Avaliação de Ferramentas de Gerenciamento de Clusters

Dalvan Jair Griebler[1], Vinicius Casali[1], Claudio Schepke[1,2]

[1]Curso Superior de Tecnologia em Redes de Computadores
Faculdade Três de Maio (SETREM)
Av. Santa Rosa, 2405 – 98.910-000 – Três de Maio – RS – Brazil

[2]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{dalvangriebler,vinicius.casali,schepke}@gmail.com

Resumo. *Este artigo tem por objetivo demonstrar e comparar ferramentas de gerenciamento de cluster, levando em consideração as suas funcionalidades e formas de operação. A importância do gerenciamento de clusters surge da necessidade de escalonar as tarefas para que as mesmas não entrem em conflitos. Cada vez mais surgem novas aplicações que exigem alto poder de processamento. Clusters vêm sendo uma ótima alternativa para suprir essa necessidade, sendo constituídos por um grande número de nós. Neste sentido, este artigo apresenta conceitos de cluster e de gerenciamento de clusters e a descrição dos recursos de algumas ferramentas de gerenciamento, resultando em uma ampla comparação das mesmas.*

1. Introdução

A utilização de *clusters* vem sendo uma alternativa para a obtenção de alto desempenho e disponibilidade (ANDREWS, 2001). Diversos centros de processamento e armazenamento de dados fazem uso desta arquitetura, uma vez que sua implantação fornece recursos de segurança, alta disponibilidade e tolerância a falhas, além de muitos recursos computacionais à disposição. *Clusters* oferecem recursos de processamento que permitem a execução de tarefas ou processos em um tempo de execução muito menor, uma vez que os mesmos são distribuídos entre cada nó do *Cluster*.

Em termos de gerenciamento, *clusters* apresentam certa desvantagem em relação a máquinas individuais. Um *cluster* geralmente é formado por um grande número de microcomputadores e processadores para ser gerenciado, fato que exige a presença de escalonadores e gerenciadores qualificados para que os recursos possam ser gerenciados corretamente e não haja falhas na execução de tarefas.

Como o gerenciamento é um fator muito importante em um *cluster*, buscou-se através deste artigo, um meio de comparar algumas ferramentas de gerenciamento atualmente utilizadas. Para tanto, foram considerados aspectos referentes à instalação, configuração e uso dos gerenciadores.

2. Cluster

Cluster é um conjunto de computadores ligados em rede que compartilham recursos de processamento, armazenamento e memória, visando a criação de super-computadores

virtuais, para a execução de tarefas que exigem alto processamento (BUYYA, 2000). Um *cluster* também é utilizado como método de alta-disponibilidade, segurança e confiabilidade dos dados. Na Figura 1 é possível identificar um diagrama de um cluster em funcionamento.

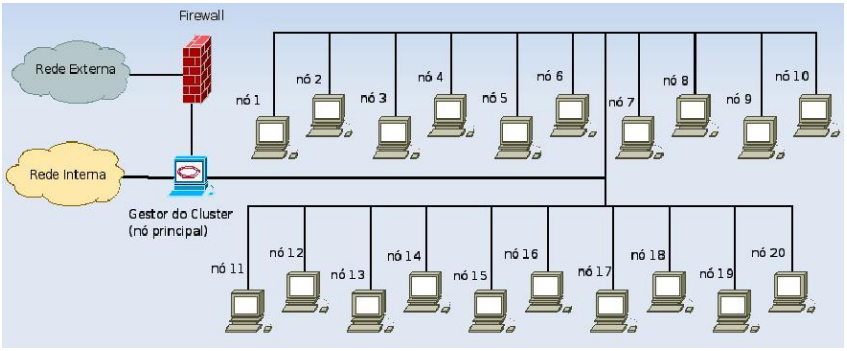

Figura 1: Diagrama de Funcionamento de um *Cluster*

De acordo com o diagrama, um cluster de microcomputadores é constituído por nós (Computadores), nó principal (Gestor do cluster), rede interna e rede externa. Cabe aos gerenciadores administrar todos estes nós, distribuindo os processos entre eles. Assim, recursos de processamento e memória, por exemplo, são corretamente gerenciados.

3. Gerenciamento de Cluster

Gerenciamento de cluster é uma forma sistemática de administrar uma rede de computadores. Mecanismos de gerenciamento disponibilizam, por exemplo, recursos de detecção de falhas e monitoramento da rede. Diferentemente de uma rede estruturada tradicional, c*lusters* trabalham com o conceito de processamento distribuído, fato que necessita um controle maior dos nós que estão em operação, uma vez que a interdependência entre os nós é maior. Através do gerenciamento é possível manter uma alta disponibilidade para a execução de processos (SCHEPKE, 2005).

Para o gerenciamento de *clusters* utilizam-se recursos de software. Estes softwares possuem recursos avançados, com a geração de gráficos para a melhor análise e configuração. Além do monitoramento, uma ferramenta de gerenciamento também é capaz de distribuir tarefas de acordo com as necessidades de processamento, memória ou até mesmo de tempo. Este papel é exercido por escalonadores, que são os responsáveis por gerenciar as tarefas que irão ser executadas pelos nós. Estes escalonadores determinam a prioridade de cada tarefa que entra na fila de execução, o tempo dedicado para cada tarefa em um determinado processador, entre outras funcionalidades.

Os escalonadores de *cluster* trabalham de forma semelhante aos escalonadores de um sistema operacional. Escalonadores de *cluster* são componentes de software, geralmente integrados em sistemas operacionais paralelos ou distribuídos. Seu papel é distribuir o trabalho computacional entres as unidades de processamento integrantes de um sistema, maximizando o desempenho global do processamento que está sendo realizado. Com isso, provê-se o balanceamento de carga entre as unidades de

processamento envolvidas em um contexto de gerenciamento. (MEFFE, MUSSI, MELLO, 2006).

Em ambientes de *cluster* existem dois tipos de escalonadores, os escalonadores estáticos e os dinâmicos:

- **Escalonadores Estáticos**. Neste tipo de escalonador o processamento é feito independente da distribuição da tarefa. Ele é executado em duas partes. Na primeira etapa é feito o cálculo do escalonamento a ser realizado, isso diz respeito à atribuição das tarefas para unidade de processamento. Já na segunda etapa é acionado um mecanismo de distribuição das tarefas que deve entrar em ação para a distribuição calculada. (MEFFE, MUSSI, MELLO, 2006).
- **Escalonadores Dinâmicos**. Neste tipo de escalonadores, a distribuição dos processos aos processadores é feita durante a execução do programa. Este paradigma trabalha com o conceito de balanceamento de carga. (MEFFE, MUSSI, MELLO, 2006).

Em geral, os escalonadores de *cluster* trabalham basicamente com três políticas de escalonamento:

- **Política da Informação**. Esta política especifica quais são as informações que deverão ser repassadas a carga dos Elementos de Processamento (EP), informando com que freqüência, as informações deverão ser atualizadas e para quem elas deverão ser enviadas.
- **Política de Transferência**. Especifica e determina sob quais condições os processos devem ser transferidos.
- **Política de Colocação.** Especifica e identifica os EP, para onde eles devem ser transferidos.

4. Ferramentas de Gerenciamento

Ferramentas de gerenciamento são softwares responsáveis por gerenciar e escalonar as aplicações de um *cluster*. A seguir são descritas algumas ferramentas de gerenciamento.

4.1 OpenPBS

OpenPBS é um sistema determinado a prever controle sobre a inicialização ou seqüenciamento de execução de grupos de tarefas. Ele permite a distribuição destes trabalhos entre vários nós de um *cluster*. OpenPBS possui um *Batch Server* que é um sistema de controle de tarefas (OpenPBS, 2008). Ele possui regras implementadas para diferentes tipos de recursos e quantidade de recursos, que podem ser usados por diferentes tarefas. Assim, possui-se um mecanismo pelo qual o usuário pode assegurar que uma tarefa tenha os recursos necessários para ser completada.

No sistema de gerenciamento OpenPBS é necessário um conjunto de componentes, como um servidor para gerenciar diferentes objetos, tarefas e filas de execução e interações típicas entre os componentes baseados em um modelo cliente-servidor. Um cliente faz a requisição para o servidor, para a execução de uma tarefa, e o servidor executa o trabalho em um de seus clientes (SCHEPKE, 2005).

O servidor de tarefas gerencia os objetos e trabalhos a serem executados como, por exemplo, as filas e as tarefas. Ele provê serviços de criação, execução, modificação, exclusão e roteamento de tarefas para os clientes (nós computacionais) responsáveis pela execução das mesmas (OPENPBS, 2008).

4.2 Torque

Torque é um gerenciador de recursos de código aberto, baseado no projeto PBS. Ele possui uma grande quantidade de recursos. As principais características são:

- **Tolerância a falhas**. Verificação de nós indisponíveis e suporte a diferentes condições de checagem de falhas.
- **Interface de seqüenciamento**. Interface de busca estendida, que provê informações mais apuradas sobre o escalonamento das tarefas. A interface permite maior controle sobre as tarefas, seus atributos e execução, possibilitando a obtenção dos dados resultantes das tarefas que foram executadas (TORQUE, 2008).
- **Escalabilidade**. Servidor de monitoramento, que têm a capacidade de trabalhar em um *cluster* significativo em termos de número de nós (acima de 15 TeraFlops e 2500 processadores), com um grande volume de tarefas (acima de 2000 processos) e suporte a um grande número e tamanho de mensagens oriundas do servidor. (TORQUE, 2008).
- **Usabilidade**. Mecanismo de *log* mais completo e *logs* com características de leitura mais simples.

4.3 Maui

Maui é um dos mais avançados escalonadores de tarefa. Ele usa políticas de escalonamento agressivas para melhorar a utilização dos recursos e minimizar o tempo de resposta das tarefas. Esta ferramenta permite a checagem de ambientes em tempo de produção, para saber se todas as alterações de configuração foram feitas de maneira correta. (MAUI, 2008).

Maui provê um controle de administração de recursos e volumes de tarefas que estão sendo executadas. Além disso, permite uma boa flexibilidade de configuração entre as suas diversas áreas, como por exemplo, em priorização de recursos, seqüenciamento, alocação, distribuição de cargas e políticas de reserva de recursos. (MEFFE, MUSSI, MELLO, 2006).

Atualmente, ele está sendo usado para gerenciar *clusters* com PlayStation. Exemplo disto é o laboratório NCSA, que tem um *cluster* capaz de executar 51 bilhões de operações matemáticas por segundo.

Devido à sua ótima eficiência e suporte a outras plataformas, o *Maui* está sendo utilizado no gerenciamento de *cluster* com bastante freqüência (MAUI, 2008).

4.4 Kerrighed

Kerrighed é conhecido como *Single System Image Operating System* (SSI OS) ou sistema de imagem única, destinado a trabalhar com computadores pessoais

(KERRIGHED, 2008). O objetivo de *Kerrighed* é alto desempenho de aplicações, alta disponibilidade do *cluster*, eficiência na administração de recursos, alto poder de customização do sistema operacional e facilidade de uso. Esta ferramenta possui as seguintes qualidades:

- Escalonador customizável para o *cluster*: Os processos e *threads* são automaticamente escalonados através dos nós do *cluster* para balancear o uso de CPU, fornecendo escalonamento sob encomenda. O sistema de escalonamento pode ser adicionado aos módulos do kernel sem reinicialização do mesmo. (KERRIGHED, 2008).
- Memória Compartilhada: *Threads* e segmentos de memória do sistema podem ser operados através do *cluster*, como em uma máquina SMP (*Symmetric Multi-Processors*).
- Mecanismos de migração de fluxo de alta performance: Pode-se migrar processos que usam fluxos (*socket, pipe, fifo, char device*, etc) sem perder desempenho de comunicação depois da migração.
- Sistema de arquivo distribuído: Um único espaço global de endereçamento de arquivos é visto no *cluster*. Todos os discos do *cluster* são fundidos em um único disco virtual, em uma customização parecida como um RAID.
- Verificação de processos: A partir de qualquer nó do *cluster* é possível verificar e reiniciar os processos.
- Interface de *Thread Posix* completa.
- Interface de processos Unix visível em todo o *cluster*.
- Características customizadas da imagem única de sistema: As SSI de memória compartilhada, escalonador global e migração de fluxos podem ser ativados ou não por base de processos.

4.5 SCMS

Scalable Cluster Management System (SCMS) é uma ferramenta interativa e extensível de gerenciamento de *cluster*. O objetivo de SCMS é permitir aos utilizadores executar a tarefa administrativa de maneira simples (WANDARTI, 2006). SCMS possui uma enorme quantidade de comandos, além de fornecer ferramentas de acompanhamento em tempo real do subsistema e interface web.

Com SCMS, o sistema de administração de tarefas de grande porte torna-se muito mais simples. As principais funcionalidades desta ferramenta são os comandos Unix paralelos, o alarme, o serviço de nomes, o serviço de eventos distribuídos e a monitoração.

O sistema de monitoração em tempo-real portável consiste de um *daemon* chamado CMA (*Control and Monitoring Agent*) que executa em cada nó e coleta estatísticas do sistema continuamente. Estas estatísticas são reportadas para um servidor de gerência de recursos central chamado SMA (*System Management Agent*) (SCMS, 2008). As informações são repassadas pelo agente para o gerente utilizando mensagens com protocolo UDP.

O serviço de alarmes consiste em um processo *daemon* chamado *alarm manager*. Em cada nó são criados *daemons* chamados *detector*. O relatório pode ser enviado por e-mail, através da execução de um comando Unix, caso a condição do alarme seja detectada. (WANDARTI, 2006).

4.6 Open Mosix

Open Mosix é um sistema que trabalha com memória compartilhada. Open Mosix facilita a troca de dados, principalmente em um *cluster* de banco de dados.

Open Mosix possui um *patch* DSM (*Distributed Shared Memory*) chamado *Migshm*. Ele permite a migração de processos que utilizam memória compartilhada no Open Mosix, tais como, apache, banco de dados, entre outros. (YOKOKURA, 2005).

Open Mosix trabalha com *checkpointing*, técnica que provê a possibilidade de salvar contextos de processos em um arquivo de disco e dar um *restore* (recuperação e continuação) dos processos a partir do arquivo (YOKOKURA, 2005). Logo, processos que tenham sofrido *checkpointing* e que sejam reiniciados posteriormente deveriam funcionar como se não tivessem sido interrompidos. Essa funcionalidade é útil para tarefas que possuem longo período de execução, como as simulações numéricas. No caso de instabilidade de sistema, falhas de energia ou reinicializações do sistema é possível continuar executando a partir do último ponto de *checkpoint*. (COSTA, 2004).

Open Mosix é um conjunto de algoritmos para compartilhamento dinâmico de recursos. Estes algoritmos são utilizados para fornecer escalabilidade e performance em um *cluster* de qualquer tamanho, onde o único componente compartilhado é a rede. A idéia principal da tecnologia Open Mosix é a capacidade de múltiplos nós que trabalham em cooperação como parte de um sistema único (YOKOKURA, 2005).

Open Mosix é também considerado um conjunto de algoritmos que suportam compartilhamento de recursos escaláveis pela migração de processos, podendo tornar as plataformas do tipo *Cache Coherent* mais próximas dos ambientes SMP (OPEN MOSIX, 2008).

5. Comparação Entre as Ferramentas de Gerenciamento

Para melhor compreender as características das ferramentas abordadas neste artigo, criou-se uma tabela comparativa apresentada na Tabela 1. Nesta tabela, as colunas identificam as ferramentas abordadas. Já na horizontal são apresentadas as características avaliadas de cada ferramenta, tais como Suporte ao SO, Facilidade de Instalação, Forma de Operação, Disponibilidade de Recursos, Limitações, Arquitetura, Escalabilidade e Desempenho.

De acordo com as informações apresentadas na Tabela 1, existem semelhanças entre as ferramentas quanto ao suporte ao sistema operacional e a forma de operação. Quanto a facilidade de instalação, OpenMosix caracterizou-se diferentemente das outras ferramentas, por estar embutido diretamente em um SO inicializável por CD. Sendo assim não necessita ser instalado no disco rígido. Quanto à disponibilidade de recursos, todas elas são ferramentas de escalonamento, variando características apenas em relação ao monitoramento e gerenciamento. Quanto às limitações, todas as ferramentas são usadas para *Cluster*, enquanto algumas podem ser usadas também em *Grids*.

Ferramentas	Open Mosix	OpenPBS	SCMS	Maui	Kerrighed	Torque
Suporte SO	Distribuição Linux	Distribuição Linux/UNIX	Distribuição Linux/Windows	Distribuição Linux/UNIX	Distribuição Linux	Distribuição Linux/UNIX
Facilidade de Instalação	Embutido no Kernel (Fontes)	Fontes	RPM/Fontes	Fontes	Fontes	Fontes
Forma de Operação	Interface Gráfica	Interface Gráfica	Interface Gráfica	Interface Gráfica	Interface Gráfica	Interface Gráfica
Disponibilidade de Recursos	Monitoramento /Gerenciamento /Escalonamento	Escalonador de Tarefas	Gerenciamento /Monitoramento	Escalonador de Tarefas	Escalonador Customizável /Gerenciador	Escalonamento de Recursos
Limitações	Cluster/Grid	Cluster/Grid	Cluster	Cluster/Grid	Cluster/Grid	Cluster/Grid
Arquitetura	Distribuída	Distribuida/ Modular	Centralizada	Distribuida/ Modular Extensível	Distribuida/ Modular	Distribuida/ Modular
Escalabilidade	----------	10.000 CPU	----------	603 CPU	256 CPU	2500 CPU
Desempenho	----------	500.000 Jobs Por Dia	----------	256 Gigaflops	----------	15 Teraflops

Tabela 1: Comparação de Ferramentas de Gerenciamento de C*luster*.

Em termos de arquitetura, somente o SCMS possui arquitetura centralizada. As demais ferramentas são todas elas distribuídas, podendo ser modular ou modular extensível. Em questões de desempenho e escalabilidade, as ferramentas Torque e OpenPBS tem um desempenho mais significativo. Já em OpenMosix, SCMS e Kerrighed, não foi possível obter essa informação.

6. Conclusão

As ferramentas de gerenciamento de *cluster* têm um papel fundamental na correta distribuição dos recursos de um *cluster* de computadores. Estas ferramentas garantem que as tarefas destinadas a um *cluster* sejam executadas em um determinado tempo, escalonando de acordo com a necessidade de cada uma delas. Graças a estas ferramentas é possível a criação de supercomputadores, com alto poder de processamento, garantindo um sistema tolerante a falhas e com alta disponibilidade.

É importante observar que muitas destas ferramentas foram implementadas em *Open Source,* tendo um papel importante para a iniciação cientifica. Com isso, é possível a avaliação prática, sem a necessidade da compra de software.

Também se observou que o ato de escalonar tarefas faz parte do gerenciamento de um cluster. Para que as tarefas sejam escalonadas, as mesmas necessitam de gerenciamento. Esta é uma das funções mais complexas e com grande responsabilidade no cluster, pois o escalonador é fundamental para a unidade de processamento distribuído, para que ele consiga suportar as falhas e trabalhar com o máximo de desempenho e qualidade.

Referências

ANDREWS, G. R. **Foundations of Multithreaded, Parallel, and Distributed Programming**. USA: Addison-Wesley, 2001.

BUYYA, R. PARMON: A portable and scalable monitoring system for cluster. **Software and Experience.** jun., 2000.

COSTA M. R. **Proposta de solução de Grandes quantidades de dados com uso aglomerado de computadores por meio da hibridação de cluster de alto desempenho com balanceamento de carga usando Linux**. Instituto Superior Tupy, Bacharelado em SI (Conclusão de Curso) Joinvile, 2004.

KERRIGHED. Disponível em: <http://www.kerrighed.org/wiki/index.php/Main_Page. Julho 2008.

MEFFE, C., MUSSI, E. O. de P., MELLO, L. R. de. **Guia de Estruturação e Administração do Ambiente de Cluster e Grid**. (Guia Cluster) 2006.

MAUI. Disponível em: <http://www.clusterresources.com/. Acesso em Julho de 2008.

OPEN MOSIX. Disponível em: <http://openmosix.sourceforge.net/. Acesso em Junho de 2008.

OPEN PBS. Disponível em: <http://www.openpbs.org/. Acesso em Julho de 2008.

SCHEPKE, C., DIVERIO, T. A., NEVES, M. V., CHARÃO, A. S. **Panorama de Ferramentas para Gerenciamento de Cluster**. Instituto de Informática, UFRGS. Diverio Programa de Pós-Graduação em Computação (artigo). Porto Alegre 2005, RS.

SCMS. Disponível em: <http://www.opensce.org/components/SCMS. Acesso em Junho de 2008.

TORQUE. Disponível em: <http://www.clusterresources.com/pages/products/torque-resource-manager.php. Acesso em Julho de 2008.

YOKOKURA A. Y. **Estudo de Viabilidade da Implantação de Técnicas de *Cluster* ao Projeto Servidor de Estações de Trabalho (SET).** Universidade do Pará, Centro de Ciências Exatas e Naturais (Conclusão de Curso) Belém, 2005.

WANDARTI, D. F. **Proposta de um Framework para Gerência de Clusters**. (Dissertação de Mestrado). Pontifícia Universidade Católica do Paraná, Paraná, 2003.