Paloma Sacramento dos Santos

Levantamento e Análise das Tecnologias de Gerenciamento de Grades Computacionais

Orientador:
Prof. Msc. Antônio Augusto Teixeira Ribeiro Coutinho

Curso de Engenharia de Computação
Departamento de Tecnologia
Universidade Estadual de Feira de Santana

Feira de Santana – BA

Setembro / 2008

Paloma Sacramento dos Santos

Levantamento e Análise das Tecnologias de Gerenciamento de Grades Computacionais

Monografia apresentada à coordenação do curso de Graduação em Engenharia de Computação da Universidade Estadual de Feira de Santana como requisito para a obtenção do título de Bacharelado em Engenharia de Computação.

Orientador:
Prof. Msc. Antônio Augusto Teixeira Ribeiro Coutinho

Curso de Engenharia de Computação
Departamento de Tecnologia
Universidade Estadual de Feira de Santana

Feira de Santana – BA

Setembro / 2008

Monografia de Projeto Final de Graduação sob o título *"Levantamento e Análise das Tecnologias de Gerenciamento de Grades Computacionais"*, defendida por Paloma Sacramento dos Santos e aprovada em 09 de Setembro de 2008, em Feira de Santana - Bahia, pela banca examinadora constituída pelos professores:

Prof. Dr. Antônio Lopes Apolinário Junior
Departamento de Tecnologia - UEFS
Examinador

Prof. Msc. Antônio Augusto Teixeira Ribeiro Coutinho
Departamento de Tecnologia - UEFS
Orientador

Prof. Msc. José Amâncio Macedo Santos
Departamento de Tecnologia - UEFS
Examinador

Dedico esta monografia a meus pais,
pelo exemplo, dedicação e carinho

Agradecimentos

A DEUS força que sempre esteve comigo.

Aos meus pais pelo incentivo e amor em todos os momentos.

Aos meus irmãos companheiros constantes.

Aos meus amigos Leo, Ligia e Duda pelo apoio nos meus melhores e piores momentos.

Aos colegas da UEFS pelos cinco anos de risos e alegrias.

Ao meu orientador Prof. Antônio Augusto por sempre acreditar no meu trabalho.

Sumário

Lista de Figuras

Lista de Tabelas

Resumo

Esse trabalho apresenta os resultados de uma pesquisa submetida à comunidade brasileira de grade computacional com o objetivo de medir o uso das tecnologia, bem como o nível de gerenciamento empregado nestes sistemas. Foram realizadas comparações entre as arquiteturas levantadas e com outras pesquisas internacionais similares realizadas em 2003 e 2005. Questões como gerenciamento de falhas, de recursos, ferramentas de monitoramento usadas e dificuldades em manter o ambiente foram abordadas.

Palavras chaves: Grade Computacional, Gerência de Grades, Sistemas de Monitoramento, Gerenciamento de Recursos.

1 Introdução

1.1 Contextualização

O aumento do poder computacional é governado há muito anos por regimes exponenciais. Essa forma de crescimento vêm se mantendo principalmente a custa do desenvolvimento de diferentes componentes como microprocessadores, unidades com maior capacidade de armazenamento e novas tecnologias de transmissão de dados. Entretanto, o que historicamente tem sido ocasionado apenas por desenvolvimentos independentes, vem contando recentemente com o apoio da computação distribuída e paralela (COULOURIS; DOLLIMORE; KINDBERG, 2005) .

Dentre esses tipos de sistemas, as grades computacionais têm se destacado como promissora e atraente plataforma de execução de aplicações paralelas devido à possibilidade de alocar uma quantidade enorme de recursos (milhares de máquinas conectadas pela Internet) com custo inferior as alternativas tradicionais, baseadas em supercomputadores paralelos. Entretanto, a revolução computacional prometida pelas grades, no sentido de oferecer computação ubíqua e acesso sob demanda a recursos e serviços, exige soluções para os desafios encontrados nas suas características principais como alta heterogeneidade, complexidade e distribuição (através de múltiplos domínios administrativos).

A gerência nesse tipo de ambiente envolve questões como a monitoração do desempenho dos recursos, a detecção de falhas, o uso de mecanismos de recuperação, a predição do desempenho de um serviço baseada nas informações de gerência e o escalonamento de tarefas (BUCHHOLZ, 2008). Atualmente, diversos grupos têm se interessado em desenvolver padrões e arquiteturas para o monitoramento de grades (TIERNEY, 2008). Esses grupos, através do Global Grid Forúm (GGF, 2008), reúnem todos os esforços no sentido de tornar esses padrões e arquiteturas interoperáveis.

Semelhante ao gerenciamento em rede de computadores tradicionais, a gerência em grade envolve a supervisão de elementos como os protocolos usados na troca de mensagens,

mecanismo de segurança e a manipulação e representação da informação de gerência. Entretanto, a gerência em grades computacionais é diferenciada do modelo tradicional no que se refere à escalabilidade e ao número de recursos e componentes geograficamente dispersos através de redes de longa distância (TIERNEY, 2008).

1.2 Objetivos

O objetivo desta monografia é realizar um levantamento do estado da arte da gerência de grades computacionais. Com este estudo foi identificado:

- Principais ambientes de grades utilizados nacionalmente e suas características.
- Mecanismos utilizados para prover e avaliar as informações de gerência desses ambientes.
- Desafios de gerência em grades.
- Trabalhos sendo desenvolvidos pela comunidade internacional e nacional relacionados a novas arquiteturas e padrões de gerência em grades.

Para auxiliar no levantamento dessas informações foi apresentado à comunidade um questionário para identificar os ambientes utilizados bem como os principais problemas relacionados à prática da gerência de grades.

1.3 Organização do Trabalho

No capítulo 2 será abordado os conceitos e características que permitem definir um sistema de grade computacional, bem como as propostas de arquiteturas voltadas à esses ambientes. O capítulo 3 trata sobre os sistemas de gerenciamento em grades. No capítulo 4 apresentamos uma pesquisa aplicada à comunidade brasileira. Na seção 4.1 foram identificados os principais ambientes de grade utilizados nacionalmente. Posteriormente, centralizamos a pesquisa em dois focos principais: gerenciamento de falhas e gerenciamento de configuração, apresentados respectivamente nas seções 4.5 e 4.6, finalizando a seção com uma análise qualitativa das informações. No capítulo 5, discutimos as lições obtidas com este trabalho e apresentamos nossas observações finais.

2 Computação em Grade - Grid Computing

2.1 Histórico, Conceitos e Características

Independente da época ou década considerada, os problemas computacionais se tornaram tão complexos, que passaram a desafiar a capacidade das máquinas existentes. Diferentes idéias surgiram para ampliar este poder de processamento, envolvendo desde a criação de supercomputadores até a interconexão de sistemas locais ou remotos partilhando recursos e armazenamento em um ambiente integrado. Partindo dessa necessidade o paradigma de computação distribuída passou a utilizar as redes locais, de alta velocidade e a Internet para construir sistemas de alto desempenho como os *clusters* e as grades computacionais.

Com o desenvolvimento da Internet nos anos 90, os primeiros projetos globais de grades computacionais surgiram e provaram que o novo conceito trabalhava muito bem. O primeiro projeto, distributed.net (DCTI, 2008) fundado em 1997, utiliza os usuários comuns da Internet, ao redor do mundo, somando o seu poder computacional para pesquisas acadêmicas. O segundo projeto, SETI@home (SETI, 2008), fundado em 1999, utiliza um software instalado nas máquinas dos voluntários para processar dados astronômicos a procura de vida extraterrestre. Em especial, esse projeto provou que além de eficiente o paradigma de grades economiza tempo e custos. Enquanto um supercomputador IBM - ASCi de 12 *TERAFlops* custa aproximadamente 110 milhões de dolares o projeto SETI@home consegue em média 15 *TERAFlops* pelo preço de 500.000 mil dólares (VOLCKAERT et al., 2004). Contudo nota-se que as aplicações e o escopo de trabalho dos supercomputadores e das grades computacionais são diferentes. Estas últimas surgiram do interesse e da necessidade de se utilizar a infra-estrutura apresentada pela Internet.

De acordo com (FOSTER, 2002), (CIRNE, 2008), (KESSELMAN; FOSTER, 1998) e (BOTELORENZO; DIMITRIADIS; GÓMEZ-SÁNCHEZ, 2003) uma grade pode ser classificada como

um sistema com as seguintes características:

- Heterogeneidade: Os componentes que formam a estrutura da grade tendem a ser extremamente heterogêneos. Ou seja, a grade deve estar preparada para lidar com softwares de várias versões, diversas tecnologias de rede, recursos de várias gerações, hardwares, instrumentos e serviços dos mais variados tipos;

- Coordenação de recursos não centralizados: Devido à alta dispersão geográfica, uma grade pode atingir uma escala global e possuir serviços espalhados em várias localidades do mundo. Surge o problema de como controlar esses recursos distribuídos sobre múltiplos domínios administrativos, cada um com suas políticas de acesso à recursos e serviços;

- Utilização de padrões abertos, protocolos e interfaces gerais: Uma grade é construída por protocolos e interfaces com propósitos múltiplos, que tratam da autenticação, da autorização, da descoberta e do acesso aos recursos. Essas questões são de tal forma fundamentais na construção do sistema que esses protocolos e interfaces devem ser construídos com padrões abertos, facilitando a interoperabilidade entre os ambientes;

- Alta escalabilidade: A grade deve lidar com uma quantidade de recursos que variam de alguns poucos até milhões;

- Acesso transparente à recursos e serviços: Essa característica permite que uma grade seja vista pelo usuário como um único computador virtual;

- Acesso confiável: Deve ser assegurada a entrega dos serviços sob as métricas de QoS (*Quality of Service*) estabelecidas;

- Acesso permanente: A grade deve conceder acesso aos recursos disponíveis adaptando-se à um ambiente dinâmico no qual o fracasso de recurso é comum.

Uma plataforma que ofereça todos os atributos acima certamente será uma grade (CIRNE, 2008). Contudo, a ausência de alguma das características listadas não deve automaticamente desqualificar uma determinada plataforma.

2.2 Classificação

Baseada em sua área de aplicação, um sistema em grade pode ser dividido em três classes (VOLCKAERT et al., 2004) e (KRAUTER; BUYYA; MAHESWARAN, 2002): grades de computação, grades de dados e grades de serviços apresentadas na Figura 1.

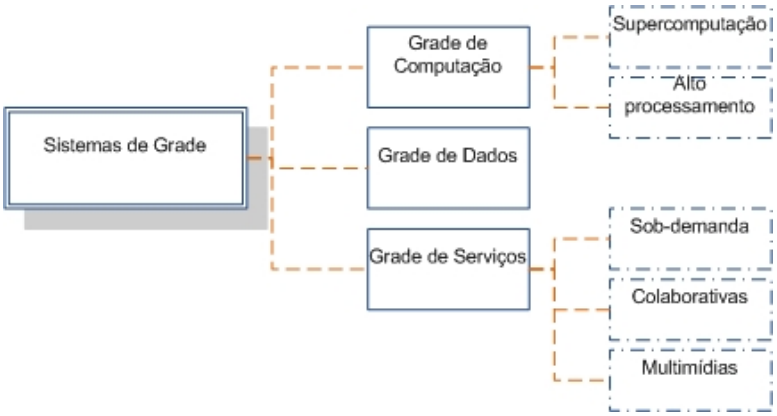

Figura 1: Taxonomia dos Sistemas de Grade (VOLCKAERT et al., 2004).

- Grades de computação podem oferecer mais processamento que qualquer uma de suas máquinas constituintes. Dentro dessa classificação, podem existir duas categorias: supercomputação distribuída e alto-processamento. A supercomputação tenta encurtar o tempo de execução de uma tarefa processando-a em paralelo entre múltiplas máquinas, enquanto o alto-processamento é projetado para processar grandes filas de trabalho (por exemplo, trabalhos de busca massiva). Exemplos de grades de computação são os projetos que trabalham com doações de ciclos de CPU como o SETI@home, distributed.net e fightAIDS@home (FIGHTAIDS, 2008);

- Grades de dados sintetizam a informação de repositórios de dados distribuídos, como bibliotecas digitais e *data warehouses*. Geralmente, esses sistemas incluem dados com finalidades especiais, permitindo correlacionar as informação presentes em multiplas fontes de dados. O projeto LCG-LHC (*Large Hadron Collider*) (LCG, 2008) é um exemplo deste tipo de grade. O objetivo desse projeto é produzir cerca de 15 Petabytes (15 milhões de Gigabytes) de dados físicos experimentais que cientistas de todo o mundo terão acesso para análise. Os projetos European DataGrid (DATAGRID, 2008) e Globus (GLOBUS, 2008) também estão voltados para o desenvolvimento de tecnologias de organização, catálogo, administração e acesso à dados distribuídos;

- Grades de serviços oferecem serviços que não podem ser providos por uma única máquina. Esta classificação pode ser divida em três categorias: grades sob-demanda, colaborativas e multimídias. Uma grade colaborativa conecta os usuários e suas aplicações em grupos de trabalho virtuais. Em grades sob-demanda os recursos são agregados dinamicamente para prover outros serviços. Uma Grade de multimídia provê uma infra-estrutura para aplicações multimídia em tempo real. Isto requer suporte a QoS através de várias máquinas diferentes, considerando que uma aplicação multimídia em uma única máquina pode ser implantada sem QoS.

2.3 Arquiteturas voltadas para Grades

O objetivo dessa seção é oferecer uma introdução teórica e uma subdivisão dos projetos de arquitetura em grades. O entendimento desses padrões são necessários para visualizar como uma possível camada de gerenciamento se enquadra nessas arquiteturas. Recentemente, a computação em grade tem se diferenciado dos sistemas distribuídos convencionais pelo surgimento de aplicações inovadoras sob o ambiente de rede. Para dar suporte a esse ambiente algumas arquiteturas estão sendo pospostas. De forma resumida, estas arquiteturas podem ser agrupadas em três tipos diferentes (YANG et al., 2003):

- Arquiteturas por abstração em camadas: Nesse tipo de arquitetura os serviços são agrupados em camadas. Quanto mais elevada a camada maior é a abstração das suas funcionalidades. Os serviços de cada camada podem usar somente os seus próprios serviços e os serviços das camadas inferiores. O benefício em se utilizar essa arquitetura reside na facilidade encontrada ao adicionar novos serviços. Porém, na maioria dos casos, a formação de camadas distintas é difícil de ser construída. Isso é um problema típico, pois serviços complexos são difíceis de serem reduzidos a protocolos básicos. Um exemplo de arquitetura deste tipo é o Globus *Data Grid Architecture* (YANG et al., 2003);

- Arquiteturas espaço conceitual: Esse método não se preocupa em formar camadas, sendo a arquitetura constituída por blocos diferentes que acabam por se agrupar em metadados, dados, recursos, serviços e protocolos;

- Arquiteturas hibridas: São definidas camadas, cada uma com seu próprio espaço conceitual. A camada mais baixa é responsável pela autorização e autenticação enquanto as camadas superiores são utilizadas, dentre outras funções, para controlar o escalonamento de recursos e o registro das tarefas.

O propósito da arquitetura em grade é permitir o compartilhamento e a interoperabilidade entre qualquer participante com recursos destinados a grade. Em ambientes altamente distribuídos essa interoperabilidade significa prover a comunicação utilizando protocolos comuns. É fundamental trabalhar com protocolos, porém somente essa alternativa não soluciona o problema da interoperabilidade, pois os protocolos atuam em problemas específicos e limitados a um domínio próprio ou a uma interface. Também não é possível criar um único protocolo para ser usado na negociação ou acesso a qualquer espécie de recurso, devido tanto pelas diferença entre esses recursos quanto por existir uma grande quantidade de redes envolvidas em uma tecnologia de grade computacional (YANG et al., 2003). No entanto, os esforços atuais estão focados na criação de protocolos e interfaces que, em conjunto, venham a criar uma estrutura que possa favorecer a interoperabilidade.

Partindo para uma análise no sentido de formalizar e propor um conjunto de especificações técnicas e ferramentas de software, pesquisadores do projeto Globus e da IBM (IBM, 2008) uniram-se para compatibilizar os desenvolvimentos de tecnologias voltadas à grades utilizando como base os conceitos de serviços orientados à web. Essa compatibilização foi provida através do padrão OGSA (*Open Grid Services Architecture*) (OGSA, 2008).

Os modelos arquiteturais para grades computacionais têm o objetivo de oferecer uma padronização que permita a interoperabilidade entre diferentes organizações virtuais. Aspectos como autenticação, autorização, mecanismos de troca e mensagem, compartilhamento de recursos, escalonamento e balanceamento de tarefas são alguns pontos que devem ser abordados para uma arquitetura de grade computacional (KESSELMAN; FOSTER, 1998). Inspirado no modelo arquitetural do TCP/IP (KESSELMAN; FOSTER, 1998) propõe dois modelos conceituais para arquitetura de grades (Figura 2).

A Figura 2(a) apresenta uma arquitetura em cinco camadas:

- Ambiente - Neste nível, os componentes implementam operações específicas locais que ocorrem em cada recurso como resultado das operações de compartilhamento nos níveis superiores;
- Conectividade - Define os protocolos básicos de comunicação e autenticação para as transações da grade;
- Recursos - Define os protocolos e interfaces que fornecem negociação, monitoramento, controle, geração de relatórios e os detalhes envolvidos nas operações com

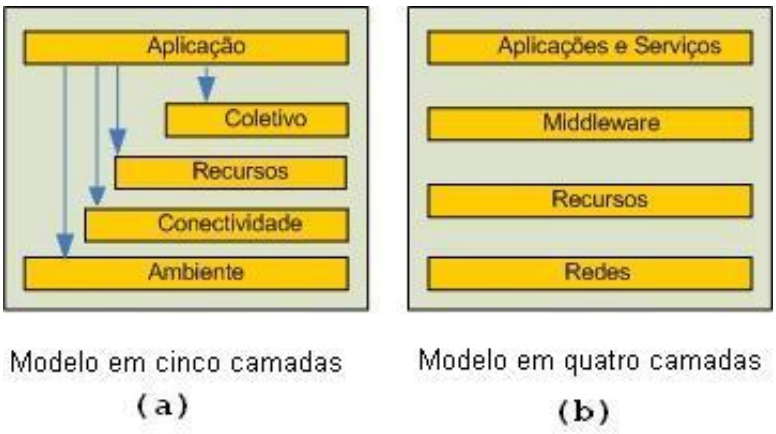

Figura 2: Modelos Arquiteturais de Grades (KESSELMAN; FOSTER, 1998).

recursos individuais;

- Coletivo - Diferente do nível de recurso, que trata os recurso individualmente, neste nível os componentes atuam na interação entre coleções de recursos;
- Aplicação - Compreende as aplicações dos usuários que fazem parte de uma determinada organização virtual.

A Figura (b) apresenta outra proposta de arquitetura sob a forma de quatro camadas:

- Aplicação e serviços - Este nível é composto pelas ferramentas de desenvolvimento e aplicações para variados fins (variando em função do problema da organização virtual);
- Middleware - Fornece os protocolos que permitem transformar os elementos (servidores, ambientes de armazenamento, redes, etc.) em um ambiente de grade unificado;
- Recursos - Compostos pelos recursos da grade como por exemplo, um cluster ou um servidor primário;
- Rede - É a base da conectividade para os recursos da grade sendo formada por switches, roteadores e infra-estrutura das redes de comunicação.

2.3.1 OGSA - *Open Grid Services Architecture*

A OGSA é uma abordagem que visa auxiliar os desenvolvedores de aplicações, padronizando a forma de interoperar entre os serviços que irão utilizar o ambiente da grade (OGSA, 2008). As características do padrão OGSA podem ser resumidas na Figura 3.

O objetivo principal da arquitetura, representado no topo da Figura, é permitir o compartilhamento, o acesso e a gerência de recursos. Para prover essas funcionalidades a OGSA utiliza as facilidades de dois paradigmas: *web services* e protocolos de grade.

Figura 3: Características do padrão OGSA .

O termo *web services* descreve um paradigma de computação distribuída que difere de outras aproximações como DCE, CORBA e Java RMI, focando-se em padrões, da Internet convencional como, por exemplo, a *Extensible Markup Language*- XML. Em computação distribuída, os *web services* podem ser usados para descrever componentes de software, métodos para acessar esses componentes e métodos de descoberta que habilitem a identificação de provedores de serviços (XML, 2008). Vários padrões de *web services* tem sido definidos pelo *The World Wide Web Consortium* - W3C (W3C, 2008), mas três em especial merecem destaque (FOSTER et al., 2002):

- *Simple Object Access Protocol*(SOAP) (SOAP, 2008): É um procolo projetado para invocar aplicações remotas através de chamadas de procedimentos remotas (Remote Procedure Call, RPC) ou trocas de mensagens, em um ambiente independente de plataforma e linguagem de programação.

- *Web Services Description Language* (WSDL) (WSDL, 2008): É um documento xml que descreve o *web service*. Esse documento contém as interfaces, operações e esquemas de codificação para um *web service* específico.

- *WS-Inspection* (WS-INSPECTION, 2008): Utiliza convenções descritas em XML para localizar descrições de serviço publicadas por um provedor de serviço. Um documento WS-Inspection (WSIL) pode conter uma coleção de descrições de serviço e ligações à outras fontes descritoras. Uma descrição de serviço normalmente é uma URL que aponta para um documento WSDL. Ocasionalmente, essa descrição pode ser uma referência a um registro UDDI (*Universal Description, Discovery, and Integration*).

As características dos *web services* como a descoberta de serviços, a independência de linguagem de programação e de ambiente de software, bem como a utilização do SOAP, do WSDL e do WS-Inspection (DANTAS, 2005), permitem a OGSA oferecer a interoperabilidade entre serviços, independente de sua localização ou plataforma de software e hardware. Utilizando essas especificações, a OGSA define um mecanismo padrão para a criação, atribuição de nomes e descoberta persistente e transiente de instâncias de serviços da grade.

3 Gerenciamento em Grades Computacionais

Segundo (VOLCKAERT et al., 2004) e (GRAUPNER et al., 2003), uma arquitetura genérica de monitoramento para grades é formada por três componentes principais, responsáveis por descobrir, anunciar e alocar os recursos de forma eficiente. Essa arquitetura deve garantir que os nós da grade se comuniquem para coordenar a distribuição e alocação das tarefas, criando e mantendo um *pool* de recursos. As características de um sistema gerenciador da grade podem ser resumidas nas funcionalidades abaixo:

- Descoberta de Recursos: É necessário a existência de um repositório em cada nó da grade ou centralizado para publicar as informações de recursos. Dessa forma, essas informações poderão ser encontradas posteriormente;
- Modelagem de Recursos: Os recursos administrados pela grade são heterogêneos por natureza. Eles são de tipos diferentes, potencialmente distribuídos em locais geográficos e domínios administrativos diversos, podendo ser governados através de várias políticas. Modelar tais recursos é uma tarefa que inclui soma-los, transformá-los em outros, atualizá-los ou removê-los;
- Pedidos e garantias de recursos: Provedores de aplicação precisam especificar seus pedidos ao sistema de gerenciamento. A especificação do pedido pode ser feita em termos de métricas niveladas (por exemplo processamento, transações por segundo) ou através de uma linguagem de especificação de recurso *Resource Specification Language*, (RSL);
- Alocação e Distribuição de Recursos: São enviados pedidos de recurso ao sistema de gerenciamento que então empreende a reserva e distribuição dos mesmos. Essa reserva é terminada quando os recursos não são mais requeridos. É dever do sistema manter uma nota de todas as reservas competitivas e assegurar que o *pool* de recursos está sendo utilizado de acordo com suas políticas;

- Garantia nos Acordos: Quando ocorrem violações nos acordos de serviços (SLAs) , é importante que o sistema de gerenciamento entre com ações corretivas. Que podem variar da realocação de recursos à recuperação transparente da falha;

3.1 GRMS - *Grid Resource Management Systems*

Um GRMS envolve tarefas complexas como cuidar de questões envolvendo segurança, tolerância à falhas e escalonamento, além de gerenciar problemas de alocação, garantia, autenticação, autorização, auditoria e contabilidade de recursos. Recursos incluem ciclos de CPU, largura de banda, espaço de armazenamento e serviços como o de transferência de dados e simulação (NABRZYSKI; SCHOPF; WEGLARZ, 2004).

Em uma grade os usuários finais submetem uma tarefa ao GRMS. Este por sua vez irá analisar as especificações da tarefa e estimar os recursos requeridos (por exemplo número de processadores requeridos, quantidade de mémoria e tempo de execução). Depois de estimar esses recursos, o GRMS tenta descobrir a localização dos mesmos e selecioná-los. Finalmente, o escalonador de tarefas interage com o GRMS local para que executem a tarefa.

Dessa forma o GRMS deve interagir com os sistema de segurança para validar um usuário, com os serviços de informação para obter a descrição e disponibilidade de um recurso e com o sistema local do nó da grade para escalonar a tarefa.

Segundo (KRAUTER; BUYYA; MAHESWARAN, 2002) dependendo da sua arquitetura do GRMS (*Grid Resource management Systems*), pode apresentar as funcionalidades abaixo:

- Organização de Nome dos Recursos: Nesse contexto, o GRMS é responsável apenas por nomear e descobrir os recursos;
- Protocolo de Disseminação de Recursos: GRMS utiliza protocolos que controlam a quantidade de dados transferidos entre os sistemas e mantém o estado do banco de dados de informação de recursos;
- Modo Escalonamento: A estrutura do escalonador vai definir a estrutura do GRMS e a escalabilidade dos sistemas. Dessa forma, o GRMS pode ser; centralizado, no qual as tarefas são submetidas a um escalonador central simples; descentralizado, onde não existe um escalonador central e sim um grupo de escalonadores individuais que cooperam entre si; e hierárquico, no qual também existem vários escalonadores porém organizados em grupos por níveis.

3.2 GMA - *Grid Monitoring Architecture*

A GMA (*Grid Monitoring Architecture*) é um projeto de arquitetura de gerência para grades desenvolvido pelo *Global Grid Forum Performance Working Group* (GROUP, 2008), onde seu objetivo é servir como um guia de implementação e especificação das características críticas no monitoramento de desempenho em grades computacionais.

A arquitetura GMA é formada por três tipos de componentes mostrados na Figura 4 (TIERNEY, 2008) (MEDEIROS et al., 2003):

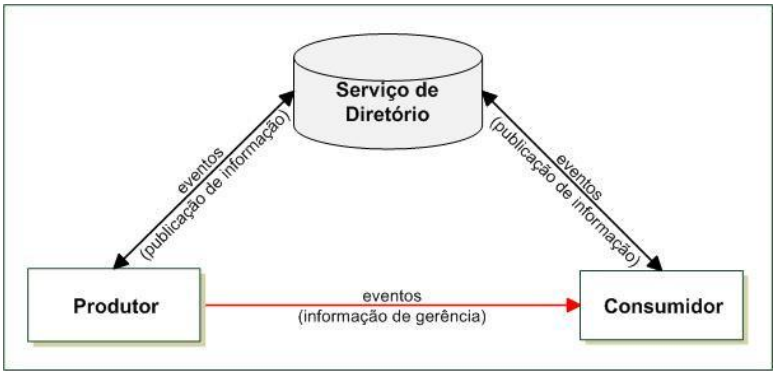

Figura 4: Arquitetura GMA (TIERNEY, 2008).

- Serviço de Diretório: local onde os produtores podem anunciar seus dados, e os consumidores podem anunciar suas necessidades;
- Produtor: disponibiliza a informação de gerência, como por exemplo, um sensor que reporta o status de determinado elemento continuamente;
- Consumidor: recebe as informações de gerência (eventos) de um produtor.

A arquitetura GMA conta com várias práticas de implementação, exemplos:

- MDS: *Monitoring and Discovery System*, presente no *Globus toolkit* V 4.0 (GLOBUS, 2008);
- R-GMA: Implementação em Java da arquitetura *Relational*-GMA (DATAGRID, 2008), construído para integrar o pacote 3 do projeto *Europen Data Grid*;
- NWS: *Network Weather Service*, opera com um conjunto de sensores distribuídos (monitores de rede, monitores de CPU, etc.) fornecendo estatísticas de Qualidade de Serviço (SERVICE, 2008);

- GrADS: *Grid Application Development Software Project.* Ferramenta interativa utilizada para o acesso fácil, manipulação e visualização de dados (GRADS, 2008).

4 A Pesquisa

Com o propósito de identificar quais os principais ambientes de grade utilizados no Brasil, foi apresentado à comunidade um questionário (apresentado no Anexo A). Esse questionário também levou em consideraçao dois focos principais: gerenciamento de configuraçao e o gerenciamento de falhas, identificando também as ferramentas utilizadas para prover gerência de recursos nesses ambientes.

O questionário foi enviado em 18 de janeiro de 2008 para 512 integrantes de grupos de pesquisas, fóruns nacionais e pesquisadores com publicações nos últimos três anos disponíveis nos principais meios da Internet sobre sistemas distribuídos, ficando ativo até 02 março de 2008. Foram obtidas 68 respostas. Em pesquisas similares, aplicadas mundialmente e que só levaram em consideração o gerenciamento de falhas como em (MEDEIROS et al., 2003) e (DUARTE et al., 2006) foram obtidas 22 e 13 repostas respectivamente. Tal resultado, além de evidenciar o interesse atual da comunidade brasileira sobre o assunto, demonstra que a pesquisa apresenta um bom panorama nacional, principalmente sobre o tratamento de falhas e a gerência de recursos em grades computacionais.

O questionário foi dividido em três blocos de questões: o primeiro bloco pretende identificar os principais ambientes de grades utilizados, o segundo bloco trata das questões de gerenciamento de recursos e o terceiro bloco trata das questões de gerenciamento de falhas. Esse último bloco será comparado com as pesquisas citadas anteriormente.

Para a confecção do questionário foi utilizado o *software LimeSurvey* (LIMESURVEY, 2008). O *LimeSurvey* é uma ferramenta escrita em *PHP Hypertext Preprocessor*, *open source*, destinado a automatizar o processo de geração, publicação e gerência de questionários. Utiliza o banco de dados *MYSQL*, para guardar as informações obtidas e manipular as estatísticas.

4.1 Metodologia

A metodologia para realizar a pesquisa foi constituída por três passos:

1. Passo: Pesquisa e cadastro dos possíveis entrevistados
2. Passo: Criação do questionário
3. Passo: Análise quantitativa dos dados
4. Passo: Análise qualitativa dos dados

Para realizar o primeiro passo, selecionar os candidatos à submissão do questionário, foram recolhidos nomes e emails das seguintes fontes:

- CNPQ - Conselho Nacional de Desenvolvimento Científico e Tecnológico. No site do CNPQ foram selecionados os pesquisadores pertencentes a grupos de pesquisas na área de sistemas distribuídos.
- SBC - Sociedade Brasileira de Computação. No portal da SBC foram encontradas as principais conferências, congressos e encontros nacionais sobre rede de computadores e sistemas distribuídos. Com os eventos selecionados foram visitados os anais dos últimos quatro anos de cada conferência. Selecionando os nomes e emails dos artigos que mencionaram a utilização de algum ambiente de grade.

Em paralelo ao primeiro passo da da metodologia, foi construído um questionário (ver Anexo A) com base nas pesquisas correlatas citadas no início deste capítulo e dos artigos analisados na fase inicial do trabalho. O passo da análise quantitativa (apresentado nas seções subsequentes) tem o intuito apenas de apresentar os dados recolhidos na pesquisa. Para facilitar a interpretação por parte dos leitores os dados serão ilustrado utilizando gráficos de barra, nos quais cada barra corresponde ao percentual de votos de uma dada opção referente a uma pergunta. O último passo da metodologia compreende a análise qualitativa dos dados (apresentada na seção 4.7) no qual foram especificado correlações entre as perguntas e separações entre os perfis dos entrevistados.

4.2 Perfil do Utilizador da Grade

Quando perguntados sobre qual perfil poderia ser enquadrado o envolvimento do seu trabalho ou pesquisa com tecnologias de grade computacional, foram obtidas as es-

tatísticas mostradas na figura abaixo, onde uma au mais opções poderiam ser escolhidas:

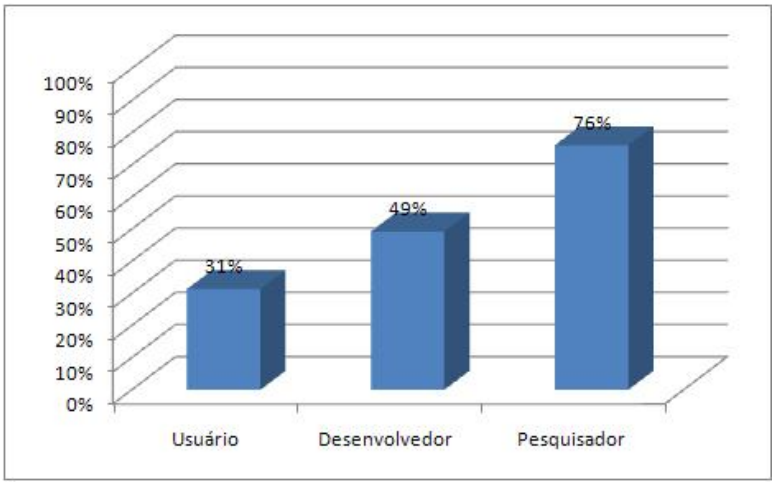

Figura 5: Perfil de Envolvimento.

4.3 Área de aplicação

Quando perguntados sobre em que área de atuação utilizavam as tecnologias de grade computacional, pergunta número dois do questionário, foram obtidas as seguintes estatísticas mostradas na figura abaixo, onde uma au mais opções poderiam ser escolhidas:

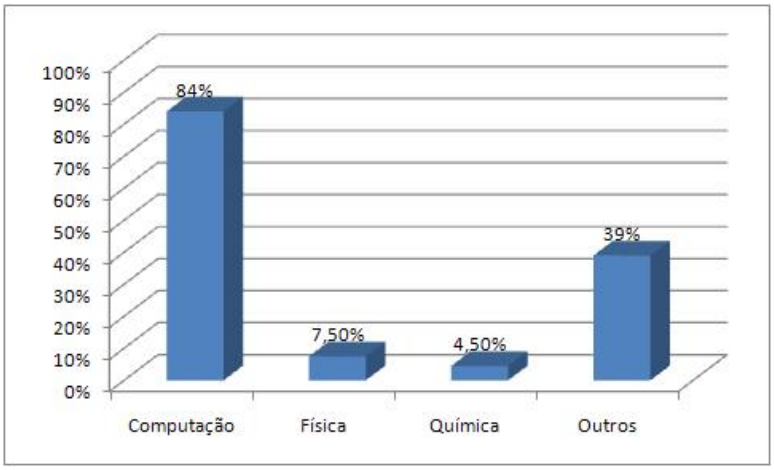

Figura 6: Área de aplicação.

Além das alternativas padrões, em outros, apareceram novas áreas não previstas no questionário como biotecnologia, biologia, métodos numéricos e neurociência.

4.4 Ambientes de Grade

Quando perguntados sobre qual ambiente de grade utilizam, foram obtidas as seguintes estatísticas mostradas na figura abaixo, onde mais de um ambiente poderia ser escolhido:

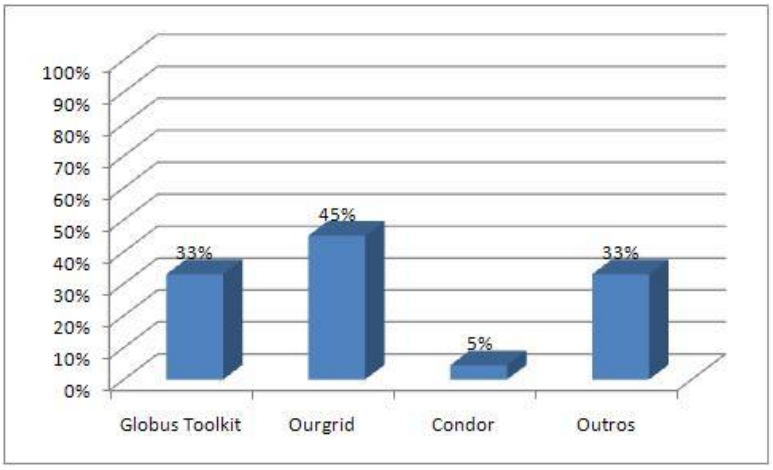

Figura 7: Ambientes de Grade.

Na opção "outros"foi identificado que os usuários utilizam também os ambientes Integrade e Join. As principais características de cada sistema serão apresentadas nas seções subsequentes. Posterior à essas descrições, será apresentado um quadro comparativo permitindo o confronto entre os ambientes de grade pesquisados.

4.4.1 Globus

O projeto Globus Toolkit (GT) vem sendo desenvolvido desde a década de 90, com o objetivo de suportar sistemas desenvolvidos em arquiteturas voltadas a serviços para o ambiente de computação em Grade. Os componentes do GT suportam diversas funcionalidades, como acesso, descoberta e gerenciamento de recursos, movimentação de dados, monitoramento do ambiente e questões de segurança (FOSTER, 2005).

A versão atual do Globus Toolkit 4 (GT-4) conta com intensa padronização fazendo uso de mecanismos como *Web Services* (WS), para definir interfaces e estruturas de comunicação. Esse middleware para o desenvolvimento de grades é licenciado pela *Globus Toolkit Public License* (GTPL 2) que permite a modificação e distribuição do código fonte.

4.4.1.1 Arquitetura do Globus

A Figura 8 ilustra a arquitetura do GT-4. Ela está dividida em linhas horizontais que representam respectivamente, o domínio do cliente, da segurança e autenticação bem como do servidor. No domínio do cliente, é oferecido um conjunto de bibliotecas que permitem a escrita de programas em Java, C e Python, permitindo ao usuário invocar operações sobre o GT-4. No domínio referente ao servidor, encontram-se os serviços responsáveis pela base da arquitetura. Esses serviços podem ser habilitados ou desabilitados conforme a necessidade de cada ambiente.

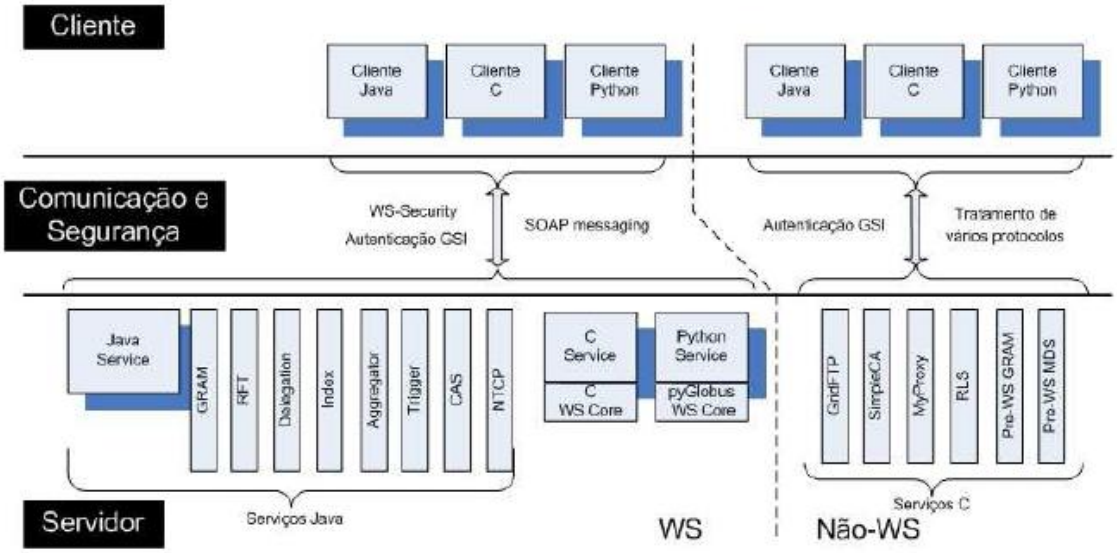

Figura 8: Arquitetura do GT4 (FOSTER, 2005).

Para administrar uma tarefa em um computador ou administrar um serviço, é necessário adquirir acesso a esse computador. Além disso, deve-se configurá-lo, organizar a execução e monitorar e administrar a computação resultante. O componente responsável por essas atividades é o GRAM (*Grid Resource Allocation and Management*). Ele oferece uma interface WS que administra computações arbitrárias em computadores remotos. Vários componentes compõem o GRAM:

- Componentes de protocolo;
- Componentes de *software*;
- *Softwares* de segurança;
- *Softwares* de gerenciamento de *jobs*;

- *Softwares* de gerenciamento de dados;
- *Softwares* de gerenciamento de tarefas.

Aplicações do GT-4 freqüentemente lidam com grandes quantidades de dados espalhados em vários locais. Coordenar todos esses dados é um problema. Para resolver essa questão foram desenvolvidas algumas soluções:

- GRID-FTP: Conjunto de bibliotecas e ferramentas, que permite a movimentação de grande quantidade de dados disco-a-disco;
- *Reliable File Transfer* (RTF): Provê administração segura na transferência entre múltiplos GridFTPs;
- *Replica Location Service* (RLS): Sistema escalável que mantém a informação de onde os arquivos estão replicados nas bases de dados;
- *Data Replication Service* (DRS): Combina RLS e GridFTP para a administração de réplica de dados;
- *Globus Data Access and Integration* OGSA-Dai: Permite acesso a informações relacionais descritas em XML;

4.4.1.2 Monitoramento e Descoberta de Recursos no Globus : *Monitoring and Discovering System (MDS)*

Monitoramento e descoberta são duas funções vitais em um sistema distribuído. Particularmente, quando esse sistema atravessa locais múltiplos, é difícil obter o detalhamento de todos os seus prováveis componentes. Através do monitorando é possível descobrir e diagnosticar os problemas que podem surgir em tais contextos, enquanto a descoberta permite identificar recursos ou serviços com propriedades desejadas. Ambas as tarefas requerem a habilidade para colecionar informação de fontes múltiplas e distribuídas. Os seguintes mecanismos são utilizados pelo GT-4:

- Descrição de recursos: Mecanismos padronizados, que descrevem as propriedades dos recursos em XML. O acesso a essas propriedades se dá por meio de *querys* ou subscrições. Esses mecanismos (basicamente especificados em WSRF e WS-Notification) são utilizados por todos os serviços e containers do GT-4 e podem ser incorporados aos serviços descritos pelos usuários;

- Serviços de *aggregator*: Colecionam dados sobre as fontes de informação registradas. Possuem um registro e um filtro de dados (*Trigger*). Como nem todas as fontes dão suporte ao WSRF e ao WS-Notification, os aggregators podem ser configurados para colecionar dados de qualquer fonte XML;

- Interfaces de vizualização: Baseadas em browser, ferramentas em linha de comando e WS interfaces, permitem os usuários examinar e acessar a informação reunida. Em particular, o serviço de WEBMDS pode ser configurado por transformações de XSLT (XSLT, 2008) para criar visões especializadas dos dados.

Esses mecanismos diferentes formam um poderoso framework (Figura 9) capaz de monitorar coleções diversas de componentes distribuídos e obter informação sobre componentes para propósitos de descoberta. O projeto *Earth System Grid* (ESG) (ESG, 2008), por exemplo, usa esses mecanismos para monitorar o estado dos serviços que utiliza e provê acesso a mais de 100 Tb de dados relacionados à pesquisas climáticas.

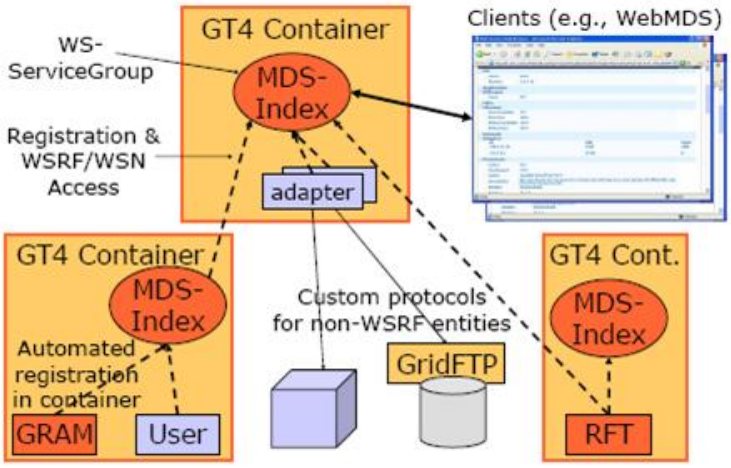

Figura 9: Arquitetura do MDS (FOSTER, 2005).

4.4.2 *Ourgrid*

O OurGrid (HP, 2008) é uma grade computacional *P2P*, aberta e cooperativa baseada em uma rede de favores, na qual os usuários doam seu poder computacional ocioso. vem sendo desenvolvida desde 2003 pela Universidade Federal de Campina Grande em conjunto com a HP-Brasil. Diferente do Globus e do Integrade (seção 4.4.4.1), o Ourgrid executa somente aplicações *bag-of-task*, onde suas aplicações executam paralelamente na grade, porém não se comunicam entre si. É um middleware interessante para rodar aplicações que fazem simulação, mineração de dados, busca massiva e renderização de imagens. O software é livre, com código aberto e licenciado pela *General Public License* (GPL/GNU) (OURGRID, 2008).

4.4.2.1 Arquitetura do *Ourgrid*

A Figura 10 ilustra a estrutura de rede de favores do OurGrid, no qual cada peer controla um conjunto de recursos de um site. Sites são estruturas que compõem a grade com interesse de trocar favores computacionais. Ao surgir uma demanda interna por recursos que o peer de um determinado site não consegue suprir, este peer irá fazer requisições à comunidade. A idéia é que os peers utilizem um esquema de prioridade baseado em quanto eles consumiram dos outros sites, ou seja, quanto mais se disponibiliza, mais poder computacional receberá em troca.

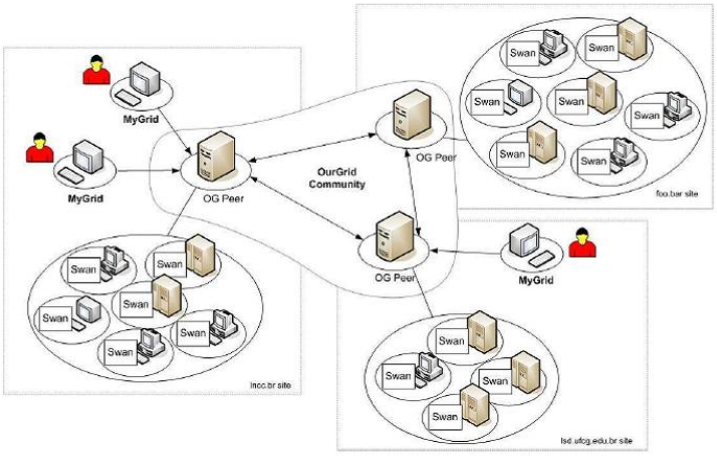

Figura 10: Arquitetura *Ourgrid*(OURGRID, 2008).

Basicamente, a arquitetura do Ourgrid é composta por três elementos (CIRNE; SANTOS-NETO, 2005):

- MyGrid Broker: Funciona como uma interface com o usuário (front-end), responsável por escalonar e coordenar o tráfego das tarefas;
- OurGrid Peer: Interage com os demais peers da grade, respondendo pelo seu domínio e alocando os recursos para o Mygrid;
- SandBoxing - Swan: Componente responsável pela segurança, tornando o ambiente confiável para execução das tarefas.

Para realizar a autenticação dos usuários, o Ourgrid oferece dois mecanismos. O primeiro, refere-se à autenticação do usuário que requer recursos em sua própria rede local. Quando isso acontece, a autenticação procede de uma forma tradicional (login e senha). Por outro lado, o OurGrid permite que o usuário tenha acesso aos GUMs (Grid Machines) que estão em outros sites. Esse acesso ocorre através do OurGrid Peer local , com o uso de certificados digitais X.509.

4.4.3 *Condor*

O sistema Condor foi desenvolvido na Universidade de Wisconsin, em Madison e possibilita a integração de diversas máquinas em aglomerados, para permitir a utilização eficaz dos recursos computacionais. Seu surgimento têm origem em 1988, sofrendo grandes mudanças até chegar a configuração atual (versão 7). O Condor tem como objetivo oferecer grande quantidade de poder computacional a médio e longo prazo. Tem como ponto forte a valorização da alta-vazão (*high throughput*) e não o alto desempenho (*high performance*)(CONDOR, 2008).

4.4.3.1 Arquitetura do *Condor*

A maneira básica de organizar máquinas participantes no sistema Condor é formar um aglomerado de máquinas (Condor *Pool*). Geralmente, as máquinas pertencentes a um aglomerado situam-se em um mesmo domínio administrativo, em um departamento de uma empresa ou uma rede local. Dentro do aglomerado, existem alguns módulos (THAIN; TANNENBAUM; LIVNY, 2005):

- *Schedd*: Permite que um usuário solicite execuções ao sistema Condor, portanto deve estar presente em todas as máquinas das quais deseja-se submeter aplicações para execução. Também, permite que o usuário monitore e controle remotamente a execução da aplicação;

- *Startd*: Permite que o sistema Condor execute aplicações na máquina em questão. Outra função do Startd é publicar periodicamente informação sobre o uso e disponibilidade de recurso da máquina;

- *Collector*: Módulo responsável por manter informações do aglomerado e receber requisições de execução. Periodicamente, cada *Startd* envia para o *Collector* informações sobre a disponibilidade de recursos. Da mesma maneira, quando um usuário solicita uma execução através do *Schedd*, este envia para o *Collector* uma requisição informando as necessidades da aplicação. O *Collector* também pode ser acessado por ferramentas externas de modo a fornecer informações sobre o funcionamento do aglomerado. Tipicamente existe um *Collector* em cada aglomerado;

- *Negotiator*: É o escalonador do aglomerado. Periodicamente, o *Negotiator* consulta o *Collector* de maneira a determinar se existem requisições de execução. Caso existam, o *Negotiator*, baseado nas informações fornecidas pelo *Collector*, emparelha requisições de execução com máquinas que possam atender tais requisições.

No Condor, tanto as requisições de execução de uma aplicação quanto as ofertas de recursos são definidas em termos de *ClassAds*, uma estrutura de dados nomeada e semelhante, em analogia, aos anúncios de classificados dos jornais. Um *ClassAd* é um conjunto de expressões que representam tanto a disponibilidade quanto a necessidade de recursos. As expressões são pares do tipo (atributo, valor) e podem incluir números, strings entre outras estruturas.

O processo de execução remota de aplicações nesse ambiente ocorre da seguinte forma: um usuário submete ao *Schedd* de sua máquina uma aplicação para execução remota, associada a um anúncio de pedido de recursos. Tal anúncio é enviado ao *Collector*. De maneira análoga, cada máquina que participa cedendo recursos anuncia periodicamente a disponibilidade dos mesmos, assim como quaisquer restrições a sua utilização.

Para trabalhar com grades, o Condor dispõe de 2 formas (EPEMA et al., 1996):

- *Flock of Condors*: É uma grade formada por vários Condor *Pools*. Esses *pools* se comunicam através de acordos de cooperação e de troca de recursos, onde não existe uma entidade centralizadora de pools. Para essa configuração, apenas é necessário instalar em cada pool um *gateway*. Esse *gateway* mantém a comunicação e torna transparente o acesso ao pool remoto;

- *Condor-G*: Utiliza recursos via Condor *Pools* e Globus, através de uma entidade centralizadora, o Condor-G *Scheduler*, que controla a execução das aplicações, submetendo-as para os pools ou para o Globus utilizando o GRAM e o MDS.

4.4.4 *Integrade*

O Integrade é um *middleware* que permite a implantação de uma grade utilizando recursos computacionais não dedicados, aproveitando o tempo ocioso de laboratórios já instalados. Por esse motivo é chamada de grade de computação oportunista, da mesma forma que o Ourgrid. É um projeto desenvolvido conjuntamente por pesquisadores de três instituições: Departamento de Ciência da Computação da Universidade de São Paulo, Departamento de Informática da Pontifícia Universidade Católica do Rio de Janeiro e Departamento de Computação e Estatística da Universidade Federal de Mato Grosso do Sul (INTEGRADE, 2008).

O Integrade possui arquitetura orientada a objetos, onde cada módulo do sistema se comunica com os demais a partir de chamadas à métodos remotos. Utiliza CORBA como sua infra-estrutura de objetos distribuídos. Nesse contexto, a arquitetura CORBA facilita a integração de módulos escritos nas mais diferentes linguagens, executando sobre diversas plataformas de hardware e software. Fornece também uma série de serviços úteis e consolidados, como os serviços de transações, persistência, nomes e *trading* (GOLDCHLEGER, 2004). O objetivo maior do Integrade é permitir o desenvolvimento de aplicações para resolver uma ampla gama de problemas paralelos (modelo BSP - *Bulk Synchronous Parallel* e modelo MPI - *Message Passing Interface*), em contraposição a diversos ambientes de grade que permitem seu uso apenas com aplicações independentes (*bag of task*) ou paramétricas.

4.4.4.1 Arquitetura do *Integrade*

A unidade estrutural básica de uma grade Integrade é o aglomerado (cluster). Tipicamente, o aglomerado explora a localidade de rede, ou seja, contém máquinas que estão próximas umas das outras em termos de conectividade. Entretanto, tal organização é arbitrária e os aglomerados podem conter máquinas presentes em redes diferentes. A Figura 11 apresenta os elementos típicos de um aglomerado Integrade:

O nó dedicado é uma máquina reservada a computação em grade. O nó compartilhado é aquele pertencente a um usuário que disponibiliza seus recursos ociosos à grade. Já o nó

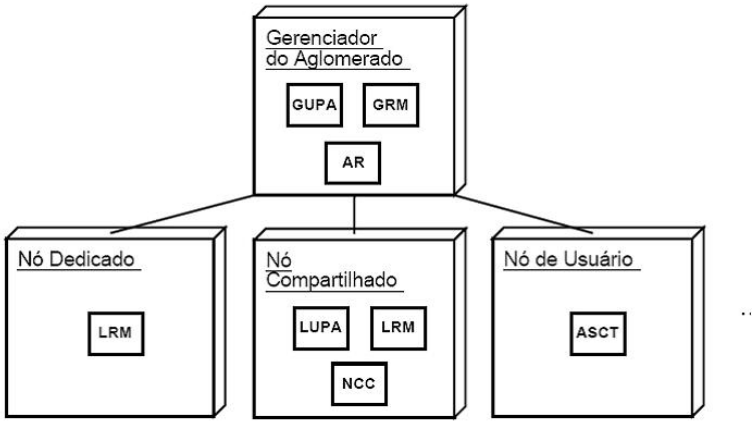

Figura 11: Arquitetura do *Integrade*.

de usuário é aquele que tem capacidade de submeter aplicações para serem executadas na grade. Finalmente, o Gerenciador de Aglomerado é o nó onde são executados os módulos responsáveis pela coleta de informações e escalonamento. Um nó pode pertencer a duas categorias simultaneamente, por exemplo, um nó que tanto compartilha seus recursos ociosos quanto é capaz de submeter aplicações para serem executadas na grade. Segundo (GOLDCHLEGER et al., 2004) o Integrade pode ser dividido nos seguintes módulos:

- *Local Resource Manager* (LRM): Componente executado em todas as máquinas que compartilham seus recursos com a Grade. Responsável pela coleta e distribuição das informações referentes à disponibilidade de recursos do nó em questão, além de exportar os recursos desse nó à Grade para permitir a execução e controle de aplicações submetidas por usuários;

- *Global Resource Manager* (GRM): Tipicamente executado no nó gerenciador do aglomerado. Atua como o serviço de informação, recebendo dos LRMs atualizações sobre a disponibilidade de recursos em cada nó do aglomerado, e como escalonador, ao processar as requisições para a execução de aplicações na grade. O GRM utiliza as informações de que dispõe para escalonar aplicações aos nós mais apropriados, conforme requisitos estabelecidos para execução das mesmas;

- *Node Control Center* (NCC): Executado nas máquinas compartilhadas, permite que o proprietário da máquina controle o compartilhamento de seus recursos com a Grade. Atuando conjuntamente com o LRM, permite que o usuário defina períodos em que os recursos podem ou não ser utilizados, independente de estarem disponíveis;

- *Application Submission and Control Tool* (ASCT): Permite que um usuário submeta aplicações para serem executadas na Grade. O usuário pode estabelecer requisitos, como plataforma de hardware e software ou quantidade mínima de recursos necessários à aplicação, além de preferências, como a quantidade de memória necessária para a aplicação executar com melhor desempenho;

- *Local Usage Pattern Analyzer* (LUPA): É responsável pela análise e monitoramento dos padrões de uso. A partir das informações periodicamente coletadas pelo LRM, o LUPA armazena séries de dados e aplica algoritmos de clustering de modo a derivar categorias comportamentais do nó. Tais categorias serão utilizadas como subsídio às decisões de escalonamento, fornecendo uma perspectiva probabilística de maior duração sobre a disponibilidade de recursos em cada nó;

- *Global Usage Pattern Analyzer* (GUPA): Auxilia o GRM nas decisões de escalonamento ao fornecer as informações coletadas pelos diversos LUPAs;

- *Application Repository* (AR): Armazena as aplicações a serem executadas na Grade. Através do ASCT, o usuário registra a sua aplicação no repositório para posteriormente requisitar sua execução.

Os componentes do aglomerado Integrade se comunicam através de dois protocolos: o Protocolo de Disseminação de Informações e o Protocolo de Execução de Aplicações. O primeiro protocolo permite que o GRM mantenha informações espalhadas pelas máquinas do aglomerado. Essas informações podem ser estáticas (arquitetura de máquina, versão do sistema operacional, quantidade total de memória, etc.) ou dinâmicas (porcentagem de CPU ociosa, quantidade disponível de memória, etc.). Já o protocolo de execução de aplicações permite que um usuário da grade submeta aplicações para execução sobre recursos compartilhados.

Como pretende ser um sistema escalável, o Integrade permite soluções para interligar diversos aglomerados. A primeira solução propõe uma hierarquia de aglomerados. Nessa abordagem, os GRM de diversos aglomerados são integrados em uma hierarquia arbitrária, definida em conjunto pelos administradores dos aglomerados. O protocolo de disseminação de informação é alterado da seguinte maneira: periodicamente, cada GRM envia ao seu superior na hierarquia uma consolidação, (como por exemplo a média e o desvio padrão) da disponibilidade de recursos nos nós de seu aglomerado. A segunda solução utiliza uma abordagem *peer-to-peer* para a interligação dos aglomerados utilizando uma tabela de

espalhamento distribuída (*Distributed Hash Table*, DHT). As entradas dessa tabela são utilizadas para identificar alternativas de conexão entre GRMs (GOLDCHLEGER, 2004).

4.4.5 Join

O projeto Join começou a ser desenvolvido em 1996, na Faculdade de Engenharia Elétrica e de Computação da Unicamp. A versão atual (1.3) caracteriza-se por ser uma plataforma extensível baseada em serviços/componentes, possuindo um escalonador sob-demanda e modelos de administração remota (JOIN, 2008). É uma plataforma altamente portável, já que pode ser executado em qualquer computador que disponha de uma implementação da Máquina Virtual Java. O Join permite o processamento paralelo baseado na ampla disponibilidade de interfaces simples para acesso à recursos da Internet. Para participar da grade, um computador só precisa estar conectado a Internet e utilizar o navegador para executar *applets* em Java. Dessa forma, o usuário se conecta a um servidor *web* determinado e declara que o seu computador está disponível por determinado período de tempo. Segundo (YERO, 2003), os principais objetivos do Join são:

- Oferecer aos programadores e usuários de sistemas paralelos uma plataforma simples e versátil para a implementação e uso de aplicações paralelas que tenham baixas taxas de comunicação;
- Permitir a implementação de aplicações paralelas que usem de forma transparente computadores com arquiteturas heterogêneas;
- Permitir que qualquer computador com acesso aos recursos da *Internet* possa participar do processamento paralelo;
- Não usar um computador ligado à *Internet* para o processamento paralelo sem que seu dono o permita explicitamente;
- Não exigir do dono do computador participante a instalação de softwares específicos para o processamento paralelo, tal como ocorre com PVM e MPI;
- Recompensar de alguma forma àqueles que coloquem seus computadores à disposição do sistema paralelo virtual;
- Garantir a segurança dos dados nos computadores participantes no sistema;
- Ser escalável a um grande número de computadores;

- Ser tolerante a falhas tanto nos computadores participantes quanto nas conexões de rede.

4.4.5.1 Arquitetura do *Join*

Uma aplicação em Join é formada inicialmente por uma tarefa, que é descarregada em um dos computadores participantes pelo servidor de aplicações. Esta tarefa pode criar novas tarefas durante a sua execução sem a intervenção do servidor (Figura 12). As tarefas são identificadas por um identificador de tarefas (TID). Se o dono do computador decide retirar o seu computador do sistema, as tarefas que estão sendo executadas nele devem ser interrompidas imediatamente e reiniciadas em outro computador.

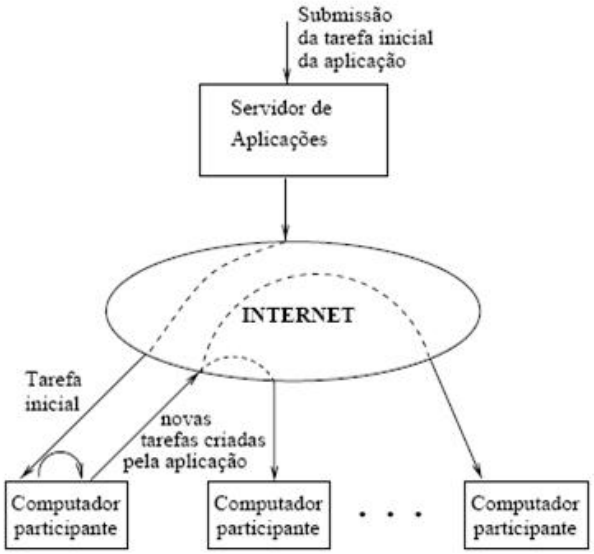

Figura 12: Modelo de aplicação submetida ao *Join* (YERO, 2003)

Basicamente, a arquitetura do JOIN é composta por quatro tipos distintos de componentes: servidor, coordenador, trabalhador e administração (*jack*) (YERO, 2003):

- Servidor: Tem como objetivo gerenciar as operações de mais alto nível da plataforma. Alguns exemplos de operações deste tipo são: iniciar a execução de uma aplicação, receber novos computadores que desejem integrar-se à plataforma ou fornecer informações globais a respeito do funcionamento da mesma;

- Coordenador: É o responsável pelo gerenciamento das atividades de computação realizadas em cada um dos grupos da plataforma. Além disso, é responsável também

pela comunicação dos grupos com entidades externas a eles. Cada grupo admite somente um coordenador;

- Trabalhador: É o responsável pela realização do trabalho computacional útil, ou seja, pela execução das tarefas de aplicações. Conjuntos de trabalhadores formam com o coordenador os grupos da plataforma;

- Módulo de Administração: Também chamado de *JoiN Administration and Configuration Kit* (Jack), tem a função de auxiliar a configuração e a operação da plataforma por parte de seus administradores. Para isso, mantém-se conectado ao componente servidor, a quem pode submeter ordens específicas e/ou buscar informações para o monitoramento da plataforma.

Cada computador participante na plataforma JOIN executa um destes componentes que, dependendo de sua função, poderá estar presente em um ou muitos computadores simultaneamente.

4.4.6 Quadro Comparativo sobre as Tecnologias de Grade

Após o detalhamento de cada arquitetura dos ambientes de grades encontrados na pesquisa, foi possível construir um quadro comparativo, apresentado abaixo, entre as diversas funcionalidades dos sistemas.

Ambiente	Join	Integrade	Condor	Ourgrid	Globus
Sistema de Grade	Grade de Supercomputação	Grade Oportunista, alto processamento	Grade de Supercomputação	Grade Oportunista, alto processamento	Grade de Serviço, Sob demanda
Tecnologia	Java	Corba	Perl, C	Java	Web-Services
Clientes	Java	Java	Java, C, Perl, Pyton	Java	Java, C, Pyton
Descoberta de recursos	Não	Não	Não	Sim Auto-man	Sim (WSRF - UDDI)
Identificação de Recursos	Não	Sim GRM, LRM	Sim ClassAd	Sim OurGrid Peer	Sim Ws-resource
Gerenciamento de Recursos	Não	Sim GUPA	Sim Collector	Sim Auto-man	Sim MDS
Escalonamento de Jobs	Sim Jack	Sim GRM	Sim Negotiator	Sim MyGridbroker	Sim GRAM
Segurança	Sim Autenticação Login/Senha	Sim Assinatura Digital	Sim Criptografia, Kerberos	Sim SandBox/Swan, Certificado Digital	Sim Certificado Digital
Tolerância Falhas	Sim Checkpoints	Sim Checkpoints	Sim Checkpoints	Sim Grid Information Services	Sim Checkpoints

Tabela 1: Quadro Comparativo Tecnologias de Grade.

4.5 Gerenciamento de Falhas

No segundo bloco, as questões solicitavam a opinião dos participantes sobre os seguintes aspectos: espécies de falhas, mecanismos de tratamento de falhas, recuperação de falhas e grau de envolvimento do usuário. Através dos dados coletados, realizamos comparações com os resultados obtidos em (MEDEIROS et al., 2003) e (DUARTE et al., 2006).

4.5.1 Espécies de Falhas

Segundo a pesquisa atual, as falhas mais encontradas nos sistemas de grade computacional (Figura 13) estão relacionadas com a configuração do sistema (74%), falhas de aplicação (37%), falhas no middleware (35%) e falhas de hardware (6,1%). As falhas de

configuração do sistema também foram apontadas com maior freqüência nas pesquisas de 2003 (76%) e 2005 (60%). Todos os outros tipos de falhas na pesquisa atual demonstraram uma queda em relação aos valores apresentados nas pesquisas anteriores, onde uma ou mais opções poderiam ser escolhidas.

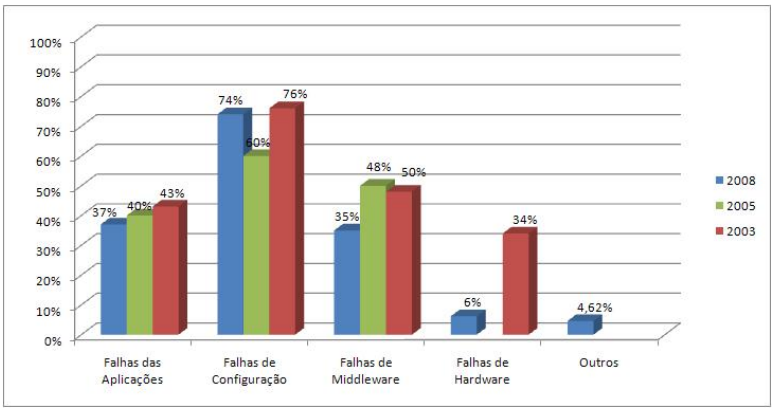

Figura 13: Espécies de falhas.

4.5.2 Mecanismos de Tratamento de Falhas

Em 2003, a maior incidência de mecanismos para o tratamento de falhas foram os mecanismos dependentes de aplicação (57%). Esse resultado indicou, até aquele ano, um baixo emprego de métodos automatizados para tratamento de falhas nos sistemas de grade computacional. Nessa mesma pesquisa, onde uma ou mais opções poderiam ser selecionadas, tanto o uso de sistemas de monitoramento quanto de mecanismos baseados em ponto de restauração estiveram presentes em 29% dos casos, enquanto o escalonamento tolerante a falhas aparecia apenas em 19% das respostas.

Em 2005, essa situação foi invertida, o escalonamento tolerante a falhas foi o mais apontado em 57% dos casos, enquanto o uso de mecanismos dependentes de aplicação diminuiu para 40% das respostas submetidas.

Em 2008, levando em consideração os mecanismos de tratamento de falhas apresentados pela Figura 14, observarmos que os sistemas de monitoramento (38%) estão entre as opções mais utilizadas, junto com o escalonamento tolerante a falhas (38%), as soluções dependentes de aplicação (37%) e os mecanismos baseados em ponto de restauração (25%). Destacamos também o crescimento de métodos de tratamento de falhas que em 2003 e 2005 ainda não estavam bem consolidados, como os testes automáticos. Embora as soluções de visualização (20%) e de testes automáticos (15%) sejam promissoras, ambas contam ainda com poucas implementações e precisam ser amadurecidas.

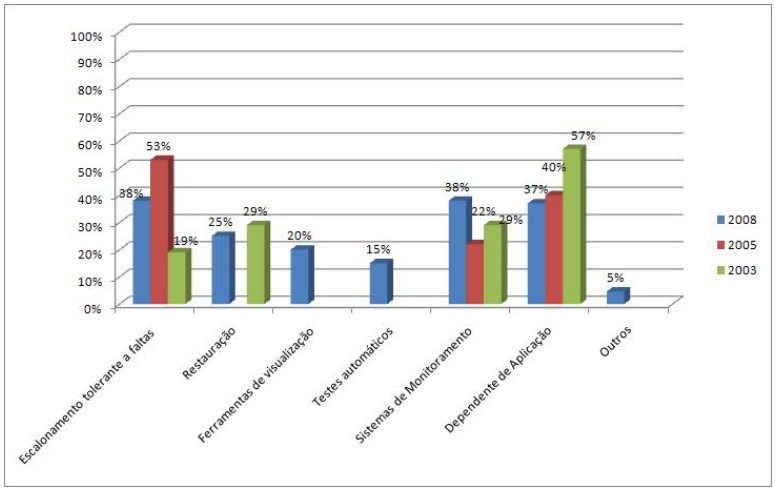

Figura 14: Mecanismos de Tratamento de Falhas.

4.5.3 Recuperação de Falhas

Nesse ponto, o maior problema dos últimos cinco anos continua sendo o diagnóstico da falha, ou seja, a identificação da sua causa principal. Apesar de compor 54% das respostas, como pode ser observado na Figura 15, nota-se um decrescimento em relação a 2003 (71%) e 2005 (61%), fato que ocorre simultaneamente com o crescimento do uso de métodos para o diagnóstico de faltas, como testes automáticos. O problema envolvido em obter autorização para corrigir a falha apareceu em 13% dos casos, pouco menor que em 2003 (14%).

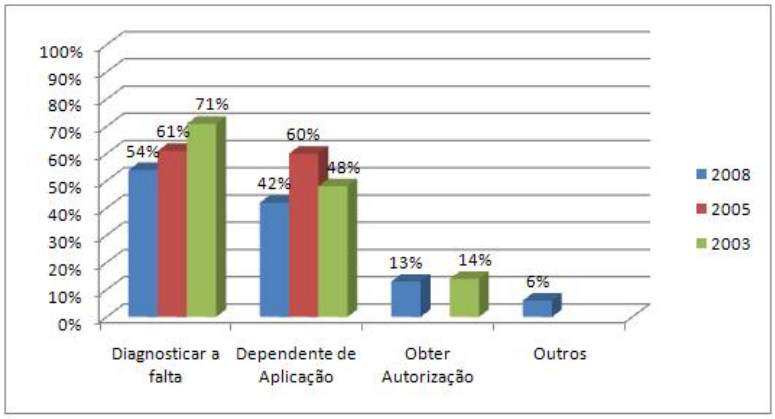

Figura 15: Recuperação de Falhas.

4.5.4 Grau de Envolvimento do Usuário

Nossa pesquisa apontou que ainda é alto o envolvimento do usuário no processo de recuperação de uma falha (Figura 16), onde é necessário que ele defina o que deve ser feito para a recuperação do sistema (34%). Por outro lado, detectamos uma melhora do quadro geral, visto que em 2003 essa porcentagem correspondia à maioria absoluta dos casos (58%). Em 2008, a maioria dos usuários estão envolvidos apenas marginalmente (38%) ou não precisam ser envolvidos porque o sistema oferece recuperação automática (28%). Em 2003, esses dados correspondiam respectivamente a 29% e 13% dos casos.

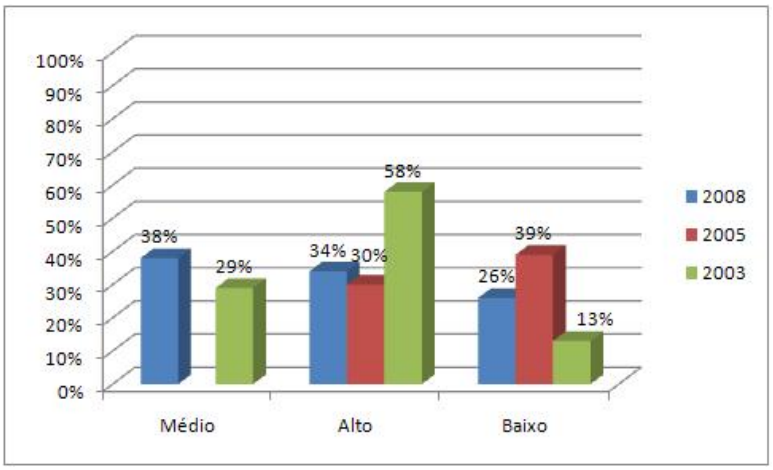

Figura 16: Grau de Envolvimento do Usuário.

4.6 Gerenciamento de Configuração e Recursos do Ambiente de Grade

As questões elaboradas buscaram levantar as ferramentas e as dificuldades envolvidas para gerenciamento do sistema de grade, requerendo a opinião dos participantes sobre seguintes aspectos: aplicações utilizadas no monitoramento, gerenciamento de recursos e esforço empregado na manutenção do ambiente.

4.6.1 Aplicações Auxiliares no Monitoramento de Grades Computacionais

A pesquisa apontou a utilização de aplicações auxiliares que estão sendo usadas para facilitar o monitoramento do ambiente de grade computacional. Na maioria dos casos,

elas permitem uma visualização gráfica de *logs* e comparações estatísticas, bem como informações sobre o estado de determinado elemento da grade.

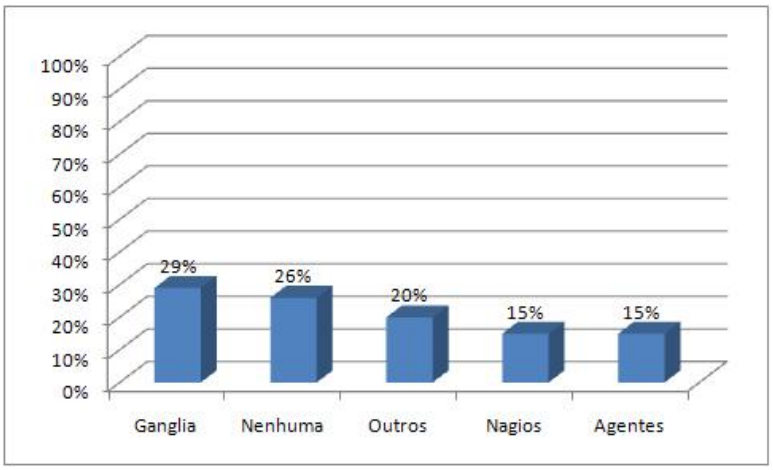

Figura 17: Aplicações Auxiliares no Monitoramento de Grades Computacionais.

Na Figura 17 podemos observar que a maioria das respostas (29%) apontam o Ganglia (GANGLIA, 2008) como a ferramenta mais usada, enquanto 20% preferem outras ferramentas, 15% utilizam o Nagios (NAGIOS, 2008) e 15% utilizam ferramentas baseadas em agentes. Porém, grande parte dos usuários [1](26%) não utilizam qualquer uma delas.

Em outras ferramentas, um ou mais usuários apontaram o uso do AutoMan (AUTOMAN, 2008), Monitoring and Discovery System (MDS) (GLOBUS, 2008), Network Weather Service (NWS) (SERVICE, 2008), NetLogger (NETLOGGER, 2008), Scali Manage (SCALI, 2008), bem como outras aplicações e componentes presentes no próprio *middleware* da grade computacional.

4.6.2 Gerenciamento de Recursos

Do total de repostas obtidas durante a pesquisa, pouco mais de 18% afirmaram possuir em seu ambiente de grade um Grid Resource Management System (GRMS), o que demonstra algum avanço na questão de centralização do gerenciamento de recursos auxiliando no desenvolvimento das economias de grades.

Na Figura 18, pode ser observado os gerenciadores de recursos, onde 35% proporcionam gerenciamento de alocação, 34% gerenciamento de disponibilidade, 23% gerencia-

[1]Nesta pesquisa, não estamos distinguindo os usuários por sua relação com o sistema de grade computacional: desenvolvedores, administradores, ou apenas usuários finais do sistema. Dependendo da natureza do sistema ou da forma que é projetado, usuários finais costumam ter pouco ou nenhum envolvimento com as ferramentas de gerenciamento do ambiente.

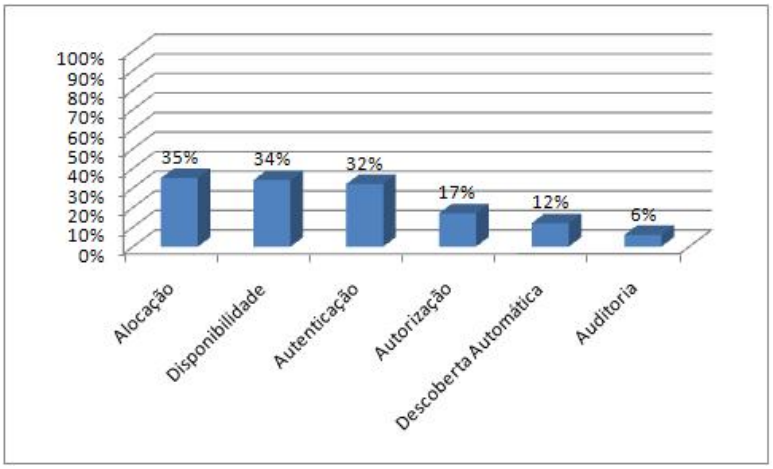

Figura 18: Gerenciamento de Recursos.

mento de autenticação, 17% gerenciamento de autorização, 12% descoberta automática de recursos e 6% auditoria, podendo o GRMS oferecer um ou mais desses recursos.

4.6.3 Esforço na Manutenção e Gerência do Ambiente de Grade

A maioria dos usuários (60%), conforme apresentado na figura 19, afirma que, após as configurações iniciais do sistema de grade computacional, os ambientes utilizados oferecem algum nível de facilidade de manutenção, com emprego de algumas ferramentas de gerenciamento.

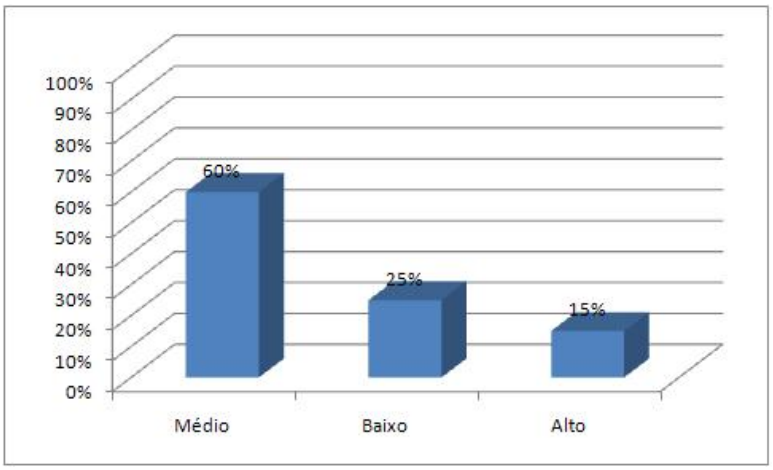

Figura 19: Esforço na Manutenção e Gerência do Ambiente de Grade.

Nessa mesma questão, em 25% das respostas os usuários indicam que não enfrentam nenhum problema em manter seus ambientes, e apenas em 15% dos casos é colocado que o sistema oferece pouca ou nenhuma facilidade de gerenciamento.

4.7 Análise Qualitativa dos Resultados

Nessa seção será feita uma interpretação dos dados obtidos com a aplicação do questionário, partindo da identificação do perfil dos entrevistados (apresentados na seção 4.2).

O perfil de usuário obteve 31% das respostas (vinte e um votos), o perfil de desenvolvedor 44% (trinta e três votos)e o perfil de pesquisador 76% (cinquenta e um votos). Dentro dessa porcentagem, dos 68 entrevistados, e por se tratar de uma questão onde mais de uma alternativa poderia ser escolhida é necessário frisar que (Figura 20):

- Cinco entrevistados se classificam apenas como usuário;
- Vinte e oito entrevistados se classificam apenas como pesquisador;
- Cinco entrevistados se classificam apenas como desenvolvedor;
- Quatro entrevistados se classificam como usuários e desenvolvedores;
- Três entrevistados se classificaram como usuários e pesquisadores;
- Catorze entrevistados se classificaram como desenvolvedores e pesquisadores;
- Nove entrevistados se classificaram nas três categorias (usuário, pesquisador e desenvolvedor).

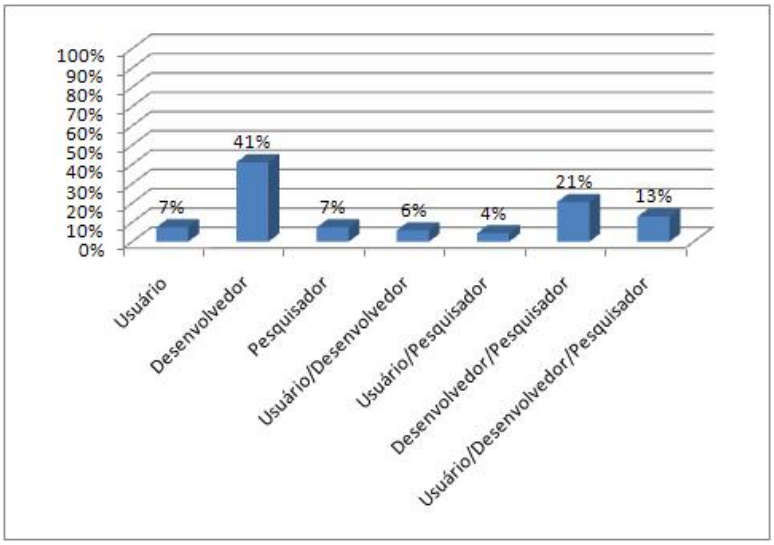

Figura 20: Análise qualitativa do perfil dos usuários.

Dentro do perfil de usuário foi possível identificar as seguintes correlações:

- Principal área de atuação foi a área de computação;
- O ambiente *Ourgrid* obteve a maioria de votos (doze), o ambiente *Globus* em segundo lugar com sete votos, o ambiente *Integrade* terceiro lugar com dois votos e o ambiente *Join* obteve um voto (Figura 21);

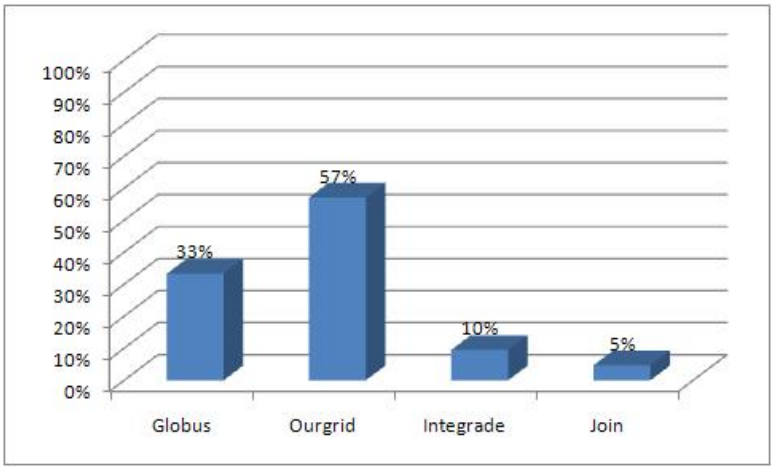

Figura 21: Ambientes de grades utilizados pelo perfil de usuário.

- O nível de dificuldade que predominou na implantação do ambiente de grade foi o nível de dificuldade médio (dezesseis votos), ficando o nível de dificuldade alto com quatro votos e o nível de dificuldade baixo com um voto (Fogura 22);

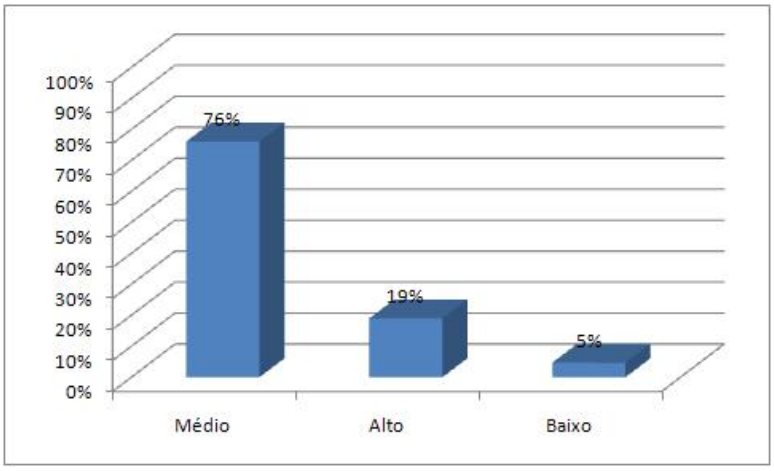

Figura 22: Dificuldade na implantação do ambiente de grade - Perfil de usuário.

- O esforço na manutenção do ambiente de grade que predominou foi o esforço médio (onze votos), o esforço baixo obteve oito votos e esforço alto dois votos (23);

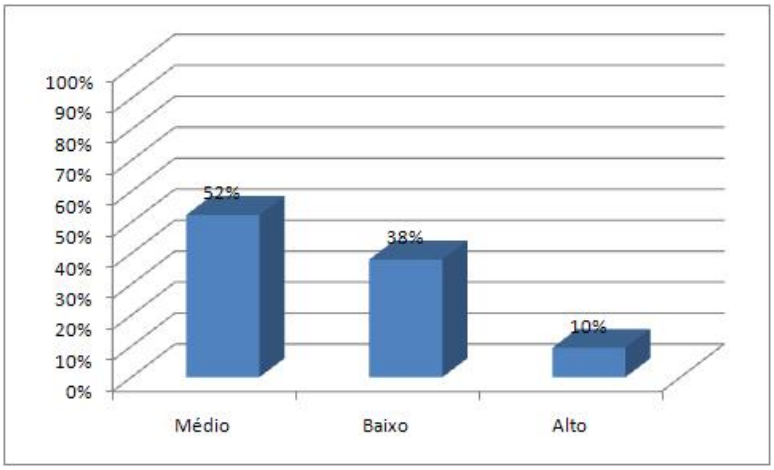

Figura 23: Esforço na manutenção do do ambiente de grade - Perfil de usuário.

- O tipo de falha mais frequente encontrada pelo perfil de usuário foram as falhas relacionadas a aplicação, com treze votos.

Dentro do perfil de desenvolvedor foi possível identificar as seguintes correlações:

- Principal área de atuação foi a área de computação;
- O ambiente *Globus* obteve maioria de votos (dez), o ambiente *Join* em segundo lugar com 8 votos, o ambiente *Integrade* e *Ourgrid* em terceiro lugar com sete votos respectivamente e o ambiente *Condor* obteve um voto (Figura 24);

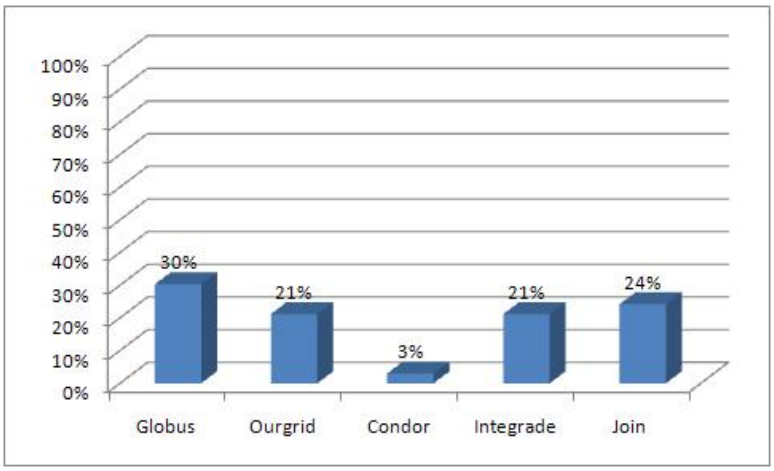

Figura 24: Ambientes de grades utilizados pelo perfil de desenvolvedor.

- O nível de dificuldade que predominou na implantação do ambiente de grade, foi o nível de dificuldade médio (vinte e cinco votos), ficando o nível de dificuldade alto com seis votos e o nível de dificuldade baixo com dois votos (Figura 25);

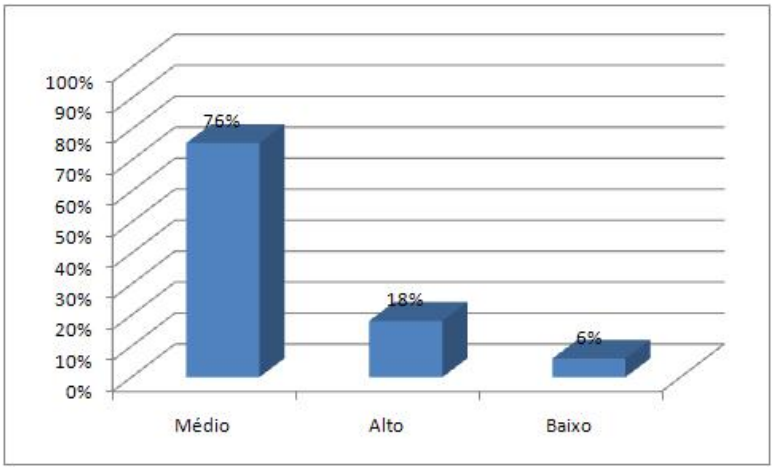

Figura 25: Dificuldade na implantação do ambiente de grade - Perfil de desenvolvedor.

- O esforço na manutenção do ambiente de grade que predominou foi o esforço médio (dezoito votos), o esforço baixo obteve nove votos e o esforço alto seis votos (Figura 26);

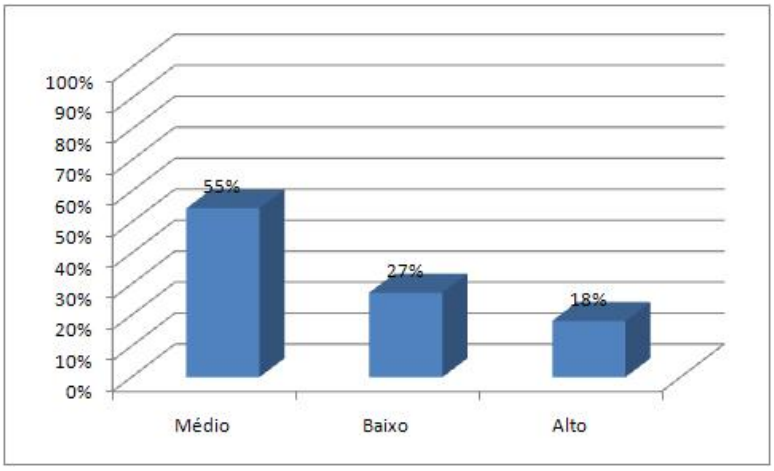

Figura 26: Esforço na manutenção do ambiente de grade - Perfil de desenvolvedor.

- O tipo de falha mais frequente encontrada pelo perfil de desenvolvedor foram as falhas relacionadas a configuração com vinte e quatro votos.

Dentro do perfil de pesquisador foi possível identificar as seguintes correlações:

- Principal área de atuação foi a área de computação;
- O ambiente *Ourgrid* obteve maioria de votos (dezenove), o ambiente *Globus* em segundo lugar com dezessete votos, o ambiente *Integrade* em terceiro lugar com

oito votos e os ambientes *Condor* e *Join* com obtiveram três votos respectivamente (Figura 27);

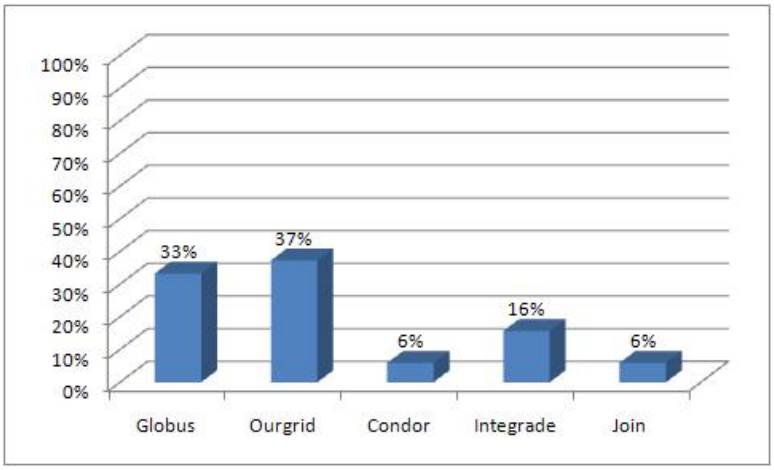

Figura 27: Ambientes de grades utilizados pelo perfil de pesquisador.

- O nível de dificuldade que predominou na implantação do ambiente de grade, foi o nível de dificuldade médio (quarenta e dois votos), ficando o nível de dificuldade alto com oito votos e o nível de dificuldade baixo com um voto (Figura 28);

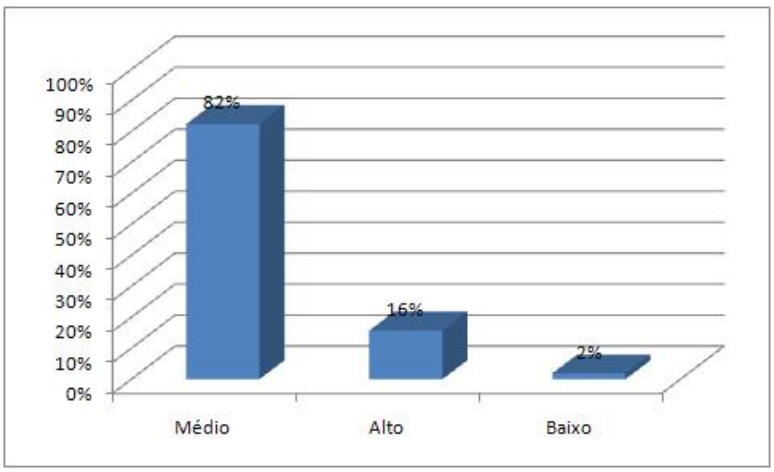

Figura 28: Dificuldade na implantação do ambiente de grade - Perfil de pesquisador.

- O esforço na manutenção do ambiente de grade que predominou foi o esforço alto (trinta e quatro votos), ficando o esforço baixo com dez votos e o esforço médio com sete votos (Figura 29);

- O tipo de falha mais frequente encontrada pelo perfil de pesquisador ,assim como no perfil de desenvolvedor, foram as falhas relacionadas a configuração, com trinta e oito votos.

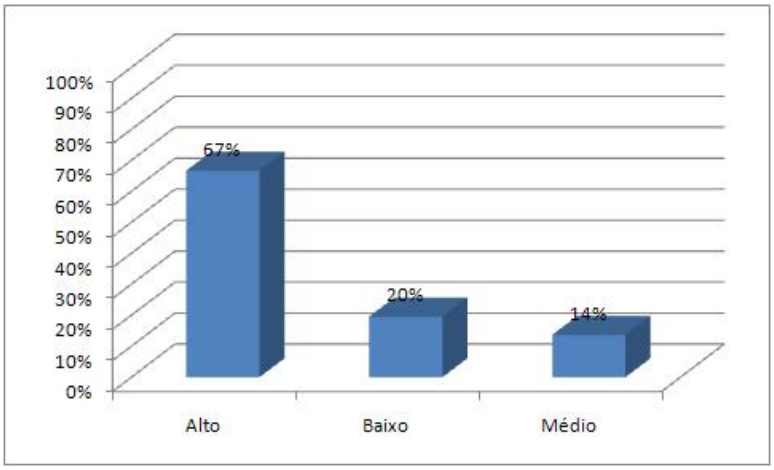

Figura 29: Esforço na manutenção do ambiente de grade - Perfil de pesquisador.

Nos ambientes de grade listados na seção 4.4 é possível distinguir quais os maiores problemas enfrentados no processo de recuperação de uma falha.

- Para o ambiente *Globus* e *Integrade* o maior problema, listados por seus utilizadores, foi o relacionado em criar uma aplicação que implementasse um comportamento especifico de recuperação;
- Para o ambiente *Ourgrid*, *Condor* e *Join* o maior problema listado foi relacionado à realização do diagnóstico da falta detectada;

O problema relacionado à ganhar autorização para corrigir componentes defeituosos não foi citado pelos utilizadores dos ambientes *Join*, *Condor* e *Integrade*. Aparecendo em poucos casos nos ambientes *Globus* e *Ourgrid*.

Relacionando esses ambientes de grade com as aplicações auxiliares nota-se que a maioria dos usuários não utiliza qualquer ferramenta, seção 4.6.1, porém os utilizadores do *Globus* preferem utilizar as ferramentas *NetLogger* e *Ganglia*. Os utilizadores do *Ourgrid* citaram apenas o *Ganglia* e os utilizadores dos outros ambientes não citaram o uso de nenhuma ferramenta de monitoração, apenas as existentes no próprio *middleware* da grade.

Pelos resultados apresentados é notório que o percentual de utilizadores dos ambientes de grades no Brasil se concentra no perfil de pesquisador. Estes exercem atividades de testes dos ambientes, comparações e procuram finalidades e aplicações para executar na grade. Na maioria dos casos são também desenvolvedores de funcionalidades para os sistemas que utilizam. Considerando todos os perfis, observa-se que o processo de instalação

e de configuração inicial é trabalhoso tanto para usuários quanto para desenvolvedores ou pesquisadores. Para esses últimos ,em especial, o processo de manter e atualizar o ambiente é considerado alto o que revela o envolvimento custoso desse perfil na constante construção e investigação do ambiente de grade utilizado.

5 Conclusão

De um modo geral, as tecnologias para o gerenciamento de grades computacionais cresceram nos últimos anos (Capítulo 3). Problemas antes sem tratamento, como a possibilidade de diagnosticar as falhas detectadas, contaram com propostas de soluções como as de (DUARTE et al., 2006).

Pelo número de alternativas e padrões apresentados no capítulo 3, fica evidente o esforço da comunidade na adoção de novos paradigmas, como aqueles adotados pelo OGSA. Também é notório o trabalho dos desenvolvedores na melhora continua das questões de gerenciamento nos ambientes de grade, integrando camadas próprias para tal finalidade nas arquiteturas dos sistemas.

Entretanto, como levantado nesse trabalho, desde 2003 até os dias atuais, a maior origem de falhas em grades ainda está relacionada diretamente com os problemas ligados à configuração do ambiente. Embora as tecnologias de grade estejam sofrendo um forte impulso nos últimos anos e recebendo diversas contribuições para seu amadurecimento, sua natureza altamente distribuída envolve diversos problemas no tratamento e configuração dos recursos.

Por exemplo, uma das funcionalidades principais para virtualização dos serviços de computação em grade é a seleção dos recursos necessários e adequados para execução das tarefas. Porém, a seleção de recursos ainda apresenta desafios para escolha, principalmente quando são considerados diversos fatores tais como o desempenho esperado para executar uma aplicação, a restrição de acesso ao recurso, o custo de execução da aplicação no recurso e a confiabilidade do recurso.

Neste contexto, os GRMS estão sendo empregados nas infra-estruturas de computação distribuída de larga-escala para facilitar o desenvolvimento de sistemas de gerenciamento, buscando resolver desafios como: balanceamento de carga, gerenciamento de tarefas e movimentação de dados. Tais sistemas permitem o desenvolvimento de funcionalidades, no nível de usuário, com uma visão mais abstrata das camadas subjacentes em grades

complexas, oferecendo: descoberta e seleção dinâmica de recursos, arquitetura de controle de avaliação, métodos de mapeamento e escalonamento avançados, podendo ainda obter suporte de outros serviços de *middleware*.

Por fim, analisando a evolução das pesquisas, descobrimos uma diminuição gradual da dificuldade de manutenção destes ambientes pela coexistência e emprego de diferentes métodos de tratamento de falhas, sistemas de monitoramento, ferramentas auxiliares e GRMS, onde este conjunto tem contribuído para apresentar uma visão mais transparente das funcionalidades da grade aos seus usuários.

Referências

AUTOMAN. 06 2008. Disponível em: http://sardes.inrialpes.fr/research/AutoMan/.

BOTE-LORENZO, M. L.; DIMITRIADIS, Y. A.; GÓMEZ-SÁNCHEZ, E. Grid characteristics and uses: A grid definition. In: *European Across Grids Conference*. [S.l.: s.n.], 2003. p. 291–298.

BUCHHOLZ, M. *Resource Management in OGSA*. 06 2008. Disponível em: www.gridforum.org/documents/GFD.45.pdf.

CIRNE, W. *Grids Computacionais: da Computação de Alto Desempenho a Serviços sob Demanda*. 06 2008. Disponível em: http://www.inf.ufrgs.br/ jkv/ap-grid.pdf.

CIRNE, W.; SANTOS-NETO, E. Grids computacionais: da computação de alto desempenho a serviços sob demanda. *Simpósio Brasileiro de Redes de Computadores*, n. 23, July 2005. Disponível em: <http://www.sbrc2007.ufpa.br/anais/2005/MC/cap1.pdf>.

CONDOR. 06 2008. Disponível em: http://www.cs.wisc.edu/condor/.

COULOURIS; DOLLIMORE, J.; KINDBERG, T. *Distributed Systems: Concepts and Design (4th Edition) (International Computer Science)*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2005. ISBN 0321263545.

DANTAS, M. *Computação Distribuída de Alto Desempenho*. [S.l.]: Axcel Books, 2005.

DATAGRID. 06 2008. Disponível em: http://www.eu-datagrid.org.

DCTI. 06 2008. Disponível em: http://www.distributed.net/.

DUARTE, A. et al. Collaborative fault diagnosis in grids through automated tests. In: *AINA '06: Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 1 (AINA'06)*. Washington, DC, USA: IEEE Computer Society, 2006. p. 69–74. ISBN 0-7695-2466-4-01.

EPEMA, D. et al. A worldwide flock of condors: Load sharing among workstation clusters. *Future Generation Computer Systems*, v. 12, p. 53-65, 1996.

ESG, E. S. G. 06 2008. Disponível em: http://www.earthsystemgrid.org/about/overviewPage.

FIGHTAIDS. 06 2008. Disponível em: http://fightaidsathome.scripps.edu/.

FOSTER, I. What is the grid? - a three point checklist. *GRIDtoday*, v. 1, n. 6, July 2002. Disponível em: <http://www.gridtoday.com/02/0722/100136.html>.

FOSTER, I. Globus toolkit version 4: Software for service-oriented systems. In: *IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779*. [S.l.: s.n.], 2005. p. 2–13.

FOSTER, I. et al. (Ed.). *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration*. 2002. Disponível em: <http://www.globus.org/alliance/publications/papers/ogsa.pdf>.

GANGLIA. 06 2008. Disponível em: http://ganglia.sourceforge.net/.

GGF. 06 2008. Disponível em: http://www.sun.com.

GLOBUS. 06 2008. Disponível em: http://www.globus.org.

GOLDCHLEGER, A. *InteGrade: Um Sistema de Middleware para Computação em Grade Oportunista*. Dissertação (Mestrado) — Instituto de Matemática e Estatística (IME)- Universidade de São Paulo, Dez 2004.

GOLDCHLEGER, A. et al. InteGrade: Object-Oriented Grid Middleware Leveraging Idle Computing Power of Desktop Machines. *Concurrency and Computation: Practice and Experience*, John Wiley and Sons Ltd., v. 16, n. 5, p. 449–459, March 2004.

GRADS, O. of. 06 2008. Disponível em: http://www.iges.org/grads/.

GRAUPNER, S. et al. Management +=grid. In: *DSOM*. [S.l.: s.n.], 2003. p. 194-196.

GROUP, G. G. F. P. W. 06 2008. Disponível em: http://www- didc.lbl.gov/GridPerf.

HP. 06 2008. Disponível em: http://h41131.www4.hp.com/br/pt/stories.html.

IBM. 06 2008. Disponível em: http://www.ibm.com.

INTEGRADE. 06 2008. Disponível em: http://www.integrade.org.br/portal.

JOIN. 06 2008. Disponível em: http://www.dca.fee.unicamp.br/projects/join/history.html.

KESSELMAN, C.; FOSTER, I. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, 1998. Hardcover. ISBN 1558604758. Disponível em: <http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/1558604758>.

KRAUTER, K.; BUYYA, R.; MAHESWARAN, M. A taxonomy and survey of grid resource management systems for distributed computing. *Softw., Pract. Exper.*, v. 32, n. 2, p. 135–164, 2002.

LCG. 06 2008. Disponível em: http://www.sun.com.

LIMESURVEY. 06 2008. Disponível em: http://www.limesurvey.org/.

MEDEIROS, R. et al. Faults in grids: Why are they so bad and what can be done about it? *grid*, IEEE Computer Society, Los Alamitos, CA, USA, v. 00, p. 18, 2003.

NABRZYSKI, J.; SCHOPF, J. M.; WEGLARZ, J. (Ed.). *Grid resource management: state of the art and future trends*. Norwell, MA, USA: Kluwer Academic Publishers, 2004.

NAGIOS. 06 2008. Disponível em: http://www.nagios.org/.

NETLOGGER. 06 2008. Disponível em: Disponível em http://dsd.lbl.gov/NetLoggerWiki/index.php/.

OGSA. 06 2008. Disponível em: http://www.gridforum.org/documents/GFD.30.pdf.

OURGRID. 06 2008. Disponível em: www.ourgrid.org.

SCALI. 06 2008. Disponível em: http://www.scali.com.

SERVICE, I. N. W. 06 2008. Disponível em: http://nws.cs.ucsb.edu/ewiki.

SETI. 06 2008. Disponível em: http://setiathome.berkeley.edu/index.php.

SOAP. 06 2008. Disponível em: http://www.w3.org/TR/soap/.

THAIN, D.; TANNENBAUM, T.; LIVNY, M. Distributed computing in practice: the condor experience. *Concurrency Practice and Experience*, v. 17, n. 24, p. 323-356, 2005.

TIERNEY, B. *A grid Monitoring Architecture*. 06 2008. Disponível em: http://www-didc.lbl.gov/GGF-PERF/GMA-WG/.

VOLCKAERT, B. et al. *Grid Computing: the next network challenge!* 2004. Disponível em: <citeseer.ist.psu.edu/volckaert04grid.html>.

W3C. 06 2008. Disponível em: http://www.w3.org/.

WS-INSPECTION. 06 2008. Disponível em: http://www.w3.org/2002/ws/arch/2/06/wd-wsa-arch-20020605.html.

WSDL. 06 2008. Disponível em: http://www.w3.org/TR/wsdl.

XML. *XML Schema Part 0: Primer Second Edition*. 06 2008. Disponível em: http://www.w3.org/TR/xmlschema-0/.

XSLT. 06 2008. Disponível em: http://www.w3.org/TR/xslt.

YANG, K. et al. Towards efficient resource on-demand in grid computing. *SIGOPS Oper. Syst. Rev.*, ACM, New York, NY, USA, v. 37, n. 2, p. 37–43, 2003. ISSN 0163-5980.

YERO, E. J. H. *Estudo sobre Processamento Maciçamente Paralelo na Internet*. Dissertação (Mestrado) — Faculdade de Engenharia Elétrica e de Computação - Unicamp, July 2003.

6 Anexo A - Questionário

Um panorama do emprego das tecnologias de grade computacional no Brasil

Perguntas

01: Em que perfil pode ser enquadrado o envolvimento do seu trabalho ou pesquisa com tecnologias de grade computacional?

Por favor escolha *todas* as que se aplicam:

- ☐ Usuário
- ☐ Desenvolvedor
- ☐ Pesquisador

Outro: ____

02: Em que área seu trabalho ou pesquisa vem buscando aplicar tecnologias de grade computacional?

Por favor escolha *todas* as que se aplicam:

- ☐ Aeroespacial
- ☐ Astrofísica
- ☐ Automotiva
- ☐ Computação
- ☐ Economia
- ☐ Física
- ☐ Química

Outro: ____

03: Qual ambiente de grade computacional é empregado por seu grupo ou instituição?

Por favor escolha *todas* as que se aplicam:

- ☐ Globus Toolkit
- ☐ Ourgrid
- ☐ Condor
- ☐ Data Grid
- ☐ Mygrid
- ☐ Glue

Outro: ____

---

* **04: Qual foi o nível de dificuldade enfrentado na implantação do seu ambiente de grade computacional?**

Escolha *apenas uma* das opções seguintes:

- ☐ Alto (necessário conhecimento de muitos detalhes dos níveis hierárquicos e subsistemas envolvidos)
- ☐ Médio (precisa ser parcialmente configurado, requerendo

conhecimento de algumas abstrações empregadas)

☐ Baixo (pouco ou nenhum trabalho requerido, oferecendo facilidades e recursos de autoconfiguração)

*** 05: Uma vez configurado, como você classifica o esforço na manutenção e gerência do seu ambiente de grade?**

Escolha *apenas uma* das opções seguintes:

☐ Alto (o sistema oferece pouca ou nenhuma facilidade nem interfaces de gerenciamento)

☐ Médio (oferece algum nível de facilidade, com emprego de algumas ferramentas de gerenciamento)

☐ Baixo (o sistema oferece boas ferramentas e interface de gerenciamento)

06: Quais os tipos de falhas mais freqüentemente encontradas em seu ambiente de grade?

Por favor escolha *todas* as que se aplicam:

☐ Falhas das Aplicações

☐ Falhas de Configuração (incompatibilidade de software, erros de configuração, etc.)

☐ Falhas de Middleware

☐ Falhas de Hardware

Outro: ☐

07: Qual estratégia de detecção e/ou correção e/ou tolerância a falhas utilizada em seu ambiente de grade?

Por favor escolha *todas* as que se aplicam:

☐ Escalonamento tolerante a faltas (fault-tolerant scheduling)

☐ Restauração (checkpointing-recovery)

☐ Ferramentas de visualização

☐ Testes automáticos

☐ Sistemas de Monitoramento

☐ Dependente de Aplicação

Outro: ☐

08: Quais são os maiores problemas enfrentados para se recuperar de um cenário de falha em seu ambiente de grade?

Por favor escolha *todas* as que se aplicam:

☐ Implementar um comportamento específico de recuperação dependente de aplicação

☐ Ganhar autorização para corrigir o componente defeituoso

☐ Realizar o diagnóstico da falta detectada

Outro:

* **09: Qual o grau de envolvimento do usuário durante o processo de recuperação de falha?**

Escolha *apenas uma* das opções seguintes:

- ☐ Alto (o usuário precisa estar envolvido no processo especificando exatamente o que deve ser feito)
- ☐ Médio (o usuário precisa estar envolvido no processo, porém apenas marginalmente)
- ☐ Baixo (o usuário não precisa ser envolvido porque o sistema oferece recuperação automática de falhas)

12: Você utiliza tecnologias e/ou aplicações para auxiliar no monitoramento do seu ambiente de grade computacional?

Por favor escolha *todas* as que se aplicam:

- ☐ Nagios
- ☐ Ganglia
- ☐ NetLogger
- ☐ Monalisa
- ☐ Aplicações baseadas em agentes
- ☐ Aplicações baseadas no JINI
- ☐ Nenhuma

Outro:

* **10: Seu sistema de grade computacional utiliza algum GRMS (Grid Resource Management System)?**

Escolha *apenas uma* das opções seguintes:

- ☐ Sim
- ☐ Não

[Responder apenas esta questão Se você respondeu 'Sim' para a questão '10 ']

* **11: Com relação ao gerenciamento de recursos, que facilidade seu GRMS oferece:**

Por favor escolha *todas* as que se aplicam:

- ☐ Gerenciamento de alocação
- ☐ Gerenciamento de disponibilidade
- ☐ Gerenciamento de autenticação
- ☐ Gerenciamento de autorização
- ☐ Gerenciamento de auditoria
- ☐ Descoberta automática

Outro: