Monitoramento de Recursos em Ambientes de Grade

Daniel da Trindade Lemos [1], Patrícia Kayser Vargas Mangan [1]

[1]Curso de Ciência da Computação – Centro Universitário La Salle (UNILASALLE) Av. Victor Barreto, 2288, Centro, Canoas - RS – Brazil

daniel_tl@hotmail.com, kayser@unilasalle.edu.br,

Abstract. *In a grid computing environment, collecting and providing resources monitoring information is challenging due to heterogeneity and scalability issues. This information can be used to help discovery as well as task scheduling systems. Besides, it can aid management decision tools. After analyzing available monitoring tools, we can observe that many tools provide information concerning CPU and memory utilization, but none provides software information. This work aims at a monitoring model for hardware and software resources, which deals with scalability and heterogeneity issues, directed to grid scheduling systems.*

Resumo. *Em um ambiente de computação em grade, a coleta e o processamento de informações de monitoramento de recursos enfrenta desafios relacionados a grande heterogeneidade e questões de escalabilidade. Essas informações podem ser usadas tanto para auxiliar ferramentas de descoberta quanto de escalonamento de tarefas. Além disso, podem auxiliar ferramentas de tomada de decisão gerencial. Após a análise de ferramentas disponíveis, nota-se que muitas disponibilizam dados sobre utilização de CPU e memória, mas nenhuma disponibiliza informações de software. Este trabalho busca um modelo de monitoramento de recursos de hardware e software, que contemple as questões de escalabilidade e heterogeneidade, para sistemas de escalonamento de grade.*

1. Introdução

Nos últimos anos, houve um significativo aumento no uso de redes de computadores em ambientes comerciais e de pesquisa. Os computadores que normalmente compõem essas redes são heterogêneos e podem encontrar-se distribuídos através de prédios, cidades e até países diferentes. Esses equipamentos precisam estar conectados através de uma infra-estrutura de hardware e software que pode ser tanto um sistema distribuído ou uma grade computacional (*grid computing*).

Segundo Mattos [18] uma grade computacional é uma infra-estrutura de hardware e software que provê acesso seguro, consistente, de forma distribuída e a custo baixo.

Grade é um tipo de sistema distribuído e paralelo que permite compartilhar, selecionar e agregar recursos autônomos distribuídos geográfica e dinamicamente executando dependendo de sua disponibilidade, potencialidade, desempenho, custo, e exigências dos usuários para qualidade de serviço [1].

Foster *et al*. [2] define uma grade como um sistema que:

1) coordena recursos de diferentes fins (aplicações científicas, comerciais, desktop, etc.);

2) usa interfaces e protocolos padronizados, abertos e para propósitos gerais

3) oferece QoS (qualidade de serviço) não triviais: segurança, autenticação, escalonamento de tarefas (distribuição do processamento), disponibilidade, tempo de resposta.

Além disso, existem três principais aspectos que caracterizam grades computacionais [9]:

- *heterogeneidade (heterogeneity)*: uma grade envolve uma multiplicidade de recursos que são heterogêneos por natureza e que podem estar dispersos por numerosos domínios administrativos através de grandes distâncias geográficas;
- *escalabilidade (scalability)*: uma grade pode crescer de poucos recursos para milhões. Isto levanta o problema da potencial degradação do desempenho à medida que o tamanho de uma grade cresce. Conseqüentemente, aplicações que requerem um grande número de recursos dispersos geograficamente devem ser projetados para serem extremamente tolerantes a latência;
- *dinamicidade ou adaptabilidade (dynamicity or adaptability):* em uma grade, a falha de um recurso é a regra, e não a exceção. De fato, com tantos recursos, a probabilidade de que algum recurso falhe é naturalmente alta. Os gerenciadores de recursos ou aplicações devem adaptar o seu comportamento dinamicamente a fim de extrair o máximo de desempenho a partir dos recursos e serviços disponíveis.

Neste contexto, uma das questões a serem analisadas em uma grade computacional é a monitoração, pois para prover uma melhor escalabilidade deve-se identificar os recursos computacionais da grade. Segundo Foster [16] A monitoração da grade é a medida e a publicação do estado de um componente da grade em um ponto particular em um determinado tempo, monitorar requer um número de finalidades, incluindo a verificação de status, pesquisando defeitos, ajudando o desempenho, e eliminando erros. Existem várias ferramentas disponíveis para esse tipo de monitoramento como NetLogger [3], Remos [4], NWS [5], LibRastro [6], Ganglia [6] entre outras.

Segundo Bona [14] o monitoramento tem que prever a dinamicidade do ambiente que a grade impõe. Ainda podem ser implementados mecanismos de tolerância a falhas que precisam conhecer o estado do sistema. Desta forma um ambiente de grades necessita prover ferramentas que permitam monitorar seus componentes.

No monitoramento de software, encontra-se um grande problema nas corporações: como inibir a utilização de softwares piratas ou sem licenças, através da monitoração dos softwares instalados. Também seria importante determinar a localização e tempo de utilização de softwares estratégicos para o funcionamento da corporação.

Além disso, em uma grade acadêmica, é possível que um determinado *software* somente esteja disponível em um determinado nó da grade. Essa informação deve ser então utilizada para o escalonamento de tarefas que possuam restrições de execução em uma determinada plataforma e/ou necessitem de um *software* específico para execução.

A ferramenta de monitoramento pode ser usada também para auxiliar uma ferramenta de descoberta onde será possível fazer busca por recursos identificados por exemplo como máquinas com uma determinada quantidade de memória ou até a localização de um determinado software.

As informações de monitoramento de *hardware* e *software* devem ser coletadas máquina a máquina, mas normalmente são centralizadas nas redes locais ou *clusters*. Além disso, periodicamente esses dados são enviados a um elemento central para permitir uma visão global dos recursos da grade. Com relação a este envio dos dados coletados, deve-se levar em conta a largura de banda para não sobrecarregar a rede e não afetar o desempenho da mesma. O envio periódico das informações deve ser feito para monitorar mudanças de características de *hardware* e *software*, enviando as diferenças e guardando um histórico para análise das mudanças.

Deste modo, a motivação para o estudo de monitoramento de recursos em ambiente de computação em grade é resolver o problema da distribuição e alocação dos recursos computacionais para: (1) auxílio na tomada de decisão dos administradores na compra de determinado *software* ou *hardware*, bem como pela renovação ou não de licenças de *software*; e para (2) permitir uma boa alocação de recursos por parte de sistemas gerenciadores de recursos (RMS ou *resource management system*) bem como um bom escalonamento de tarefas por gerenciadores de aplicação em contexto de computação em grade.

Este texto apresenta a proposta de um novo módulo ou parte de um serviço a ser usado por um sistema gerenciador de aplicações para ambiente em grade. Esse serviço pode ser usado por sistemas desenvolvidos a partir de plataforma como EasyGrid [7], GRAND *(Grid Robust Application Deployment*) [9] ou Globus [12]. O Globus fornece um serviço de monitoração e descoberta chamado MDS (*Monitoring and Discovery Service*), mas alguns requisitos que estão sendo levantados neste trabalho parecem não ser totalmente atendidos por esse serviço. O objetivo é obter e publicar informações de *hardware,* algo que é essencial e que já existem ferramentas que disponibilizam, bem como disponibilizar informações de software, o que permitirá um uso geral, podendo ser utilizado também por administradores de rede e gerentes.

Deste modo, um dos objetivos deste trabalho é estudar técnicas e modelos de monitoramento de recursos, a fim de determinar uma forma de monitoramento de recursos, que contemple monitoração tanto de aspectos estáticos quanto dinâmicos de hardware e software para um ambiente de grade, para auxiliar escalonadores de recursos e também administradores de ambiente em grade.

Este trabalho é classificado como um estudo de caso de natureza aplicada, pois, vai testar modelos de monitoramento para um ambiente específico. Ele está sendo desenvolvido dentro do contexto do modelo GRAND [9]. O GRAND (*Grid Robust Application Deployment*) [9] é um modelo de gerenciamento hierárquico de aplicações em ambiente de grade. Seu objetivo é tratar de forma escalável da submissão e monitoramento de aplicações que disparam um número muito grande de tarefas (centenas ou milhares).

O restante deste texto apresenta-se organizado do seguinte modo. Inicialmente, apresentam-se a metodologia, que aborda o que foi pesquisado e métodos utilizados. A seção de estado da arte aborda o levantamento sobre o problema e ferramentas de monitoramento em grade, a seção de modelo proposto aborda a modelagem de como o

protótipo irá funcionar e como será integrado com o GRAND [9], e a seção de considerações finais aborda até onde chegamos com os estudos e até onde iremos.

2. Metodologia

Considerando o contexto de pesquisa do projeto GRAND e das instituições que colaboram para a construção deste ambiente, bem como a realidade de ambientes distribuídos empresariais como o Banrisul, detectou-se a necessidade de informações comuns de monitoramento. Assim, inicialmente foi definido o foco do trabalho e os rumos a serem tomados. Para isso, buscou-se principalmente no IEEE e ACM trabalhos relacionados. Em especial, trabalhos que tratam de monitoramento em grade computacional e modelos hierárquicos ou *peer-to-peer* (P2P) para gerenciar escalabilidade foram analisados. Detectou-se que ferramentas de gerenciamento de recursos que tratam da escalabilidade de processos, normalmente precisam de informações relacionadas a monitoração de softwares e hardwares, como CPU, memória, espaço em disco ou softwares específicos para executar uma tarefa. As técnicas utilizadas tanto pelas ferramentas de gerenciamento como de monitoramento foram estudadas para efetuar a análise de requisitos.

Foi tratado procedimentos de monitoramento de softwares para auxiliar gerentes de infra-estrutura, na identificação da plataforma utilizada e possíveis potencialidades do nó para se submeter a tarefas que sejam escalonadas a ele.

Dentre os sistema de licenciamento de software encontrados na literatura os dois principais tipos o EULA (*End-User License Agreement*) [8], que é a licença de uso do produto, e o PUR (*Product User Rights*) [8], que define regras especiais (como, por exemplo, direito de *downgrade*) para clientes que tenham contratos de licenciamento em volume.

Para definição do modelo também foi pesquisado a respeito do GRAND [9], porque se pretende incluir o serviço de monitoramento implementado neste trabalho no contexto deste projeto. Por isso o próximo passo será a análise do protótipo do AppMan [10] que implementa parte do modelo.

Após esses estudos preliminares, detectamos que nenhuma ferramenta analisada implementa todas as funcionalidades desejadas, como por exemplo, o monitoramento da utilização de determinados softwares.

Um resumo desta análise é apresentado na próxima seção.

Como próximas etapas, deve-se concluir a análise e projeto do sistema, bem como realizar a implementação de um protótipo. Esse protótipo será avaliado em dois contextos. Com relação aos aspectos de coleta de forma escalável, deve-se utilizar a rede do Banrisul. Com relação aos aspectos de integração ao GRAND e a passagem de dados de monitoramento em diferentes nos da grade, deve-se realizar testes em um ambiente de grade experimental, interligando, em princípio, quatro instituições de pesquisa: UniLaSalle (lab24h), UFRGS (gradeP), UFRJ (LabIA) e LNCC (Sinergia).

3. Estado da Arte

Foram analisadas tanto ferramentas que provêm monitoramento em redes locais e em grade, quanto escalonamento de processos em ambiente de grade. Algumas ferramentas fazem monitoramento de utilização de CPU e medição para definir se o processo está alocando muito processador, outras medem a memória disponível e várias outras informações. Além disso, de um modo geral elas indicam quando uma máquina está ou não ativa no ambiente de grade. Quanto as ferramentas de escalonamento, elas foram analisadas para a avaliação de quais características deveriam ser monitoradas para auxiliar um bom escalonamento.

Alguns ambientes de escalonamento como o Condor [24], usam a monitoração para detectar a ociosidade da máquina. Cada máquina publica seu estado para o gerenciador central (central manager), que é o responsável pelo processamento.

Neste caso, ele envia a cada vez informações "estáticas" (quantidade de memória, processador, etc) e "dinâmica" (se há usuário com sessão aberta na máquina, ocupação da CPU, etc). Isso pode causar um tráfego na rede desnecessário, visto que várias destas informações não mudam entre uma atualização e outra.

Acredita-se que uma abordagem, tal como a proposta neste trabalho que será apresentada na próxima seção, em que apenas os dados que mudaram desde a última amostragem são enviados, seria mais eficiente para um ambiente de grade.

A Tabela 3.1 resume algumas das principais características das ferramentas de monitoramento ou visualização analisadas. Na primeira coluna podemos observar as plataformas que as ferramentas executam, na segunda coluna temos a linguagem nas quais foram implementadas, na terceira temos a coluna que diz se a ferramenta integra a um outro *framework*. Finalmente, coluna de informações indica se a ferramenta monitora CPU, Memória e softwares, a última coluna mostrará se a ferramenta devolve os dados monitorados de uma forma gráfica ou textual. Note que a ferramenta Pajé é apenas uma ferramenta de visualização de informações coletadas por outras ferramentas de monitoramento, tendo ainda a possibilidade de mostrar a execução de processos e *threads* ao longo do tempo.

Tabela 3.1 - Lista de ferramentas de monitoramento

Ferramenta	Plataforma	Linguagem	Integra outras Ferramentas	Informações			Visual. Dos Dados	
				CPU	Mem.	Software	Gráfica	Textual
GridRM [22]	Linux/Windows	Java	Ganglia	X	X		X	
NWS [5]	Unix/Linux	C		X	X			X
NetLogger [3]	Linux/Windows	C/C++, Java, Perl e Python	Ganglia e MonAlisa	X				X
R-GMA [20]	?	SQL, C	Qualquer que utilize GMA		X			X
Remos [4]	?	C						X
EasyGrid [7]	Linux/Windows	Java		X				X
GridBus [21]	Unix /Windows	C, Java, C# e Perl	Alchemi	X			X	
HawKeye [23]	Linux/Solaris	C, Perl	Condor	X	X		X	X
Ganglia [6]	Linux	C	LibRastro	X	X		X	
Pajé [6]	Linux	C	LibRastro				X	
Descoberta e Monitoramento de recursos em Rede de Computação usando Agentes Móveis [19]	Linux/Windows	Java		X	X		X	

Nesta Tabela 3.1, os "x" indicam presença da característica, enquanto uma "?" indica a ausência de informação conclusiva na literatura.

4. Modelo Proposto

Neste trabalho propomos um modelo para realizar o monitoramento de máquinas em um ambiente de grade. Este modelo busca resolver as deficiências de ferramentas tradicionais de monitoramento quanto a escalabilidade e obtenção de características de *software* dos nós da grade.

Inicialmente, apresenta-se uma visão geral do modelo seguido do levantamento de requisitos e análise de projetos preliminares.

4.1. Visão Geral

A Figura 4.1 abaixo mostra o modelo do funcionamento do agente que fará a busca de determinadas características e as enviará ao servidor para serem disponibilizadas aos usuários (pessoas ou sistemas computacionais).

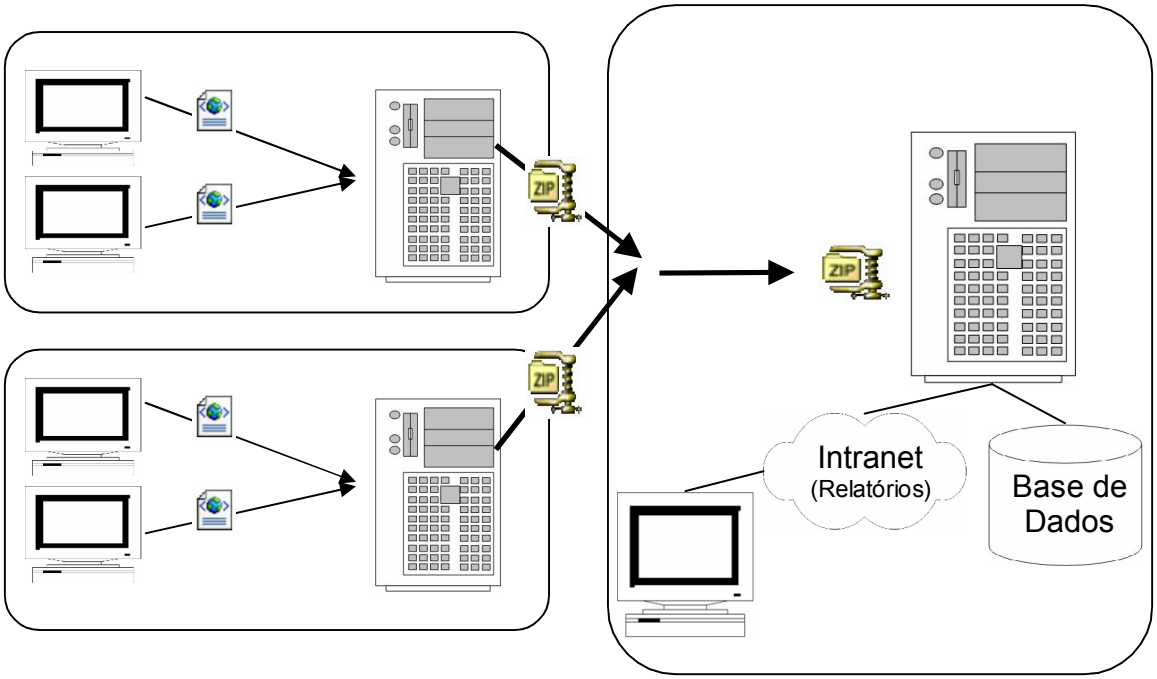

Figura 4.1 – Sistema de Gerenciamento de Recursos

Estas informações serão coletadas de acordo com os níveis de prioridade e freqüência de atualização. Algumas informações serão coletadas somente uma vez ao dia, pois a possibilidade de alteração das mesmas é pequena, por exemplo, memória disponível, domínio e percentual de espaço livre em disco. Também terá informações coletadas ao longo do dia com um intervalo de amostragem de acordo com testes. Por exemplo, monitoramento de utilização e instalação de *softwares*. Informação de utilização além de precisar ser coletado com mais freqüência, precisa ser disponibilizado, sempre que houver mudança significativa de modo a fornecer indicações úteis aos gerenciadores de recursos a aplicações.

Na Figura 4.1 nota-se que elipses representam um nó da grade. Dentro delas temos recursos computacionais quaisquer, tipicamente computadores *desktop*, que publicam informações sobre seu estado em formato XML. Dentro de cada elipse ou nó da grade, existe um computador denominado centralizador que publica as informações do conjunto de recursos do nó da grade.

Cada um dos centralizadores está ligado a um centralizador pai, que é responsável por obter informações de todos os centralizadores filhos, formando uma árvore de centralizadores.

Caso a grade possua muitas máquinas e/ou muitos domínios (da ordem de milhares), haverá um conjunto de centralizadores pais interligados através de uma arquitetura *peer-to-peer* (P2P).

O principal problema da computação em grade e prover a escalabilidade, por isto é muito importante a utilização de sistemas de monitoramento que funcione bem com dezenas, centenas e até milhares de nós. Para funcionar com milhares de máquinas teremos

uma quantidade muito grande de informação, com isto o custo de armazenamento e processamento será elevado. Uma alternativa é usar uma hierarquia interligada de forma p*eer-to-peer* (P2P), com isto terá a vantagem de não ter um único ponto de falha.

Segundo Clark [13] na computação P2P as máquinas compartilham dados ou recursos, como ciclos de computação ociosos ou capacidade de armazenamento, via Internet ou redes privadas. Em uma rede P2P a noção de cliente e servidor é relativa, de forma que as máquinas apresentam funções tanto de clientes como de servidores.

Na computação P2P as máquinas podem se comunicar diretamente e gerenciar suas tarefas sem utilizar servidores centrais. Isto permite que a computação P2P seja mais escalável que o modelo tradicional cliente-servidor, apresentando-se como uma das melhores abordagens para criar redes globais de compartilhamento de recursos.

4.2. Monitoramento de Hardware

Deseja-se utilizar os modelos e padrões de implementação do GGF [11], pois ele é utilizado por vários projetos de grade incluindo o GRAND (***G**rid **R**obust **A**pplicatio**n** **D**eployment*) [9]. O GGF prevê, por exemplo, que é possível utilizar índices que avaliam número de processos na fila e uso de CPU, memória, rede, entre outros. O fórum de discussão sobre grades GGF empenhou-se na criação da GMA (*Grid Monitoring Architecture*) [15], a qual define uma arquitetura de monitoração de recursos, incluindo rede, CPU, disco, memória, através de sensores e centralização das informações em um serviço de diretório para a busca posterior de recursos disponíveis.

O modelo de gerenciamento GRAND realiza a submissão e o controle de forma distribuída e hierárquica de aplicações compostas por uma grande quantidade de tarefas em um ambiente de computação em grade. Este modelo disponibiliza mecanismos para o gerenciamento hierárquico que pode controlar a execução das tarefas preservando a localidade dos dados ao mesmo tempo que reduz a carga das máquinas de submissão.

As informações disponibilizadas, serão utilizadas para que não se submeta muitas tarefas a serem executadas em uma máquina, assim o usuário que estiver utilizando a mesma, nem notará que sua máquina está sendo utilizada com outras tarefas.

Como proveremos a descoberta dinâmica de recursos, quando um nó estiver disponível poderemos utilizá-lo para submissão de outra tarefa. Note que neste contexto, o a descoberta (*discovery*) é um serviço que permite que um sistema recupere a descrição de recursos.

4.3. Monitoramento de Software

O modelo proposto neste trabalho também irá monitorar *softwares* instalados e a utilização de alguns previamente configurados, as informações de softwares instalados servirão para auxiliarem administradores de infra-estrutura. Para um escalonamento de tarefas convencional, basta saber se um determinado *software* encontra-se ou não instalado em uma máquina. Mas para definição de quais informações seriam necessárias para

auxiliar um administrador de sistemas, foi preciso realizar um levantamento dos tipos de licenças mais comuns.

Existem dois tipos principais de Licenciamento de software o EULA (*End-User License Agreement*) [8], que é a licença de uso do produto, e o PUR (*Product User Rights*) [8], que define regras especiais (como, por exemplo, direito de *downgrade*) para clientes que tenham contratos de licenciamento em volume.

Existem alguns sub-tipos de licenciamento onde temos :

- *Open License* - é um programa de licenciamento em volume destinado a empresas que pretendem adquirir cinco licenças ou mais de um determinado título de software. As empresas que participam do *Open License* podem acessar suas informações de licenciamento através do site seguro Microsoft *eOpen*.

- *Open Business* - modalidade que oferece aos clientes melhor preço melhor que aquele praticado no varejo, adquirindo cinco ou mais licenças. Através do *Open Business*, as empresas podem combinar qualquer conjunto de produtos Microsoft para se qualificar ao mínimo de cinco licenças.

- *Open Volume* - modalidade de licenciamento que permite uma economia potencial maior para as empresas, desde que o pedido inicial de um ou mais grupos de produtos seja em maior quantidade (aplicativos, sistemas, servidores).

- *Open Value* - permite que as empresas mantenham seu *software* sempre atualizado através do programa *Software Assurance* (AS), dividindo o pagamento do licenciamento em parcelas anuais a partir de um pedido inicial de cinco ou mais licenças. Se a empresa optar pela alternativa de âmbito corporativo, a economia será ainda maior.

- *Select License* - é um programa de licenciamento em volume concebido para empresas que tenham 250 ou mais PCs, e que podem prever e programar a aquisição de suas licenças de software por um período de três anos. Com o *Select License* os clientes recebem um nível de preço de volume para cada grupo de produtos selecionados (aplicativos, sistemas ou servidores), com base em uma estimativa de três anos.

- *Enterprise Agreement* - é um programa de licenciamento em volume destinado a organizações que tenham 250 ou mais PCs e que pretendem padronizar determinados produtos pagando por eles preços especiais. Há três tipos de *Enterprise Agreements: Enterprise Agreement* é provavelmente a melhor alternativa para empresas com 250 ou mais computadores *desktop* que pretendem padronizar sua plataforma em um ou mais Produtos da Plataforma Corporativa Microsoft [*Office Professional*, *Windows Professional* atualização e *Core CALs (Client Access*

License)] a partir de um acordo de três anos. *Enterprise Subscription Agreement* destina-se a clientes corporativos com 250 ou mais computadores *desktop* que preferem licenciar produtos Microsoft mediante uma assinatura. O *Enterprise Subscription Agreement* permite que as organizações padronizem sua plataforma em um ou mais Produtos da Plataforma Corporativa Microsoft através de um acordo de três anos.

Existe ainda o licenciamento Acadêmico. As instituições acadêmicas podem se qualificar para um dos programas de licenciamento acadêmico Microsoft. Os produtos acadêmicos estão disponíveis através dos canais de Varejo e de Licenciamento em Volume e podem ser adquiridos por clientes que cumprirem os requisitos necessários ao preço acadêmico.

Outro tipo de licença é o Programa Open License Governamental. O Programa Open License Governamental oferece preços especiais de licenciamento em volume para órgãos governamentais de pequeno e médio porte. É uma opção de licenciamento ideal para aquelas organizações públicas que precisam adquirir quantidades menores de licenças e estão em busca de um modelo de licenciamento simples e flexível.

Define-se "Organização Governamental" como qualquer órgão, secretaria ou entidade federal estadual ou municipal assim caracterizada por estatuto. Para adquirir produtos dentro da modalidade Open License Governamental, a organização deverá cumprir os requisitos necessários.

Finalmente, existe o Programa Open License Filantrópico. O Programa Microsoft Open License Filantrópico permite que entidades sem fins lucrativos adquiram várias licenças de software (ao invés de múltiplas caixas de software) a preços reduzidos. Para se qualificar ao Programa Open License para entidades filantrópicas , a organização precisa estar registrada com esta característica. As informações relativas ao Licenciamento Open para entidades filantrópicas são específicas à política adotada nos Estados Unidos.

Embora possa ser aplicado para ambiente coorporativo, de fato este modelo de busca de informações de software é fundamental para algumas aplicações acadêmicas no contexto do projeto GRAND. A identificação de softwares em ambientes não corporativos pode ser feita para determinar se um software free encontra-se ou não instalado em uma máquina para aplicar o escalonamento correto das tarefas de uma aplicação. Por exemplo, no contexto do GRAND, uma das aplicações implementadas de bioestatística necessita do YAP Prolog para a execução das tarefas.

4.4. Levantamento de Requisitos

Segundo Larman [23], requisitos são "uma descrição das necessidades ou desejos para um produto". Com essa fase, busca-se a identificação não ambígua de características que um sistema computacional deve atender. Realizou-se o levantamento de requisitos considerando-se as necessidades típicas de escalonadores de recursos como o GRAND e junto a gerentes do Banrisul para a parte de licenças. Os seguintes requisitos, foram construídos de acordo com o formato apresentado por Larman [23].

Caso de Uso : Solicitação de informações

Atores : Usuários (Pessoas ou Sistemas escalonadores)

Finalidade :

- *Pessoas* - Obter informações pesquisadas através da interface referente a determinado nó da grade;

- *Sistema* - Obter informações para auxiliar no escalonamento através de características contidas em um arquivo xml.

4.5. Caso de uso

Na Figura 4.2 mostra os meios de iteração com os dois possíveis usuários do sistema se for um usuário (pessoa) mostrará um gráfico de estado do nó da grade isto é, vê através de interface gráfica e se for um usuário (sistema) mostrará informações sobre o estado do nó da grade através de um arquivo XML.

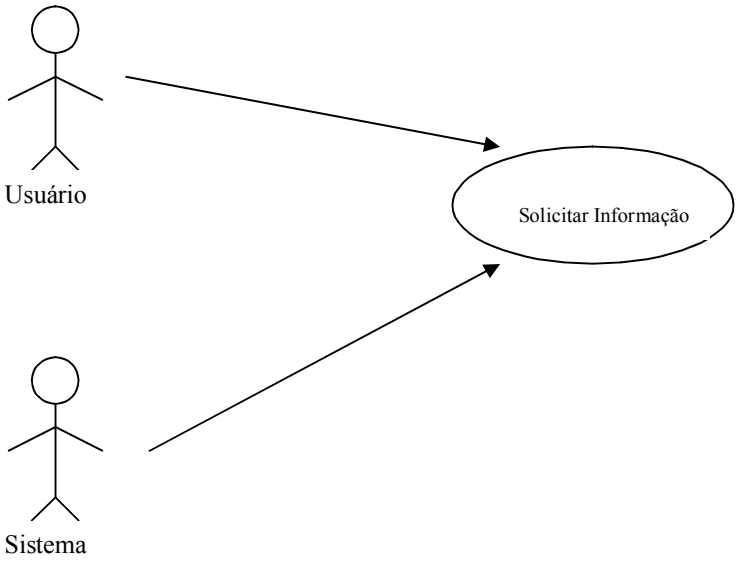

Figura 4.2 - Diagrama UML de caso de uso

4.7 Diagrama de Seqüência

Diagrama que mostra a seqüência do tráfego das informações pelo sistema.

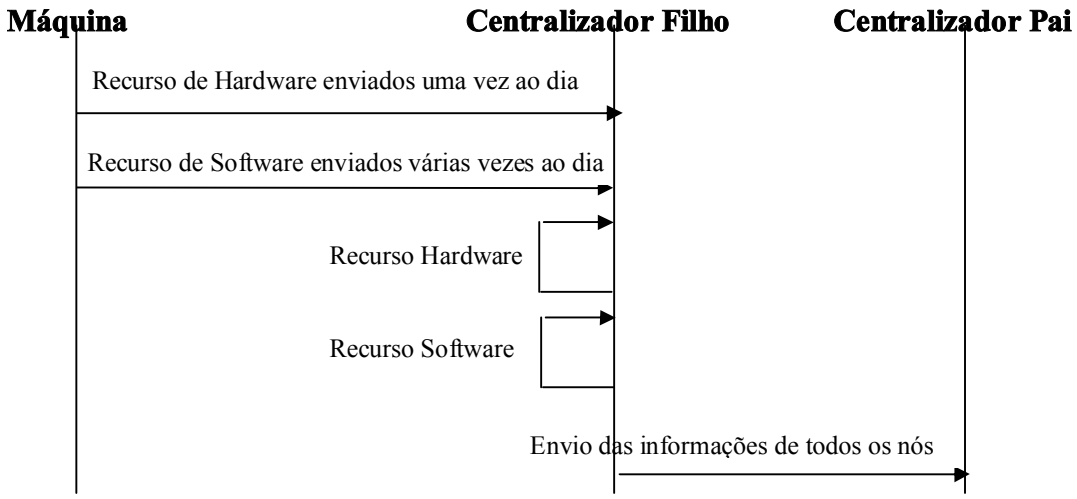

Figura 4.3 - Diagrama UML de Seqüência do sistema

Recursos de *hardware* são tipos diferentes de recursos de *software*, pois alguns são enviados uma vez ao dia, como memória disponível, e informações de *software* serão enviadas mais de uma vez ao dia, como *softwares* instalados e utilização dos mesmos.

Diagrama que mostra o fluxo caso o centralizador não mande informações no tempo previamente estabelecido.

Figura 4.4 - Diagrama UML de Seqüência caso não seja recebida informação

Caso precise enviar um ack dizendo que está ativo enviará também as informações coletadas.

5. Considerações Finais

Os estudos realizados neste trabalho serviram para identificar as inúmeras ferramentas que provêm o monitoramento de informações para auxiliar no gerenciamento e

escalabilidade de processos. Foi visto e discutido o modelo que será utilizado, e não foi encontrada nenhuma ferramenta que faça a monitoração de *softwares*. As ferramentas analisadas somente se preocupam com a utilização do *hardware*. Além disto, normalmente elas buscam apenas informações de uso de *hardware* para escalonamento, ou então apenas dados sobre as características das máquinas. Poucos são os sistemas de monitoramento que provêm ambos os tipos de informações sobre recursos de *hardware*. Acreditamos que a disponibilização destas informações, bem como a proposta de uma organização hierárquica e p*eer-to-peer* constituem uma importante contribuição científica no contexto de computação em grade.

A tolerância a falhas é um ponto muito importante a ser tratado por qualquer sistema computacional que execute em ambiente de grade. Neste trabalho, iremos abordar apenas questões de falhas temporárias de comunicação.

A falha do componente que possui as informações de monitoração poderia ser facilmente tratada por uma abordagem primário-backup [17]. No entanto, esse não é o ponto principal deste trabalho e será deixado como trabalhos futuros.

Uma das contribuições esperadas neste trabalho é uma interface para publicação de informações para o usuário. Essa interface servirá mais para verificação na fase de testes e depuração, uma vez que o foco do trabalho é auxílio a tomada de decisões de escalonamento no protótipo AppMan [10] criado no projeto GRAND [9]. A construção de uma interface com características elaboradas considerando as questões de escalabilidade de recursos e tarefas apesar de constituir um interessante tópico de pesquisa, encontra-se fora do escopo deste trabalho.

7. Referências Bibliográficas

[1] http://www.gridcomputing.com/gridfaq.html, acessado em 19/04/2006;

[2] FOSTER, I; KESSELMAN,C.; TUECKE,S. The anatomy of the grid;enabling scalable virtual organizations. The International Journal of High Performance Computing Applications. V.15, n.3, Fall 2001.;

[3] http://www-didc.lbl.gov/NetLogger/, acessado em 19/04/2006;

[4] http://www-2.cs.cmu.edu/~cmcl/remulac/remos.html, acessado em 19/04/2006;

[5] http://nws.cs.ucsb.edu/, acessado em 19/04/2006;

[6] NEVES, M.; SCHEID, T.; SCHNORR, L. M.; CHARÃO, A.. Integração de Ganglia, libRastro e Pajé para o Monitoramento de Aplicações Paralelas. Quinto Workshop em Sistemas Computacionais de Alto Desempenho, WSCAD., 2004.

[7] NASCIMENTO, A. P., SENA, A. C., SILVA, J. A., VIANNA, D. Q. C. , BOERES ,C., REBELLO, V. E. F. Managing the Execution of Large Scale MPI Applications on Computational Grids, SBAC, 2005.

[8] http://www.microsoftvolumelicensing.com/userights/, acessado em 30/05/2006;

[9] VARGAS ,P. K., GRAND: Um Modelo De Gerenciamento Hierárquico De Aplicações Em Ambiente De Computação Em Grade, Tese de Doutorado. 2006.
[10] VARGAS, P. K.; SANTOS, L. A. S.; DUTRA, INÊS DE CASTRO; GEYER,CLÁUDIO F. R.," An implementation of the GRAND hierarchical application management model using the ISAM/EXEHDA system," In: III Workshop on Computational Grids and Applications. January 31 February 2, 2005. Petrópolis, RJ, Brazil.
[11] http://www.ggf.org/, acessado em 19/04/2006;
[12] http://www.globus.org/, acessado em 19/04/2006;
[13] D. CLARK, "Face-to-Face with Peer-to-Peer Networking", IEEE Computer, Vol. 34, No. 1, pp. 18-21, 2001;
[14] BONA, L. C. E. , HyperGrid: Uma Plataforma Distribuída e Confiável para Computação em Grade, Tese de Doutorado ,2004;
[15] TIERNEY,B. *et al*., A grid monitoring architecture. Available at http://www.didc.lbl.gov/ggf-perf/gma-wg/papers/gwd-gp-16-3.pdf;
[16] FOSTER I; KESSELMAN,C,"Grid2 - BluePrint", 2003, p.321-351;
[17] BUDHIRAJA, N. et al., The Primary-Backup Approach. In: Distributed Systems. ACM Press. New York, p.199-216, 1993.;
[18] MATTOS, E.C.T, A Análise e construção de um ambiente de grade computacional peer-to-peer com ênfase no balanceamento de carga. São Carlos, 12/2003.;
[19] JUNIOR,J.C., JUNIOR, I.D.M., Descoberta e Monitoramento de Recursos em Redes de Computadores usando Agentes Móveis, UECE;
[20] http://www.r-gma.org/, acessado em 19/04/2006;
[21] http://www.gridbus.com/, acessado em 04/06/2006;
[22] SMITH,G.,BAKER, M., GridRM: An Extensible Resource Monitoring System, University of Portsmouth, 06/2003;
[23] LARMAN, C, Utilizando UML e Padrões,Porto Alegre, 2000;
[24] TAHIN,D., TANNEMBAUM, T., LIVNY, M. Condor and the Grid in: BERMAN, F.; FOX, G,. HEY,T. Grid Computing: Making the global infrastructure a reality. [S.I.]: John Wiley, 2003;