atômicos e outras tarefas de alto desempenho. Em certas situações, quando necessidades de performance podem rapidamente tornar o equipamento insuficiente para a tarefa, um único computador não é o mais adequado. Manter apenas um computador realizando uma tarefa importante também não é uma garantia segura de que o serviço vai estar sempre disponível, pois problemas de hardware ou software podem causar a interrupção do serviço.

A tendência que surgiu para suprir estas necessidades são as arquiteturas baseadas em agregados. A idéia é agrupar estações de trabalho (nós) através de uma rede local, para que trabalhem de forma integrada a fim de se obter um poder de processamento equiparável a um supercomputador. E o que é melhor, pelo preço de alguns microcomputadores [Manika 2000].

Juntamente com o surgimento dessas arquiteturas, veio a necessidade de desenvolver sistemas para gerenciá-las de maneira adequada, a fim tirar o máximo de proveito que elas

podem oferecer. E esta necessidade vem se tornando um dos grandes desafios na área de processamento paralelo e distribuído.

A gerência de agregados tem como principal objetivo manter os recursos com alta disponibilidade, durante o maior tempo possível, para que os usuários obtenham o maior desempenho de suas aplicações. Isso pode ser alcançado através de sistemas que permitam, ao administrador, um eficiente controle de uso dos recursos utilizados pelas aplicações e, aos usuários, mecanismos de execução dos programas de forma simples. Sistemas que fornecem estas características são chamados de CMS (*Cluster Management System*, ou Sistemas de Gerenciamento de Agregados - SGA) [Baker, Fox e Yau 1995].

O estudo de caso apresentado é o sistema gerenciador de agregados Crono, sendo que o enfoque deste artigo é mostrar a importância do ambiente de software livre no desenvolvimento deste tipo de sistema.

2. GNU/Linux como Ambiente de Desenvolvimento

O GNU/Linux por ser um sistema operacional bastante utilizado em máquinas agregadas (*clusters*) e também por ser baseado em software livre, fornece grandes vantagens ao desenvolvimento de SGA´s.

Uma das principais vantagens do software livre é que os sistemas desenvolvidos neste ambiente possuem código aberto e estão disponíveis para pesquisa. Isto certamente facilita o desenvolvimento de novos *softwares* de uma forma mais eficiente devido ao reaproveitamento de rotinas e idéias já exploradas. Existe também a possibilidade de alterar ferramentas já existentes a fim de cooperarem de maneira adequada a um novo *software*.

Com o GNU/Linux, os SGA´s podem facilmente manipular os arquivos de configuração do sistema operacional, permitindo definir restrições de acesso e variáveis de ambiente e enviar informações inesperadas aos usuários conectados a máquina hospedeira, através dos terminais do sistema operacional.

Um SGA gerencia recursos de máquinas agregadas, que podem ser o tempo de uso e número de nós por usuários. Entretanto ele também pode gerenciar recursos de mais baixo nível. Alterando o núcleo do GNU/Linux para se adequar ao SGA, pode-se, por exemplo, implementar políticas de uso de memória, de processador e de rede.

Os SGA´s podem ser construídos utilizado o compilador C da GNU (gcc) que é outro recurso bastante difundido entre os desenvolvedores, tanto para ambientes de software livre como para comercial. Os SGA´s geralmente são constituídos de um conjunto de processos, sendo que estes precisam trocar informações entre si. Para isto também estão disponibilizados, sob forma de software livre, as POSIX Threads (pthreads) [Lewis e Berg 1998] e os sockets [Stevens 1998].

3. Sistema de Gerência de Agregados Crono

O sistema para gerência de agregados Crono tem como objetivo controlar os usuários quanto ao uso de recursos em um ou vários agregados. Esse tipo de controle se faz necessário, principalmente, para que as aplicações dos usuários não interfiram umas nas outras. O Crono fornece dois tipos de alocação de recursos. O primeiro, chamado de alocação espacial, é onde os usuários têm exclusividade no uso dos nós. E o segundo, chamado de alocação temporal, é disponibilizado para uso compartilhado dos nós. Este último é utilizado, por exemplo, nos casos onde os usuários estejam apenas testando suas aplicações e, consequentemente, não necessitam de performance.

A arquitetura do sistema Crono é formada por 5 componentes, sendo eles a Interface do Usuário, o Gerenciador de Acesso, o Gerenciador de Requisições, o Gerenciador de Execução e o Gerenciador do Nó. Os componentes da arquitetura e o fluxo de informações, representados respectivamente pelos retângulos e pelas setas, podem ser vistos na figura 1.

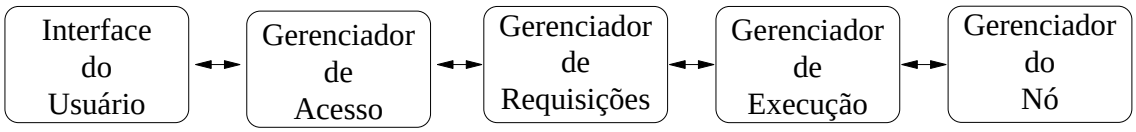

Figura 1 – Arquitetura do Sistema de Gerência de Agregados Crono

A Interface do Usuário (IU) consiste de um meio para o usuário interagir com o sistema através de uma interface gráfica ou utilizando o próprio ambiente *shell* da hospedeira (*bash*, *tcsh*, etc.). Através deste componente são disponibilizadas ferramentas que possibilitam aos usuários e administrador do sistema executar diversas tarefas, entre elas tem-se: mostrar informações relativas a fila de requisições, submeter *jobs* para execução, liberar os recursos, alterar o tempo e quantidade de nós das requisições, configurar o ambiente de execução e obter informações sobre direitos de acesso, nome dos agregados e nós que o usuário pode requisitar.

O Gerenciador de Acesso (GA) é responsável por autenticar usuários e verificar os direitos de acesso sobre o sistema. Neste componente o administrador, através de arquivos de configuração, pode criar grupos de usuários e atribuir restrições de acesso a estes grupos ou usuários distintos. Essas restrições são definidas pelo tempo máximo e quantidade máxima de nós para alocação e reservas de nós. Além disso, pode-se fazer essas atribuições de forma diferenciada pelo período do dia e da semana e cada agregado gerenciado pode ter seu conjunto específico grupos de usuários de restrições de acesso.

O Gerenciador de Requisições (GR) do Crono tenta aproveitar os recursos disponíveis (tempo e nós) que seriam desperdiçados usando o conceito de "o primeiro a chegar, é o primeiro a ser atendido" (FIFO), mas sem prejudicar os usuários que estejam da fila de espera. Ou seja, se um usuário espera ser atendido num determinado horário previsto, ele será atendido no máximo naquele horário, podendo ser atendido antes. Outra característica deste componente é a possibilidade de alteração dinâmica de recursos, isto é, os usuários podem pedir mais tempo ou mais nós do agregado para suas requisições, tanto antes quanto durante o horário previsto para seu atendimento.

O Gerenciador de Execução (GE) é o último componente do sistema pelo qual as requisições trafegam antes de serem efetivamente atendidas. Este componente recebe as requisições do GR e faz o tratamento de falhas (problemas nos nós), prepara o ambiente de execução, submete os *jobs* para os nós ou libera o acesso para execução de programas (dependendo se for modo *batch jobs* ou modo interativo, respectivamente) e faz o pós-processamento das alocações.

O Gerenciador do Nó (GN) é responsável por fornecer e alterar informações dos nós do agregado, através de requisições feitas pelo gerenciadores de Requisições e Execução. A principal funcionalidade deste componente é não permitir que usuários façam acesso indevido aos nós sem estarem devidamente autorizados. No próximo item será descrito de forma detalhada o processo de controle de acesso aos nós.

4. Controle de Acesso aos Nós dos Agregados

Como visto no item anterior, para que não haja interferência na performance das aplicações de diferentes usuários, deve-se fazer a proteção dos nós, ou seja, garantir que os usuários façam

uso apenas dos recursos computacionais permitidos pelo Crono. Esse controle de acesso é feito alterando arquivos de configuração de acesso do sistema GNU/Linux [Ha e Nquyen 1999].

Quando é feito o processo de *login* em um nó, o arquivo *login.access* é verificado afim de permitir ou não o acesso do usuário. Para garantir que apenas usuários que estejam em seus períodos de acesso possam utilizar os recursos liberados, o Crono altera o *login.access* dos nós a cada vez que um usuário entra ou sai deste período.

Entretanto, alterar o arquivo *login.access* não é o suficiente para restringir esse acesso. Os usuários podem ainda utilizar o comando *rsh (Remote Shell)* para executar operações remotamente sobre os nós. Isso porque o servidor do *rsh* não faz todo processo de *login*, apenas a autenticação dos usuários. Dessa forma, o *login.access* é ignorado.

Para resolver esse problema, o Crono também altera o arquivo *hosts.equiv* dos nós. O servidor *rsh* analisa esse arquivo e, caso o usuário não esteja autorizado, o servidor *rsh* requisitará a senha, realizando então o processo de *login*. Assim, será analisado o arquivo *login.access*, que não permitirá que os usuários não autorizados tenham acesso.

Ainda existe um problema que é em relação ao arquivo *.rhosts*, pois o servidor *rsh* o analisa antes do *hosts.equiv*. Isto abre a possibilidade de o usuário alterar o seu próprio *.rhosts* e executar comandos remotamente sem que seja solicitada a sua senha. Isto é facilmente resolvido, bastando inicializar o servidor *rsh* com a opção de ignorar o arquivo *.rhosts* do usuário.

5. Conclusão

A utilização de máquinas agregadas vêm sendo uma arquitetura alternativa para suprir a demanda computacional na área de alto desempenho. Estas arquiteturas, compostas de várias estações de trabalho, rodam em muitos casos o sistema operacional GNU/Linux e, com isso, há uma forte tendência na utilização de software livre neste tipo de ambiente.

Sistemas de Gerenciamento de Agregados são muito importantes para a organização do uso dos recursos nesse tipo de arquitetura. Os desenvolvedores de SGA's encontram no software livre um ambiente ideal para trazer a tona as idéias e técnicas importantes ao desenvolvimento de ferramentas de gerenciamento. Isso se deve principalmente ao grau de conhecimento que se pode ter do sistema operacional GNU/Linux e também aos códigos abertos dos programas que fornecem o reaproveitamento de rotinas e idéias já exploradas.

Referências

Baker, Mark A., Fox, Geoffrey C. e Yau, Hon W. "*Cluster Computing Review*" Novembro de 1995.

Manika, Guilherme W. "*Revista do Linux: Supercomputador a preço de banana*", edição no. 2, Janeiro de 2000.

Lewis, Bil e Berg, Daniel J. "*Multithread Programming with Pthreads*" Editora Sun Microsystems Press, 1998.

Stevens, Richard W. "*Unix Networking Programming*", Editora Printice Hall, vol. 1, edição no. 2, 1998.

Ha, Bao e Nquyen, Tina "*Slackware Linux Unleashed*", Editora Sans, Dezembro de 1999.