

# Comparação de ferramentas para gerenciamento de clusters\*

Claudio Schepke, Tiarajú A. Diverio  
Programa de Pós-Graduação em Computação  
Instituto de Informática, UFRGS  
{cschepke,diverio}@inf.ufrgs.br

Marcelo V. Neves, Andrea S. Charão  
Laboratório de Sistemas de Computação  
Curso de Ciência da Computação, UFSM  
{veiga, andrea}@inf.ufsm.br

## Resumo

*O uso de clusters tem sido uma das alternativas mais adotadas para o desenvolvimento de sistemas computacionais que visam o alto desempenho. A configuração, manutenção e uso deste tipo de arquitetura envolve diversos fatores, sendo simplificada pela existência de ferramentas poderosas que possibilitam a simplificação da administração dos recursos. Neste trabalho são comparadas algumas ferramentas de configuração e instalação do sistema e de programas que possibilitam o gerenciamento e monitoramento de todo um cluster. Assim, busca-se descrever as diferentes alternativas existentes para cada caso, ressaltando-se as principais funcionalidades dum sistema de gerenciamento.*

## 1. Introdução

*Clusters* [31] tem sido comumente utilizados em aplicações de simulação, biotecnologia, petroquímica, modelagem de mercados financeiros, mineração de dados, processamento de imagens e servidores de música e jogos para a *Internet*. Um dos motivos para a adoção de *clusters* como alternativa para o alto desempenho se deve ao fato da arquitetura ser facilmente escalável, podendo agregar capacidades de processamento baseados num grande número de computadores. No entanto, a falta de uma centralização administrativa desses recursos é um dos grandes desafios da área.

No contexto de *clusters* também existe uma falta de consenso em definir o que é gerenciamento. Algumas linhas de pesquisa acreditam que gerenciamento é apenas o escalonamento de processos nos computadores [3]; outros defendem o gerenciamento como a forma em que são mantidos os recursos computacionais através da configuração de ferramentas e bibliotecas [2]; enquanto alguns utilizam ainda a definição de gerenciamento para as tarefas administrativas, como a manutenção de usuários, grupos e permissões

e fatores relacionados a segurança. O conceito de gerenciamento de *clusters* adotado neste trabalho envolve diversos fatores, desde a instalação do sistema operacional até a definição de recursos para a programação paralela.

Este artigo tem por objetivo definir as principais funcionalidades relacionadas ao gerenciamento em *clusters*, bem como comparar algumas ferramentas que permitem a realização dessa tarefa. A próxima seção descreve os conceitos e técnicas envolvidos em cada uma das partes do gerenciamento considerados neste trabalho. Na seqüência são apresentadas algumas ferramentas utilizadas na instalação do sistema operacional, na configuração, no escalonamento de tarefas, na monitoração de *clusters*, além de mecanismos que possibilitam a integração desses recursos. A quarta seção busca destacar as principais diferenças entre as ferramentas, definindo um perfil do que é esperado numa ferramenta de gerenciamento. Por fim são apresentadas as conclusões obtidas com a realização deste trabalho.

## 2. Gerenciamento de Cluster

Gerenciamento de *cluster* pode ser considerado qualquer tarefa que disponibilize um bom aproveitamento dos recursos de um *cluster* [2]. Desta forma, a idéia é prover mecanismos que auxiliem a administração dos nós de forma integrada. Analisando-se funcionalidades disponibilizadas pelas ferramentas existentes, pode-se dividir o gerenciamento em três partes: instalação/configuração, escalonamento e monitoramento. A seguir serão vistas algumas características de cada uma dessas áreas.

### 2.1. Instalação e Configuração

A instalação/configuração de um *cluster* envolve a instalação do sistema operacional e de outros componentes de *software* em cada um dos nós e a administração e atualização desses recursos. Devido a esta tarefa envolver diretamente cada um dos nós, as ferramentas apresentam dificuldades em disponibilizar soluções genéricas para uma diversidade de sistemas operacionais e demais tipos de progra-

---

\* Este trabalho é fomentado pelo CNPq.

mas. Assim, é comum encontrar sistemas para a instalação e configuração de *clusters* ligados diretamente a uma versão ou distribuição do sistema operacional.

Para a instalação do sistema existem basicamente duas estratégias. A primeira faz uso de uma lista que contém o nome dos pacotes a serem instalados, os quais podem ser obtidos pelos nós através de um servidor de arquivos remoto. A segunda utiliza um sistema de imagens - instalações completas do sistema - que são copiados de um servidor usando, por exemplo, os protocolos de transferência PXE e TFTP no momento em que o computador é inicializado. Após a inicialização, os nós podem usar um sistema de arquivos distribuídos (NFS) ou outro protocolo de rede para a obtenção de configurações do sistema e de outros componentes de *software*.

A atualização e configuração de *software* em *clusters* pode ser feita através de recursos que disponibilizam a execução paralela de comandos, sendo estes muitas vezes realizados de forma remota. Existem também opções que mascaram e automatizam o processo de distribuição das atualizações, podendo estas estarem relacionadas às ferramentas de escalonamento e monitoração.

## 2.2. Escalonamento

*Clusters* são utilizados em aplicações que usam vários processadores para resolver um determinado problema. A maneira como estes são distribuídos e utilizados pelos usuários, mediante uma prévia requisição, vai depender do tipo de escalonamento de tarefas adotado. Os objetivos de um escalonador são maximizar a utilização do *cluster* e da quantidade de aplicações executadas, reduzir o tempo de resposta, mesclar requisições dos usuários com ordens administrativas e dar a ilusão de uma máquina única e dedicada [13]. Alguns desses objetivos são contraditórios entre si como, por exemplo, executar diferentes tarefas simultaneamente para maximizar o uso dos nós e diminuir o tempo de execução total de uma aplicação. Cabe ao administrador escolher a melhor política de escalonamento.

As políticas de escalonamento de um *cluster* podem estar baseadas em diversos fatores [1]. Uma delas diz respeito a utilização dos nós em relação ao tempo e ao espaço, que pode ser classificado em sistema de planejamento ou agendamento e sistema de enfileiramento. Num sistema de agendamento, as reservas de nós são delimitadas por tempo, não sendo possível alterar o número de nós alocados durante a execução [13]. Quando uma tarefa ultrapassar o limite de tempo pré-estabelecido acontece o cancelamento da execução. Já num sistema de filas é possível compartilhar a execução dos processos em relação ao tempo e ao espaço [1]. No compartilhamento de tempo os processos são executados em vários nós usando a mesma estratégia que é adotada em computadores com um único processador e vários

processos, enquanto que no compartilhamento por espaço a aplicação ou o usuário fornece ao sistema escalonador a quantidade de nós que irá ser utilizada.

## 2.3. Monitoramento

Monitoramento consiste na coleta e análise do estado do *cluster* em um dado momento ou em momento passados, através de um histórico [30, 20]. A análise das informações monitoradas é importante porque pode permitir a descoberta de possíveis falhas ou gargalos no *cluster*, além de facilitar a descobertas de erros de lógica ou desempenho nos programas em execução. Para isso são coletadas informações relevantes (métricas) sobre o estado do *cluster*. Essas métricas podem ser relativas ao desempenho dos módulos computacionais, como utilização de CPU, memória e rede, ou relativas aos processos em execução, como a trocas de mensagens, criação de *threads* ou mesmo a utilização dos recursos alocados à uma aplicação.

Em relação as métricas, estas podem ser coletadas periodicamente (cíclica), com armazenamento histórico dos dados, ou somente no momento em que os dados são necessários (por demanda). A forma de apresentação também pode variar; algumas ferramentas apresentam graficamente os dados de monitoramento, enquanto outras apenas disponibilizam os dados para algum cliente. Além desses pontos é importante avaliar fatores relacionados com a implementação. Uma implementação aberta e extensível permite a adição de novas métricas a serem coletadas e suporte a outros clientes, como uma aplicação de usuário que pode usar as métricas para tomada de decisão em um algoritmo de balanceamento de carga, por exemplo.

## 2.4. Integração de Recursos

Todas as partes do gerenciamento descritos anteriormente podem estar associadas a uma única ferramenta. A idéia está em construir um *middleware* no qual os componentes do *cluster* são vistos como um sistema único. Desta forma, a instalação, configuração, gerenciamento de tarefas e *software* relacionado as aplicações do usuário poderiam ser facilmente gerenciados. Neste sentido, as ferramentas podem fazer uso tanto de recursos já implementados, como também desenvolver novos componentes que sejam fortemente acoplados.

Os mecanismos de integração ou sistemas integrados apresentam diferentes estratégias e técnicas de funcionamento. A classificação/avaliação dessas ferramentas está relacionada com os recursos de instalação e configuração, as distribuições de sistema aplicáveis, as ferramentas de escalonamento e monitoração utilizadas, além de outros componentes.

### 3. Ferramentas

Esta seção apresenta alguns recursos para a instalação de sistemas operacionais, para a atualização e configuração do sistema, além de ferramentas para o escalonamento de tarefas, monitoração de recursos e integração de ferramentas. A seguir são descritas as principais características para cada um dos recursos, segundo as funcionalidades apresentadas na seção anterior.

#### 3.1. Instalação do Sistema Operacional

##### Instalação Automática

**Kickstart** [22] É um sistema desenvolvido para a distribuição *RedHat Linux*. Ele permite colocar todas as seleções que o usuário faz durante a instalação manual, como seleção da linguagem, partições e pacotes a serem instalados, em um arquivo de configuração. Assim é eliminado toda interação possível durante a instalação;

**Fully Automatic Installation - FAI** [11] É um conjunto de *scripts* e arquivos de configuração para a instalação do sistema *Debian Linux* em um *cluster* com um grande conjunto de nós. FAI usa um método escalável, onde cada nó realiza a sua própria instalação a partir de um arquivo de configuração existente num servidor. Para tanto, um nó cliente carrega um sistema temporário, via rede ou disquete, que serve para começar a instalação propriamente dita;

**Replicator** [5] É outro recurso desenvolvido exclusivamente para sistemas *Debian Linux*, funcionando como um duplicador de instalações;

**LUI** [8] A ferramenta foi desenvolvida pela IBM, sendo utilizada na instalação de *clusters* heterogêneos. LUI trabalha com o conceito de recursos. Tudo o que pode diferenciar um nó dos outros é considerado um recurso, como especificação de hardware, versão do *kernel*, pacotes de software e atributos de rede. Já as máquinas são consideradas objetos. Assim, na instalação do sistema operacional os recursos são alocados para os objetos, sendo possível alocar um conjunto de recursos diferente para cada objeto;

**ALICE** [6] É um sistema para distribuições *SuSE Linux* que permite instalar e configurar várias máquinas automaticamente com o mínimo possível de interação. ALICE é baseado em interfaces como *syslinuxrc*, *YaST* e *suseconfig*. Além de instalar o sistema operacional, o sistema também possibilita criar grupos e usuários, ativar serviços, etc;

**LCFG** [25] É um mecanismo de configuração e instalação automática que disponibiliza um repositório central de especificações de onde cada máquina pode ser automaticamente configurada e instalada. Modificações realizadas nessa central, onde os clientes são descritos através

de uma linguagem especial, são automaticamente refletidas nos nós. A instalação utilizando LCFG é semelhante ao FAI, uma vez que também são utilizados arquivos de configuração localizados no servidor.

##### Carga automática do sistema

**SystemImager** [9] É um conjunto de *scripts* que simplificam os procedimentos para a preparação da carga remota. Com ele também é possível atualizar as imagens já distribuídas aos nós. As atualizações são rápidas porque somente as partes modificadas são mandadas ao cliente. SystemImager também permite o armazenamento de várias imagens em um servidor, podendo estas serem associadas à nós específicos. Desde 2001, SystemImager e LUI foram integradas num único sistema conhecido como *System Installation Suite* (SIS), buscando ser uma solução completa em termos de instalação;

**Rembo Toolkit** [28] É um inicializador remoto desenvolvido a partir da ferramenta *BpBatch*. Rembo utiliza o protocolo PXE para a cópia das imagens. A ferramenta também permite a execução de várias ações antes do sistema operacional ser carregado. Assim, é possível particionar o disco rígido, autenticar usuários e criar uma imagem dum disco rígido clonando o estrutura de partições;

**Ka-deploy** [26] É uma ferramenta que faz parte do *Ka Clustering Tools*, que permite replicar uma máquina Linux em várias cópias ao mesmo tempo. A carga remota em *clusters* cria uma corrente de dados entre os nós: cada nó copia os dados para seu disco local e envia uma cópia para o restante das máquinas;

**ClusterWorx** [17] É outro exemplo de ferramenta para auxiliar o processo de carga remota. Ele foi desenvolvido pela empresa *Linux NetworX*, funcionando também como um gerenciador de imagens.

##### Atualização e Configuração do Sistema

**SHOC** [32] É um recurso que permite ao administrador, através de um *shell bash*, usar o agregado com se o mesmo fosse uma única máquina;

**ICE Box** [18] É o sistema gerenciador de *hardware* da *Linux NetworX*, que também permite o diagnóstico e correção de falhas;

**Scalable Cluster Management System - SCMS** [27] É um sistema desenvolvido pela Universidade Kasetsart (Tailândia) que possui recursos úteis para a configuração e atualização remota como, por exemplo, comandos UNIX lançados em paralelo, o que permite a execução da mesma tarefa em vários nós ao mesmo tempo e a instalação de pacotes RPMs em paralelo. SCMS, assim como outras ferramentas, possui outros recursos relacionados ao gerenciamento.

### 3.2. Escalonamento

**Computing Center Software - CCS** [15, 19] É uma ferramenta desenvolvida pelo Centro de Computação Paralela de Paderborn (Alemanha). O objetivo de CCS é gerenciar sistemas MPP (*Massively Parallel Computing*) e *clusters* usando um sistema de planejamento. Para isso, a ferramenta permite o acesso de recursos concorrentes de forma exclusiva, processamento simultâneo do modo interativo e de fila, maximização do uso do *cluster* através do particionamento dinâmico e do escalonamento, além de oferecer tolerância a falhas em acesso remotos.

A arquitetura de CCS é modular, o que permite integrar um grande número de sistemas. Cada um dos módulos tem uma funcionalidade específica interagindo com os demais. Quanto ao gerenciamento, CCS é considerado um sistema de planejamento, uma vez que é necessário alocar primeiramente os recursos a serem utilizados. Depois de ter feito a alocação das máquinas o usuário precisa aguardar o início do tempo destinado a ele para iniciar as execuções;

**Portable Batch System - PBS** [23] É uma ferramenta desenvolvida inicialmente pela NASA e posteriormente apresentado em uma versão comercial. Além da versão comercial existe também uma versão de código aberta conhecida como openPBS [19]. PBS possui suporte a tarefas tanto para um único sistema como para múltiplos sistemas. Devido a flexibilidade da ferramenta, os sistemas podem ser agrupados em diferentes formas.

PBS é composto por quatro partes principais: comandos, servidor de tarefas, executor de tarefas e o escalonador de tarefas. O módulo comandos permite o lançamento de tarefas em lote. O servidor de tarefas e o escalonador de tarefas são executados no *front-end* do cluster. Já o executor de tarefas executa em cada um dos nós as tarefas enviadas pelo escalonador. Um cliente requisita uma tarefa através do servidor de tarefas. Este, por sua vez, coloca a tarefa numa fila, a fim de que o escalonador defina a política de execução a ser seguida; Assim PBS é classificado como uma ferramenta de enfileiramento.

**Condor** [33] É uma ferramenta desenvolvida pela Universidade de Wisconsin-Madison nos Estados Unidos, que pode ser usada para gerenciar *clusters* e múltiplos *clusters*. Algumas das características de *Condor* são a submissão distribuída de tarefas, prioridades para usuários e tarefas, suporte a múltiplos modelos de tarefas, *checkpointing* e migração, suspensão de tarefa e posterior continuação, autenticação e autorização. Essas características, aliada a incorporação de Condor no ambiente *Globus* [10] para *Grid*, fazem dela uma ferramenta poderosa no gerenciamento de recursos computacionais distribuídos.

Condor utiliza um esquema de divulgação para combinar as aplicações aos recursos disponíveis. O sistema tem

um gerenciador central responsável pelo recebimento da divulgação dos processos que monitoram recursos e também das características solicitadas por um usuário para executar um programa. Essas características podem ser a quantidade de memória disponíveis em uma máquina, o tamanho do disco, a largura de banda da rede daquele computador, entre outras. Após as informações estarem no gerenciador central outro processo varre-as tentando combinar uma tarefa com a sua requisição de recursos. Como não é necessário pré-alocação, nem a descrição de quando um processo deve ser executado, Condor é classificado como um sistema de enfileiramento;

**Maui** [14] É um escalonador de tarefas configurável e otimizado usado em *clusters* e supercomputadores. A ferramenta é capaz de suportar diferentes técnicas de escalonamento com justiça, prioridades dinâmicas, reserva e compartilhamento de recursos. As técnicas de otimização adotadas permitem aumentar a utilização dos recursos e diminuir o tempo de resposta na execução de tarefas paralelas.

Maui é implementado em Java, o que permite a extensão e utilização da ferramenta em diversos ambientes. A ferramenta necessita apenas de uma máquina virtual (JVM) em execução para ser utilizada. A estrutura de Maui é composta por escalonador, objeto escalonável, tarefas e reservas. O escalonador é responsável por escalonar as tarefas, que representam os recursos alocados por um determinado tempo para uma tarefa. As tarefas também podem ser consideradas uma implementação de um objeto escalonável. Um objeto escalonável por sua vez é uma abstração de uma tarefa, tendo características como tempo de submissão para a execução, tempo de execução inicial, duração, geometria da tarefa, entre outras. Maui pode funcionar como um sistema de enfileiramento, quando o usuário simplesmente colocar a tarefa na fila ao requisitar uma execução, ou como um sistema de planejamento, quando é utilizado o sistema de reserva da ferramenta;

**Crono** [21] A ferramenta possui como objetivo principal o gerenciamento de *clusters* pequenos e médios. Ela foi desenvolvida na Pontifícia Universidade Católica (PUCRS), disponibilizando serviços necessários para compartilhar um *cluster* entre vários usuários. Crono é considerado um sistema de planejamento uma vez que a utilização do *cluster* ocorre por meio de agendamentos. A arquitetura da ferramenta é composta de quatro partes, sendo estas responsáveis por realizar a interface com o usuário, gerenciar o acesso (validação das requisições), gerenciar as requisições (escalonar pedidos e preparar o ambiente de execução) e gerenciar o nó.

Além destes existem ainda outras ferramentas de escalonamento como OAR, DQS (semelhante a outros gerenciadores de filas), Loadleveler (IBM), LSF e SGE (usados

em *grids*) e ferramentas derivadas de *softwares* já existentes como OpenPBS/Torque e PBSPro.

### 3.3. Monitoramento

**Ganglia** [20] É uma ferramenta utilizada em *clusters* e *grids* para a monitoração distribuída e escalável. A monitoração distribuída de *clusters* é feita de maneira que cada nó de monitoração (*gmond*) possua uma cópia do estado atual de todo *cluster*, garantindo tolerância a falhas, uma vez que é possível obter as mesmas informações de qualquer nó. Já um módulo centralizador (*gmetad*) solicita de tempos em tempos os dados dos nós monitorados de maneira que as informações lidas do *cluster* sejam atualizadas. Com *Ganglia* é possível monitorar qualquer tipo de informação, uma vez que o usuário pode definir métricas específicas através de outra aplicação, além daquelas já coletadas pelo próprio sistema.

Para o armazenamento *Ganglia* (*gmetad*) utiliza o sistema *RRDtool* (*Round Robin Database*), um sistema que permite armazenar de forma compacta seqüências temporais de dados em um banco de dados circular. Todos os dados coletados pelo *RRDtool* podem ser visualizados graficamente através de uma interface *Web*. Além disso, *Ganglia* também possui uma biblioteca (*libganglia*) que auxilia na criação de clientes, facilitando a sua adaptação às necessidades do administrador do *cluster*. Quanto ao gerenciamento, basta simplesmente executar *gmond* em uma máquina para adicionar um nó ao *cluster* monitorado;

**Parmon** [4] É uma ferramenta comercial que possui uma arquitetura centralizada, sendo dividida em duas partes: servidor e cliente. O servidor é responsável por monitorar o nó e já vem compilado para a arquitetura da máquina, enquanto que o cliente, por ser implementado em Java, pode executar em qualquer computador, independente de arquitetura, desde que pertença a mesma rede. Parmon permite adquirir informações dos recursos do sistema de vários nós, acompanhar processos e *logs* do sistema, além de definir eventos de alerta (*trigger*) ao administrador do *cluster*. Também é possível monitorar CPU, memória, rede e disco e executar alguns comandos paralelos. A centralização de todos os dados monitorados ocorre no cliente, o qual converte os dados e os mostra de forma gráfica e *on-line*, podendo também apresentar algumas informações de configuração do sistema de forma textual.

A monitoração ocorre por demanda cíclica, ou seja, no momento em que o cliente faz uma requisição ele envia a métrica a ser monitorada e o tamanho do período, o qual pode ser ajustado. O servidor então começa a monitorar a métrica até que o cliente deixe de requerer a métrica, apresentando-as graficamente sob diversas formas. Parmon não é extensível, pois o servidor atende somente a requisições do cliente, não sendo possível a utilização de seus da-

dos por outro programa. Além disso, Parmon não permite monitorar outras métricas. Para a instalação de Parmon é necessário a distribuição de GNU/Linux RedHat 8.0, ou superior, ou Solaris e Java 1.4.2, para rodar o cliente. Já o servidor pode executar em qualquer distribuição de GNU/Linux ou Solaris;

**SCMS** [27] A ferramenta tem como objetivo monitorar de forma simples, eficiente e robusta *clusters* de pequeno e médio porte através de uma arquitetura centralizada organizada num módulo de monitoração e num módulo de centralização, o qual armazena os dados monitorados num repositório próprio e atende as requisições dos clientes. A ferramenta permite monitorar o uso de CPU, memória, rede e disco, além de fornecer informações úteis sobre a configuração dos nós do *cluster*. A coleta de dados ocorre em ciclos ou por demanda, no caso das informações de configuração do sistema.

A apresentação gráfica dos dados monitorados por SCMS ocorre no cliente. O cliente pode selecionar as métricas e os nós que devem ser monitorados, desde que estes estejam listados num arquivo de configuração da ferramenta. SCMS disponibiliza uma API com rotinas em C, TCL/TK e Java, para permitir que outros programas utilizem os dados monitorados;

**RVision** [7] É uma ferramenta de monitoração desenvolvida com o objetivo de ser adaptável à diferentes *clusters*, tendo uma arquitetura aberta e configurável. Para manter essas características a ferramenta possui uma interface para a comunicação dos clientes com o núcleo da ferramenta.

A arquitetura de *RVision* é centralizada, sendo composta de um programa monitor e de um programa centralizador. Ao invés do monitor, é possível utilizar também um agente SNMP em seu lugar, sendo possível assim a adição de novas métricas. A fonte de dados pode ser selecionada através de um arquivo de configuração de seção, onde também pode-se selecionar o tipo de análise (*on-line* ou *post-mortem*), tipo de obtenção dos dados (demanda, cíclico ou por alteração, ou seja, o módulo monitor só envia quando uma condição for satisfeita), nós a serem monitorados e outras configurações. A forma de visualização é variável, dependendo do cliente implementado. Em caso de monitoração *post-mortem*, o núcleo é responsável por armazenar os dados em arquivos.

Outras ferramentas de monitoração não incluídas neste trabalho são Mirador, CIS (Cluster Information Service), SRMS (SMILE Resource Monitoring System), Co-Pilot (PCP), Ka-Admin, ClusterProbe e SIMONE.

### 3.4. Ferramentas de Integração

**NPACI ROCKS** [29] Atualmente é uma das ferramentas mais populares para a integração de recursos. ROCKS foi

desenvolvido para a distribuição *Red Hat*, possibilitando o gerenciamento da instalação (fontes, pacotes e controlador de *drivers*). do *software* e do estado do *cluster*. A instalação no *cluster* ocorre de forma automática através da ferramenta *Kickstart*, que realiza a propagação das instalações de forma escalar através do uso de uma lista específica de pacotes para cada *software*. Assim, é possível realizar a instalação de componentes heterogêneos. A atualização de *software* também é algo possível de ser feito. Já para o escalonamento é possível fazer a escolha de ferramentas como SGE, OpenPBS e Condor;

**Open Scalable Cluster Environment - (OpenSCE)** [24] É um projeto de código aberto que tem como objetivo criar um ambiente de cluster escalável e extensível. Assim é possível obter clusters de maneira fácil e eficiente, reduzindo a complexidade de manutenção de clusters. OpenSCE também prove um ambiente amigável que possibilita aumentar a produtividade dos usuários. Para tanto, o projeto busca integrar ferramentas que possam interajam entre si como SCMS, SCMSWeb, SQMS (Simple Queuing Management System), KSIX (Kasetsart System Interconnected eXecutive) e MPITH (MPI Thin and High-Performance).

**Open Source Cluster Application Resources (OSCAR)** [16] É um *middleware* que proporciona num ambiente integrado, os recursos mais utilizados em *cluster*. OSCAR disponibiliza a configuração automática de componentes, bem como a instalação eficiente do ambiente básico como o sistema operacional e ferramentas de administração e operação. Outros componentes relacionados a este *middleware* são o escalonamento de tarefas feito através de com OpenPBS, bibliotecas de comunicação como LAM/MPI, Mpich e PVM e o monitoramento com *Clumon* ou *Ganglia*. A versão corrente de OSCAR possui suporte para as distribuições *Linux Red Hat*, *Fedora* e *Mandriva*;

**Extreme Linux Cluster Administration Toolkit (xCAT)** [12] Desenvolvido pela IBM, é um sistema que automatiza alguns processos de instalação e configuração para a distribuição *RedHat*. Ele permite ligar e desligar as máquinas remotamente, acessar a BIOS através de um console e usar uma espécie de *shell* paralelo para executar o mesmo comando em vários nós. Além da instalação e configuração, a ferramenta possibilita o diagnóstico e correção de falhas de *hardware* e *software*. Assim é possível receber alertas do hardware por SNMP e acessar logs de software e hardware nos nós para descobrir alguma irregularidade. Quanto ao gerenciamento de tarefas, este é feito através de OpenPBS ou Maui. Já para o monitoramento é utilizada a ferramenta *Ganglia*. O sistema também possui de forma integrada implementações de MPI e PVM.

Outras ferramentas que buscam apresentar soluções semelhantes são: *Scyld Beowulf*, *Clustermatic*, *ClusterKnoppix*, *Warewulf* e *Score*

## 4. Análise e Comparação de Ferramentas

### 4.1. Instalação

A Tabela 4.1 apresenta um quadro comparativo entre as duas formas utilizadas pelas ferramentas de instalação de sistemas operacionais em *clusters*, destacando os pontos fortes e fracos de cada um dos modelos. Cabe ao administrador decidir, segundo as características apresentadas na tabela, a melhor forma de instalação a ser adotada para um determinado *cluster*, a fim de escolher uma ferramenta que atenda a esse propósito. No caso da instalação automática podem ser utilizadas *Kickstart* ou *FAI*, enquanto que a carga remota pode ser feita através de *SystemImager* ou *Ka-deploy*. Outra possibilidade é usar uma ferramenta que possui as duas modalidades de instalação como é o caso de *SIS*.

### 4.2. Escalonamento

O escalonamento de tarefas envolve o tipo de política de gerenciamento de recursos adotado. Neste trabalho foi levado em conta apenas o fator tempo e espaço de alocação através da análise de sistemas de agendamento e enfileiramento. A Tabela 4.2 apresenta um resumo das ferramentas de escalonamento abordadas neste trabalho, realizando uma comparação entre os sistemas computacionais aplicados, arquitetura da ferramenta e tipo de escalonamento disponível. Analisando-se os três fatores percebe-se que as ferramentas apresentam diferentes enfoques de desenvolvimento, cada qual buscando ser uma alternativa para sistemas com diferentes características.

### 4.3. Monitoramento

Uma comparação entre as ferramentas de monitoramento observadas mostra que existem características específicas em cada uma delas. A Tabela 4.3 avalia as ferramentas *Ganglia*, *Parmon*, *SCMS* e *RVision* em relação a algumas características para a monitoração. Através da tabela. percebe-se que *Ganglia* possui tolerância a falhas, uma vez que a arquitetura é distribuída, enquanto *RVision* possibilita a coleta de dados sob diversas formas. Em relação as outras características, existem pequenas diferenças entre as ferramentas, o que novamente ressalta a ênfase de projeto em cada uma delas.

### 4.4. Sistemas de integração

Sistemas de integração constituem-se de um *middleware* para *cluster*. A idéia de integrar diferentes mecanismos de gerenciamento pode ser visto na Tabela 4.4, onde são utilizadas diferentes ferramentas em cada um dos sistemas.

**Tabela 1** - Comparação entre ferramentas de instalação do sistema operacional

Característica	Instalação Automática	Carga Remota
Forma de instalação	Lista de pacotes	Imagem do sistema
Sistema operacional	Específico	Independente de sistema
Tipos de cluster	homogêneos e heterogêneos	homogêneos
Pontos positivos	Facilidade de atualização	Facilidade de instalação
Pontos negativos	Dependência de pacotes	Criação de novas imagens

**Tabela 2** - Comparação entre ferramentas de escalonamento de tarefas

Característica	CCS	PBS	Condor	Maui	Crono
Sistema	MPP e cluster	multi- e cluster	multi- e cluster	supercomp. e clusters	cluster pequenos e médios
Arquitetura	modular	modular	sistema central	modular e extensível	modular
Agendamento	planejamento	enfileiramento	enfileiramento	ambos	planejamento

Dependendo da funcionalidade é possível a utilização de mais de uma ferramenta, o que mostra a flexibilidade desses sistemas. Em alguns deles não são utilizados projetos de terceiros, obrigando o desenvolvimento de ferramentas próprias. Em relação a estes *middlewares*, é possível notar também uma preocupação com a disponibilização de recursos de programação paralelos.

## 5. Conclusão

A utilização de *clusters* tem sido uma das alternativas na busca por um maior poder de processamento, apresentando uma boa relação de custo-benefício. No entanto, uma das dificuldades para a utilização dessa arquitetura está no gerenciamento dos recursos. Para suprir essa deficiência, existem diversas ferramentas que buscam apresentar soluções específicas ou de forma integrada.

Neste trabalho foram apresentadas algumas ferramentas vinculadas a instalação, configuração, escalonamento de tarefas e monitoramento de recursos, buscando traçar um perfil para cada uma das áreas do gerenciamento, bem como contribuir com uma análise comparativa entre as ferramentas. Além dos recursos existentes para cada uma das funcionalidades, é possível a utilização de sistemas integrados. Neste sentido, alguns esforços já foram apresentados, como é o caso das ferramentas ROCKS, OpenSCE, OSCAR e xCAT. No entanto, todas elas possuem alguma deficiência ou são pouco flexíveis em termos de integração com outros recursos. Assim, em trabalhos futuros estas ferramentas poderão ser estendidas na busca por uma proposta que possa ser um padrão para o gerenciamento de *cluster*.

## Referências

- [1] S. V. Anastasiadis and K. C. Sevcik. Parallel application scheduling on networks of workstations. *Journal of Parallel and Distributed Computing*, 43(2):109–124, 1997.
- [2] M. Baker, G. Fox, and H. Yau. Cluster computing review, 1995.
- [3] M. Brune, A. Keller, and A. Reinefeld. Resource management for high-performance pc clusters. In *Proc. of 7th International Conference (HPCN Europe)*, volume 1953 of LNCS, Amsterdam, The Netherlands, April 1999. Springer.
- [4] R. Buyya. PARMON: A portable and scalable monitoring system for clusters. *Software Practice and Experience*, 30(7):723–739, jun 2000.
- [5] S. Chaumat. Replicator 2.0 for Debian/GNU Linux 2.2 Manual, 2000.
- [6] A. F. F. Herschel, P. Hollants. ALICE: Automatic Linux Installation and Configuration Environment, 2000. <http://www.suse.de/~fabian/alice/>.
- [7] T. C. Ferreto, C. A. F. de Rose, and L. de Rose. Rvision: An open and high configurable tool for cluster monitoring. *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02)*, page 75, 2002.
- [8] R. Ferri. Linux Utility for cluster Installation (LUI), 2003. <http://oss.software.ibm.com/developerworks/projects/lui/>.
- [9] B. Finley. Systemimager, 2003. <http://systemimager.org>.
- [10] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. *International Journal of Supercomputer Applications*, 11(2):115–128, 1997.
- [11] M. Gärtner, T. Lange, and J. Rühmkorf. The fully automatic installation of a Linux cluster, 1999.
- [12] M. Govindaraju, S. Krishnan, K. Chiu, A. Slominski, D. Gannon, and R. Bramley. XCAT 2.0 : A Component Based Programming Model for Grid Web Services. In *Grid 2002, 3rd International Workshop on Grid Computing*, 2002.
- [13] M. Hovestadt, O. Kao, A. Keller, and A. Streit. Scheduling in HPC Resource Management Systems: Queuing vs. Planning. In D. G. Feitelson and L. Rudolph, editor, *Proc. of the*

**Tabela 3 - Comparação entre ferramentas de monitoração**

Características	Ganglia	Parmon	SCMS	RVision
Arquitetura	distribuída	centralizada	centralizada	centralizada
Visualização	on-line e post-mortem	on-line	on-line e post-mortem	on-line e post-mortem
Análise	gráfica e textual	gráfica e textual	gráfica e textual	variável
Coleta de dados	cíclica	demanda cíclica	cíclica	cíclica, demanda ou alteração
Extensibilidade	possui	não possui	possui	possui

**Tabela 4 - Comparação entre ferramentas de integração**

Recursos	ROCKS	OpenSCE	OSCAR	xCat
Instalação	Kickstart	Possui	Possui	Possui
Configuração	Possui	SCMS	Possui	Possui
Escalonamento	SGE, OpenPBS e Condor	SQMS	OpenPBS	OpenPBS e Maui
Monitoramento	-	SCMS	Clumon e Ganglia	Ganglia
Programação	-	MPITH	LAM/MPI, MPICH e PVM	MPI e PVM
Distribuição	Red Hat	-	Red Hat, Fedora e Mandriva	Red Hat

- 9th Workshop on Job Scheduling Strategies for Parallel Processing, volume 2862 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2003.
- [14] D. Jackson, Q. Snell, and M. Clement. Core Algorithms of the Maui Scheduler. *Lecture Notes in Computer Science*, 2221:87–94, 2001.
- [15] A. Keller and A. Reinefeld. CCS Resource Management in Networked HPC Systems, 1998.
- [16] B. Li. OSCAR: Open Source Cluster Application Resources, 2005. <http://oscar.openclustergroup.org>.
- [17] Linux Networx. Clusterworx, 2005.
- [18] Linux Networx. Icebox Cluster Management Appliance, 2005. <http://linuxnetworx.com/icebox/>.
- [19] R. Magrin, A. Santos, R. Ávila, and P. Navaux. Gerenciamento de Agregados OpenPBS x CCS. *Escola Regional de Alto Desempenho (ERAD)*, 2003.
- [20] M.L. Massie and B.N. Chun and D.E. Culler. The Ganglia Distributed Monitoring System: Design, Implementation, and Experience. *Parallel Computing*, 30(7), July 2004.
- [21] M. A. S. Netto and C. A. F. D. Rose. Crono: a configurable management system for linux clusters. In *The Third LCI International Conference on Linux Clusters: The HPC Revolution*, 2002.
- [22] J. O’Kane. Kickstart. *Sys Admin: The Journal for UNIX Systems Administrators*, 9(1):33–34, 36, Jan. 2000.
- [23] OpenPBS.org. The Portable Batch System, 2003. <http://www.openpbs.org>.
- [24] OpenSCE Project. OpenSCE. Open Scalable Cluster Environment, May 2005. <http://www.opensce.org>.
- [25] Paul Anderson and Alastair Scobie. LCFG: The Next Generation, 2002. <http://www.lcfg.org/doc/ukuug2002.pdf>.
- [26] Philippe Augerat and Wilfrid Billot and Simon Derr and Cyrille Martin. A scalable file distribution and operating system installation toolkit for clusters, 2003. <http://ka-tools.sourceforge.net/publications/file-distribution.pdf>.
- [27] Putchong Uthayopas and Arnon Rungsawang. SCMS: An Extensible Cluster Management Tool for Beowulf Cluster. In *Proceedings of Supercomputing ’99 (CD-ROM)*, Portland, OR, nov 1999. ACM SIGARCH and IEEE. Department of Computer Engineering, Kasetsart University.
- [28] Rembo Technology. Rembo Toolkit, 2005. <http://www.rembo.com>.
- [29] Rocks Cluster Distribution. ROCKS, May 2005. <http://www.rocksclusters.org>.
- [30] C. Roder, T. Ludwig, and A. Bode. Flexible status measurement in heterogeneous environment, 1998.
- [31] S. Steiner. Building and installing a beowulf cluster. *J. Comput. Small Coll.*, 17(2):78–87, 2001.
- [32] C. M. Tan, C. P. Tan, and W. F. Wong. Shell over a cluster (SHOC): Towards achieving single system image via the shell, Sept. 30 2002.
- [33] C. Team. *Condor Version 6.2.2 Manual*. University of Wisconsin-Madison, Professor Miron Livny, 7367 Computer Science, 1210 West Dayton St, Madison, WI 53706-1685, 6.2.2 edition, 2001.