**Luiz Eduardo Simeone**

# Ferramenta de apoio à administração de redes de computadores e suporte ao usuário

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências para a obtenção do título de Bacharel em Ciência da Computação

Orientador
Prof. Samuel Pereira Dias

Co-Orientador
Prof. Heitor Augustus Xavier da Costa

Lavras
Minas Gerais - Brasil
2004

**Luiz Eduardo Simeone**

**Ferramenta de apoio à administração de redes de computadores e suporte ao usuário**

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências para a obtenção do título de Bacharel em Ciência da Computação

Aprovada em *24 de Junho de 2004*

Prof. Reginaldo Ferreira de Souza

Prof. Samuel Pereira Dias
(Orientador)

Prof. Heitor Augustus Xavier da Costa
(Co-Orientador)

Lavras
Minas Gerais - Brasil

# Sumário

# Lista de Figuras

*"As coisas mais importantes da vida são invisíveis e imortais."*
*Frank S. Land*
*Fundador da Ordem DeMolay*

# Agradecimentos

Agradeço aos professores e servidores do Departamento de Ciência da Computação, em especial ao Prof. Samuel Pereira Dias, pelo companheirismo, amizade, orientação e paciência. Agradeço à coordenação do curso de pós-graduação *lato-sensu* "Administração em Redes Linux", principalmente o Prof. Joaquim Q. Uchôa, que me cedeu espaço e equipamento para trabalhar no que eu realmente gosto: *Software* Livre.

Agradeço aos colegas de curso, que sempre estiveram prontos a ajudar, ensinar e dividir os momentos que a vida universitária , em especial Carlos Eduardo (Sidnelson), Rodrigo Toso, Flávio, André Fialho, Rafael Frinhani, André Barros (Todé), Elaine, Matheus, Geovane e Érika. Agradeço aos colegas do Expresso Nepomuceno Rafael, Gustavo, Márcio e Kênia pelo espaço que me foi cedido para que eu pudesse concluir este trabalho. Agradeço também aos colegas de república que tive durante o curso: João Paulo, Danilo, Diego, José Henrique, Márcio, Diego Souza e Thiago pelo companheirismo, ensinamentos e vivência.

Agradeço a minha noiva Luciana, meu irmão Fernando, meu pai Eduardo, minha mãe Orani e toda a minha família pela ajuda, incentivo, apoio e carinho durante os anos de universidade. Agradeço à Ordem DeMolay, que foi a guia da minha conciência, e ao Pai Celestial pela oportunidade de aprender a cada dia com todas as pessoas, desde a mais simples até o mais importante.
Obrigado.

## Resumo

Este trabalho tem como objetivo apresentar a situação atual do suporte ao usuário e redes de computadores nas organizações. Após essa apresentação o texto propõe uma ferramenta com *interface web* para a documentação de equipamentos e chamados técnicos, utilizando programação orientada a objetos e ferramentas de desenvolvimento livres.



**Palavras-chave:** suporte ao usuário, redes de computadores, *software* livre, *helpdesk*



## Abstract

The research objective is to present the situation of user and computer networks support. After analyze this presentation, this work will propose a software with web interface to control support's calls and hardware administration, using object-oriented programming and free development tools.



**Keywords:** user support, computer networks, free software, helpdesk

# Capítulo 1

# Introdução

O suporte ao usuário de computador é um problema que existe em praticamente todas as organizações onde existem computadores. Muitas vezes essa questão é deixada de lado pela direção das empresas que, na maioria dos casos, somente visualizam a área de negócio da organização. Com isso o setor de tecnologia da informação fica em segundo plano, como uma área que somente consome recursos e nunca atende de forma satisfatória as necessidades da mesma.

Com as novas tecnologias da informação, a *internet* e a explosão na capacidade de processamento dos computadores, os usuários mudaram de perfil, como explicado na Seção 2.1. As suas necessidades em relação aos equipamentos, *software* e, principalmente, suporte mudaram. Muitas vezes as organizações não se dão conta disso mantendo o funcionamento do suporte igual ao dos tempos do terminais "burros"[1]. Por conta disso, os usuários ficam insatisfeitos com o suporte, o setor fica sem crédito com a direção da organização, e isso acaba gerando uma lacuna dentro da mesma.

O suporte ao usuário é peça-chave para o sucesso da política de tecnologia da informação da organização. Muitas vezes as próprias organizações, o meio acadêmico e os demais setores não dão a devida atenção à questão.

## 1.1 Motivação

Este trabalho foi motivado pela necessidade de documentar vários aspectos de um ambiente computacional. Até então isso era feito como no período anterior à invenção da escrita: o conhecimento do ambiente, da estrutura e dos equipamentos era transmitida do funcionário mais antigo para o funcionário mais novo. Outro problema que motivou o trabalho foi a não de utilização de um sistema de registro,

[1]Terminais Burros: estações de trabalho sem capacidade de processamento. Todo o processamento é feito em um computador central

organização e coordenação dos chamados técnicos nos laboratórios do DCC[2] da UFLA[3]. Essa ausência foi detectada no trabalho do autor como monitor, auxiliando na administração da rede do DCC da UFLA.

Muitas vezes as solicitações eram feitas no corredor do Departamento de Ciência da Computação. Com isso, as mesmas eram esquecidas, não eram atendidas em tempo satisfatório e até nem chaegavam ao conhecimento dos administradores da rede. Com um sistema de documentação desses chamados técnicos, as falhas ficariam mais evidentes e corrigi-las também seria mais fácil.

## 1.2 Objetivos

Os objetivos deste trabalho são:

- Fazer um levantamento dos trabalhos que já existem na área de suporte ao usuário.
- Levantar as necessidades, juntamente aos administradores de rede do Departamento de Ciência da Computação, de gerenciamento de chamados técnicos e equipamentos.
- Propor um conjunto de ferramentas para auxiliar a tarefa de administar redes e gerenciar o atendimento aos usuários e os resultados obtidos.

## 1.3 Estrutura do texto

Esta seção tem por objetivo apresentar os demais capítulos do texto. O Capítulo 2 trata de vários aspectos do suporte ao usuários. O Capítulo 3 trata da administração de redes e as funções do administrador de redes. O Capítulo 4 traz uma introdução ao *software* livre, da mesma forma que o Capítulo 5 apresenta a programação orientada a objetos. O Capítulo 6 trata da metodologia utilizada para o desenvolvimento e o Capítulo 7 da modelagem da aplicação proposta por este texto. O Capítulo 8 apresenta a ferramenta desenvolvida enquanto o Capítulo 9 traz as conclusões.

[2]Departamento de Ciência da Computação, `http://www.dcc.ufla.br/`
[3]Universidade Federal de Lavras, `http://www.ufla.br/`

# Capítulo 2

# Suporte ao usuário

O suporte ao usuário, segundo [Cordani e Nagel 1995] não é um produto, mas sim um processo. Estão envolvidos nesse processo desde a documentação fornecida ao usuário, o *helpdesk* até o as equipes de campo. O propósito deste capítulo é ilustrar alguns aspectos desse processo.

## 2.1 Mudanças nos computadores e seus usuários

No princípio, os recursos computacionais eram escassos e compartilhados entre várias pessoas. Os usuários eram pessoas altamente treinadas, e que raramente precisavam de ajuda para operar a máquina. O acesso físico aos computadores era restrito e somente um pequeno grupo realmente operava o computador. E esse grupo acabava ditando as regras para as pessoas que queriam utilizar o computador para executar seus *jobs*[1]. Segundo [Cordani e Nagel 1995], o suporte estava concentrado apenas nos problemas no equipamento, como falhas de hardware. Com a evolução e disseminação dos computadores, principalmente dos microcomputadores, o perfil do usuário mudou muito. Hoje o usuário não se interessa mais pelos detalhes técnicos ou aspectos operacionais, mas por saber como executar uma determinada tarefa, sem riqueza de detalhes. Muitas vezes o suporte não está preparado para atender esse anseio do usuário. O suporte apenas responde técnicos sobre o funcionamento de algum aplicativo ou equipamento. No entanto, ele não consegue responder questões simples, que facilitam a utilização do computador como uma ferramenta.

## 2.2 Suporte Distribuído

De acordo com [Pear 1988], um modelo de suporte distribuído tem duas características básicas; uma delas é o foco de cada uma das equipes centrado numa área

[1] ***Jobs:*** tarefas, conjuntos de programas a serem executados

de cobertura bem restrita. Com isso a equipe consegue fornecer um suporte mais adequado aos usuários. Uma equipe de suporte mais próxima conhece as tarefas das pessoas e pode colaborar de uma forma mais efetiva para a realização dessas tarefas. O suporte local, portanto, deve ser o primeiro contato para a resolução dos problemas dos usuários. A outra característica é estabelecer um único ponto de contato para informações, recursos e serviços em cada uma das seções. Esse único ponto de contato pode fornecer informações mais precisas sobre a necessidade do departamento ou seção. Além disso, filtrar as requisições que chegam, resolvendo com mais facilidade os problemas simples. Porém pode ser um grande engano acabar com o núcleo central do suporte. Ele pode facilitar várias coisas, entre elas a aquisição de equipamentos, *software* e suprimentos a um custo mais baixo, além de elaborar e administrar padrões para toda a instituição. Outra função importante do suporte central é funcionar como coordenador das equipes de suporte local, promovendo a troca de informações e experiências entre as equipes.

Além do proposto por [Pear 1988], [Saini e Lavoie 1991] propõem a criação de grupos de usuários para que eles possam discutir assuntos como novas tecnologias, novas formas de utilizar o computador como ferramenta, de acordo com as necessidades do grupo. Esses grupos também colaboram com a equipe de suporte de forma a facilitar ainda mais a interação entre os usuários e incrementar os conhecimetos da equipe local sobre as necessidades dos usuários. Outro ponto proposto é a criação de uma espécie de comitê para discutir as políticas a serem utilizadas na instituição na área de tecnologia da informação, como compra de *hardware* e *software* e utilização de recursos instalados, políticas de segurança, entre outras.

## 2.3 Equipes de suporte em universidades: utilização de estudantes como mão-de-obra

Além de suporte distribuído, as Universidades, principalmente as que têm cursos na área de informática, podem utilizar estudantes como mão-de-obra nas equipes de suporte ao usuário, trabalhando nas folgas das aulas, de acordo com [Diagle 2003, McRitchie 2003]. Isso é interessante, pois a instituição consegue material humano capacitado para as funções técnicas a um custo muito baixo. Freqüentemente os estudantes são até voluntários, ganham experiência trabalhando em um ambiente diferente do encontrado nas salas de aula, enriquecem o seu currículo com mais horas de estágio e podem até receber *pró-labore*[2].

Conforme [Diagle 2003], a seleção dos estudantes deve ser feita levando em conta o conhecimento prévio e a área em que cada um deles mais se identifica. Essas características devem ser levadas em consideração ao decidir o local onde o aluno será alocado. O treinamento pode ser feito de várias formas, entre elas:

[2] *Pró-labore:* remuneração pelo reabalho realizado.

mini-cursos, desafios, jogos, questionários e treinamento individualizado. Após a efetivação desses estudantes, devem ser fornecidas todas as ferramentas que eles necessitarão durante o seu trabalho, incluindo documentação, uma fonte de consulta sobre problemas e soluções, ferramentas para manutenção (alicates, chaves diversas, etc.) e ferramentas de comunicação. Podem ser utilizados meios de comunicação síncrona como o ICQ[3] e o Microsoft MSN[4], além murais *on-line*, *e-mails*, rádios *walk-talkies* e o telefone.

Já para [McRitchie 2003] o processo de seleção deve ser mais rigoroso, utilizando um sistema de avaliação baseado em pontuação, com provas escritas e entrevistas. O treinamento é composto por três fases básicas. Na primeira fase, o estudante é colocado em um projeto e deve utilizar seu conhecimento para solucionar os problemas propostos. Na segunda, o novato é colocado junto de um outro aluno experiente, chamado mentor, com o intuito de ambienta-lo com a rotina do trabalho. Na terceira fase, são feitas simulações, onde os novatos são colocado em situações que irão encontrar no dia-a-dia do trabalho.

## 2.4 Situação do suporte ao usuário

Segundo [Govindarajulu 2002], no início dos anos 80 e também da era PC[5], o *helpdesk*, que era a forma de suporte encontrada na época, era visto como um enviado divino. Porém, ainda segundo [Govindarajulu 2002], atualmente, o índice de insatisfação dos usuários com o suporte oferecido é muito alto. Essa insatisfação fez com que os usuários procurassem ajuda em outras fontes, como colegas de trabalho e amigos. Outro ponto interessante é que os usuários finais tendem a usar o *helpdesk* o mínimo possível, provavelmente pela existência de um equipe de suporte local, ou de pessoas que possam sanar as dúvidas no próprio ambiente de trabalho. Um dado interessante é que existem mais equipes de pequeno e médio porte oferecendo suporte operacional, em detrimento a equipes de grande porte. Esses dados mostram a necessidade de mudanças no suporte ao usuário.

De acordo com [Thompson 1994], o suporte deve agir de forma a não somente resolver os problemas dos usuários, mas também torná-los mais auto-suficientes. Foram criadas várias formas alternativas, incluindo treinamento baseado em áudio, através de fitas cassete e CDs, vários tutoriais que explicam como fazer tarefas passo-a-passo e informativos diversos que são enviados por *e-mail* e fixados em vários locais. As conseqüências disso foram o aumento no numero de chamadas em mais de 30% e uma maior satisfação do usuário em relação ao suporte.

[3]Sigla sem significado, porém quando pronunciada na língua inglesa tem som parecido com o da expressão *I Seek You*,`http://www.icq.com`

[4]Microsoft Network,`http://www.msn.com`

[5]**PC:** *personal computer* ou computador pessoal. Arquitetura criada pela IBM no início dos anos 80. Tornou-se padrão de mercado vigente até hoje

## 2.5 Automatização de *Helpdesks*

O objetivo desta seção é introduzir alguns conceitos sobre os *helpdesks* e algumas funcionalidades, recursos e tecnicas de implementação dos mesmos

Segundo [Saul, Black e Larsson 2000] um helpdesk deve possuir um sistema de rastreamento de chamados baseado em *tickets*. Esse sistema facilita a localização de equipamentos pela equipe de suporte e é capaz de fornecer ao usuário uma informação de quais providências já foram tomadas em relação ao chamado, além do estado atual. Outro ponto importante é um mecanismo de autenticação que possa identificar automaticamente os usuários e verificar qual o tipo de acesso que ele terá à ferramenta. Também propõe um sistema de notificação, que periodicamente fornece ao usuário informações sobre os chamados e equipamentos em poder do suporte. A interface do sistema deve torná-lo facil de usar e ser intuitiva para usuários menos experientes.

Já [Diem 1998] propôs um sistema que além das características acima, automaticamente aponta qual a tipo de problema de acordo com o chamado. Outra funcionalidade proposta foi um sistema de escalonamento de tarefas, que escolhe qual integrante da equipe de suporte vai atender o chamado conforme horário, tipo de problema e disponibilidade de cada um. Também foi proposto um histórico de problemas para facilitar consultas e levantamentos estatísticos.

A proposta de [Link 2002] é incluir no *helpdesk* informações sobre a rede, os servidores e os serviços através de monitoramento, para que o usuário possa se informar das novidades e situação atual da rede através do próprio *helpdesk*. Também foi incluída na proposta de [Link 2002] que envia boletins informativos diversos aos usuários, e meios de comunicação entre os membros da equipe de suporte. A idéia é transformar o *helpdesk* em um centro de informações.

Já [Coventry e Kane 1993] vão mais além e propõem um sistema de *helpdesk* semi-automático. Para os autores é impossível que alguém seja especialista em tudo, então eles propõem um sistema especialista, utilizando inteligência artificial distribuída. O sistema é baseado em vários quadros-negros assíncronos, que são agentes inteligentes. Esses agentes se comunicam e cada um tem uma função específica, como: interface com o usuário, controle dos outros quadros-negros e registro de chamados ou diário. De acordo com as análises dos *logs*[6] da ferramenta, após um ano de operação, as questões mais freqüentes dos usuários estão baseadas em seis palavras-chave:

1. **Como**
    - Como visualizar um arquivo na tela.
    - Como imprimir utilizando a impressora do laboratório.
2. **Posso**

[6] ***Log:*** registro de eventos

- Posso ter UNIX instalado em meu PC?

3. **Preciso**
   - Preciso transferir arquivos para o servidor.
   - Preciso de um teclado novo.
4. **Não consigo**
   - Não consigo enviar *e-mail* para o domínio comp.ufla.br.
   - Não consigo instalar o OpenOffice.org[7] em minha conta.
5. **Problemas**
   - Problemas ao formatar o disquete.
   - Problemas no Open Webmail[8].
6. **Quais**
   - Quais os visualizadores de imagem disponíveis no laboratório 01?

Essas informações são obtidas do agente responsável pelo diário, que também registra informações como taxa de sucesso no atendimento aos chamados e informações sobre os usuários que acessaram a ferramenta. As informações acima poderiam ser utilizadas para organizar treinamentos, apostilas e material de consulta como *FAQ*[9] e orientar o trabalho da equipe de suporte.

[7] `http://www.openoffice.org`
[8] `www.openwebmail.org`
[9] **FAQ:***Frequently Asked Questions* ou perguntas mais freqüentes

# Capítulo 3

# Administração de Redes

Este capítulo traz os conceitos básicos sobre a administração de redes e as funções do administrador de rede, e também sua importância dentro da organização.

De acordo com [Nemeth et al. 1995], as funções básicas de um administrador são:

- **Gerenciar contas de usuários.** É função do administrador criar as contas dos novos usuários, remover as contas dos usuários que não existem mais e gerenciar suas quotas;

- **Fazer *backups* do sistema.** Fazer os *backups* é uma das funções mais importantes do administrador. Os *backups* devem ser automatizados e executados nos momentos mais adequados;

- **Instalar novo *software*.** Quando um novo *software* ou uma atualização está disponível, é dever do administrador instalá-los em local apropriado, e gerenciar os existentes evitando que eles possam ser danificados por usuários ou por outro *software*;

- **Monitorar o sistema.** O administrador deve sempre verificar os registros e a situação de vários outros pontos como, espaço em disco e situação da rede, empenhado na máxima disponibilidade do sistema;

- **Manutenção.** É dever do administrador fazer os reparos no sistema quando ele pára de funcionar corretamente por um motivo qualquer. Muitas vezes, é mais difícil identificar o problema que corrigi-lo;

- **Manter a documentação do sistema.** É fundamental que seja mantida documentação sobre todos os aspectos do sistema, entre eles o *hardware*, o *software* instalado, a rede, as configurações, a política de *backup*, para que no futuro a manutenção torne-se mais fácil e rápida, além de organizada;

- **Auditar a segurança.** O administrador deve sempre checar o estado da segurança do sistema, evitando elementos intrusos que podem desde danificar o sistema até obter informações confidenciais. Isso deve ser feito através de programas de auditoria e da análise dos registros do sistema;

- **Ajudar os usuários.** Como administrador o sistema tem um conhecimento maior sobre a sua operação, é importante que ele forneça ajuda aos usuários, e essa atividade toma um tempo considerável. Entretanto, o treinamento oferecido aos usuários e às equipes de suporte pelos administradores possuem impactos positivos na confiança depositada pelos usuários no suporte e no próprio setor de tecnologia da informação

Quanto mais o administrador aprende sobre o sistema, mais a comunidade de usuários depende dele. Com o crescimento da rede da organização, pode ser que o administrador de redes, em diversas situações, seja a única pessoa capaz de desempenhar várias funções vitais para essa organização. Isso pode levar à sobrecarga de funções, ou seja, muitas vezes além de administrador de redes, o indivíduo tenha que atuar como engenheiro de *software*, gerente de projeto, etc. Por isso um sistema de documentação de rede é fundamental para as organizações. Se o único colaborador que conhece o funcionamento da rede deixa a organização sem deixar a documentação, o seu substituto pode demorar muito tempo para entender a rede, ou até nem conseguir.

Muitos administradores tratam as requisições com displicência, gerando uma imagem ruim perante as outras pessoas que dividem o ambiente de trabalho. Ao contrário, ele deve organizar bem o seu tempo e manter sempre uma postura próativa para que ele possa contribuir de maneira adequada com a organização em que trabalha e tornar o ambiente de trabalho mais agradável. Isso inclui estabelecer um cronograma de trabalho para realizar todas as suas tarefas, programar as paradas para manutenção com antecedência e em horário que cause o menor impacto possível na rotina da organização e buscar sempre melhoria na qualidade de serviço.

# Capítulo 4

# *Software* Livre

O Software Livre, tradução do inglês *free software*, onde *free* se traduz como livre e não grátis, é o software que está associado à liberdade e não ao preço, segundo [FSF 2003]. Logo o usuário tem a liberdade de executar, distribuir, copiar, estudar, modificar e aperfeiçoar o *software*. Mais precisamente, são quatro tipos de liberdade para os usuários do *software*:

- A liberdade de executar o programa, para qualquer fim (liberdade 1);
- A liberdade de estudar como o programa funciona e adaptá-lo para as suas necessidade (liberdade 2). Acesso ao código-fonte é um pré-requisito para esta liberdade;
- A liberdade de redistribuir cópias para que você possa ajudar ao seu próximo (liberdade 3);
- A liberdade de aperfeiçoar o programa e liberar os seus aperfeiçoamentos, para que toda a comunidade se beneficie (liberdade 4). Acesso ao código-fonte é um pré-requisito para esta liberdade.

Mas para que essas liberdades sejam efetivas é importante que elas sejam concedidas de forma irrevogável pela entidade que desenvolveu o *software*. Para tanto é feito um acordo entre o desenvolvedor e o consumidor chamado "licença de uso". Essa licença rege os direitos e obrigações de cada uma das partes envolvidas, como um contrato, de forma que sejam asseguradas as liberdades fundamentais do *software* livre, e também os direitos intelectuais do desenvolvedor. A licença mais difundida e famosa é a GNU[1] GPL (*General Public License*), mas existem outras licenças que caracterizam o *software* como livre, sendo que elas podem ou não ser compatíveis com a GNU GPL. Ser compatível com a GPL significa que é possível combinar um módulo que foi distribuído sob essa licença compatível

[1] "*GNU is Not Unix*" ou "GNU Não é Unix", `http://www.gnu.org/`

com um módulo coberto pela GPL para formar um programa maior, sem violar os termos da GPL. Licenças como a licença BSD[2] modificada e a licença *Intel Open Source License* são exemplos de licenças que são totalmente compatíveis com a GPL. Já a *IBM Open License*, a Licença Pública do Mozilla e a Licença Pública da *Sun* aplicadas a um *software* o caracterizam como *software* livre, mas elas não são compatíveis com a GNU GPL.

Existem vários projetos e organizações que apóiam o desenvolvimento do *software* livre, mas os dois principais são: o Projeto GNU e a Iniciativa *OpenSource*. Os dois projetos serão apresentados nas seções seguintes.

## 4.1 Projeto GNU

O Projeto GNU foi idealizado em 1983, por Richard Stallman, como uma forma de trazer de volta o espírito cooperativo que prevalecia na comunidade de informática nos seus primórdios. Em 1971, quando Richard Stallman iniciou a sua carreira no MIT[3], ele trabalhava em um grupo que usava exclusivamente *software* livre. As empresas de informática, na época, distribuíam *software* livre. Os programadores eram livres para cooperar entre si e frequentemente faziam isso. Nos anos 80, quase todo o *software* era proprietário, o que significa que ele tinha donos que proibiam e impediam a cooperação entre os usuários. Como o sistema operacional é condição *sine quœnon*[4] para a utilização de um computador, esse foi o primeiro objetivo do projeto: construir o sistema operacional GNU de tal forma que ele fosse compatível com o Unix, visto que já era uma plataforma testada e estável, e isso também facilitaria a migração dos usuários do Unix para o GNU. Mas um sistema operacional não é somente composto por um núcleo, mas também por interpretadores de comando, editores de texto, dentre outras aplicações. O Projeto GNU começou a implementação por essas aplicações, entre elas o GNU EMACS e o GCC (*GNU Compiler Collection*). No início dos anos 90 o objetivo do projeto GNU foi atingido quando o finlândes Linus Torvalds disponibilizou um *kernel*[5] sob licença GPL chamado Linux. A união dos vários aplicativos desenvolvidos pelo projeto GNU com o Linux originou o famoso GNU/Linux, que erroneamente é chamado somente de Linux. Além disso, o Projeto GNU desenvolve várias outras aplicações como jogos por exemplo, e até um gerenciador de janelas[6], o GNOME[7]. Portanto o objetivo do Projeto GNU é fornecer ao usuário softwares livres que atendam necessidades do usuário de forma completa, tornando

[2] *Berkeley Software Distribution*

[3] *Massachussets Institute of Technology*, `http://www.mit.edu/`

[4] Sine quænon: sem a qual, indispensável

[5] Núcleo do sistema operacional, responsável por gerenciamento de memória e acesso à disco, entre outras funções

[6] Programa que gerencia o ambiente gráfi co do sistema

[7] *GNU Object Model Envoirement:*`http://www.gnome.org/`

assim o *software* proprietário desnecessário. Para dar suporte logístico e financeiro ao projeto GNU, Richard Stallman criou a *Free Software Foundation*[8][9] (Fundação para o Software Livre), que arrecada fundos para sustentar o projeto através de doações, venda de manuais e livros impressos, promove congressos e eventos, entre outras atividades.

## 4.2 Iniciativa *Open Source*

Segundo [OSI 2004] a Iniciativa *Open Source* é um programa de *marketing* para o software livre. A expressão "*software* livre" não soa bem aos executivos, pois muitas vezes eles associam o *software* livre a piratas digitais e outros tipos de grupos ou indíviduos que praticam atos ilícitos utilizando o computador. Isso é puramente preconceito em relação às pessoas que desenvolvem o *software* livre, mas é a realidade. A Iniciativa *Open Source* usa as idéias de sucesso do *software* livre, mas muda a atitude perante as empresas. De acordo com [OSI 2004], a Iniciativa *Open Source* utiliza as mesmas pessoas, o mesmo *software*, as mesmas licenças e troca o rótulo do produto de *software* livre para *software open source*, ou código aberto. Com essa mudança de rótulo, os responsáveis por tecnologia da informação das empresas começaram a adquirir soluções *open source*. Segundo a Iniciativa *Open Source* [OSI 2004], para um *software* ser considerado de código aberto, ele tem que obedecer a dez critérios:

- Livre redistribuição.
- Codigo fonte disponível.
- Os trabalhos derivados devem ser distribuídos sob a mesma licença.
- Integridade do código fonte do autor. Isso significa que o código-fonte modificado só pode ser distribuído se a licença permitir ou o código modificado tem que estar identificado como modificado.
- Não pode existir discriminação entre pessoas e/ou entre grupos.
- O uso do *software* não pode ser restrito pela licença.
- A licença de uso do *software* deve ser a mesma para todos os usuários.
- A licença de uso não pode ser específica para um produto.
- A licença não pode restringir outro *software*, como por exemplo, exigir que o sistema operacional seja de código aberto.

[8] `http://www.fsf.org/`

[9] A FSF no brasil é refresentada pelo CIPSGA (Comitê de Incentivo à produção de *software* GNU e alternativo), `http://www.cipsga.org.br`

- A licença deve ser tecnologicamente neutra, ou seja, não deve exigir a utilização de uma tecnologia específica.

O que diferencia a Iniciativa *Open Source* do Projeto GNU é a visão do mesmo produto. O Projeto GNU tem uma postura mais voltada para a comunidade usuária de *software*. Já a Iniciativa *Open Source* tem uma postura comercial, visualizando no *software* de código aberto uma oportunidade de bons negócios, mas sem deixar de totalmente de lado as questões ideológicas que motivaram a criação do *software* livre nos anos 80.

# Capítulo 5

# Programação orientada a objetos

Este capítulo tem como objetivo ilustrar os conceitos básicos sobre porgramação orientada a objetos, utilizada para o desenvolvimento da aplicação proposta neste trabalho.

## 5.1 Classes, objetos e instâncias

Segundo [Ambler 1998] um objeto é um indivíduo, lugar, coisa, tela, relatório ou conceito que seja aplicável ao sistema. De acordo com [Cardoso 2003] o mundo real é orientado a objeto, ou seja, existem vários objetos diferentes, com comportamentos diferentes, que se relacionam de diversas formas. Esses objetos têm características, que são chamados de atributos, e executam os mais diversos tipos de ações ou comportamentos, que são chamados de métodos, serviços, operações ou atividades. Esses métodos são responsáveis por alterar o estado do objeto.

No mundo real, ainda existem vários grupos de objetos que tem características semelhantes, e podem ser descritos de uma forma abstrata através de seus atributos e serviços. Essa abstração dos objetos semelhantes é chamada de classe. Na classe são definidos quais características e comportamentos que os objetos que dela fazem parte. O objeto é a instância de uma classe.

## 5.2 Abstração da realidade

O mundo real é de grande complexidade e, dependendo do problema que o programa se propõe a resolver, muitas das características do objeto real são irrelevantes para o sistema. Então é feita uma abstração do objeto real, de forma que sejam descritas na classe somente os atributos e métodos que realmente importam para o problema em questão. Se o problema fosse outro, muito provavelmente a abstração feita a partir do objeto real seria diferente, ou seja, o objeto seria descrito na classe de forma diferente ([Ambler 1998]).

## 5.3 Encapsulamento

Diferentemente da programação estruturada, onde os dados são acessados diretamente pelos mais diversos procedimentos do programa, sem proteção ou organização. Na programação orientada à objeto, os atributos de um objeto não podem ser acessados ou manipulados diretamente por um agente externo.

Para alcançar os atributos de um objeto, um agente externo, que pode ser um outro objeto, deve executar um dos métodos do objeto em questão para tal. Isso torna a programação mais segura e simples.

Quem está utilizando aquele objeto não precisa conhecer como ele funciona internamente, mas apenas a sua *interface* com o mundo exterior, e evita que os atributos do objetos sejam manipulados de forma indevida. Isso é chamado de encapsulamento.

## 5.4 Herança

Varios objetos tem características semelhantes, porém não são da mesma classe. A relação de herança permite que objetos mais específicos, como por exemplo os da classe ”carro”, herdem os atributos e métodos de uma classe mais genérica, neste caso a classe “automóvel” e apenas implemente os atributos e métodos que são específicos dessa classe. Da mesma forma, poderia ser criada uma classe “caminhão” que herdaria os atributos e métodos da classe “automóvel” apenas implementando as peculiaridades de um caminhão. Neste caso, a classe “caminhão” é uma classe filha ou subclasse de “automóvel”, da mesma forma que a classe “automóvel” é a classe mãe ou superclasse de “caminhão”.

Pode ser que uma classe mãe não implemente todos os métodos que são declarados, deixando essa implementação por parte das classes filhas. Neste caso a classe mãe é chamada de classe abstrata, pois não podem ser instanciados objetos a partir dela. Em alguns casos, a classe filha pode redefinir os métodos descritos na classe mãe, de forma a permitir um comportamento diferente da classe filha. Esse procedimento é chamado de redefinição ou sobrecarga, de acordo com [Ambler 1998].

## 5.5 Polimorfi smo

De acordo com [Cardoso 2003], polimorfismo é a propriedade que indica que um método pode, mesmo tendo o mesmo nome, executar ações diferentes. Por exemplo, as classes “digitador” e “analista” têm um método chamado “trabalhar” herdado da classe mãe “funcionário”. Na classe “analista”, o método trabalhar pode executar uma ação diferente do método trabalhar na classe “digitador”. Isso é chamado de polimorfismo dinâmico. Já quando existem em uma mesma classe

dois métodos de nome igual, mas que executam ações diferentes, configura-se o polimorfismo estático.

# Capítulo 6

# Metodologia

Inicialmente foi feito um estudo para a definição das ferramentas, ambientes e linguagens de programação a serem utilizadas. A linguagem PHP (*PHP Hypertext Preprocessor*)[1] foi escolhida por ser uma linguagem simples, de fácil utilização e muito parecida com C++. Outros aspectos favoráveis à PHP foram o volume de documentação disponível para consulta, o ambiente já disponível nos servidores do Departamento de Ciência da Computação da UFLA e sua fácil integração com o servidor web Apache[2]. Foi feito um estudo da linguagem PHP, como [Zandstra 2000, Bakken et al. 2004], suas funcionalidades, bibliotecas, utilização e questões de segurança. Esse estudo foi de grande importância para evitar que fossem reimplementadas funções já prontas na linguagem. Também foram estudadas as especificações da linguagem HTML ([Ragget 2002, W3C 1999]) para a construção do código da *interface* da ferramenta. Outro ponto foi a escolha do SGBD (sistema gerenciador de banco de dados) a ser utilizado para o desenvolvimento inicial do sistema. O MySQL [3] foi escolhido por ser um SGBD de fácil instalação e configuração, com vasta documentação e, principalmente, de fácil integração com o PHP. Também foi feito um estudo das funcionalidades do MySQL de forma a facilitar o desenvolvimento. Além disso, foram escolhidos os ambientes de desenvolvimento a serem utilizados durante o trabalho, sendo eles o Anjuta[4] como editor de código PHP, o Bluefish[5] como editor de código HTML, o *Umbrelo UML Modeller*[6] como ferramenta para o desenvolvimento dos diagramas utilizados durante o processo de desenvolvimento, e o DBDesigner[7] para o desenvolvimento do modelo de dados a ser utilizado.

[1] `http://www.php.net/`
[2] `http://www.apache.org/`
[3] `http://www.mysql.com/`
[4] `http://www.anjuta.org/`
[5] `http://bluefish.sourceforge.nl/`
[6] `http://uml.sourceforge.org/`
[7] `http://www.fabforce.net/`

Na modelagem foi feita uma simplificação das técnicas de Engenharia de *Software* [Cardoso 2003] [Ghezzi, Jahzayeri e Mandrioli 1991], utilizando, no trabalho como um todo um processo de desenvolvimento em espiral, com quatro fases (análise, projeto, implementação e teste). Com o auxílio da ferramenta CASE[8] *Umbrello UML Modeller*, foi gerada estrutura do código fonte, com comentários iniciais nos arquivos, nos métodos e atributos de cada classe. O trabalho de implementação foi feito partindo das classes mais básicas, chegando até a classe que faz a *interface* com a base de dados. Foram feitos testes ao final da implementação de cada uma das classes. Em seguida foram feitos testes com as classes funcionando em conjunto. O resultado dos testes foi utilizado para realimentar o processo de desenvolvimento, ajudando a corrigir os erros e refinar a solução.

Os requisitos foram levantados levando em consideração as opiniões de monitores que trabalham na administração de redes de computadores e suporte nos vários Departamentos da Universidade, e de profissionais que atuam no mercado de trabalho em empresas dos ramos mais diversos, desde postos de gasolina até transportadoras. Foram feitas entrevistas orais com perguntas sobre o dia-a-dia de cada organização e as necessidades em relação ao suporte e à documentação da rede. Isso foi feito para tornar o produto deste trabalho o mais adaptavel possível, nos diversos ambientes onde ele possa ser utilizado. Durante a análise dos requisitos levantados foram levados em consideração o público-alvo do sistema, e a entrada e saída dos dados. Também foram considerados os possíveis relatórios que seriam gerados a partir das informações e quem poderia acessar tais informações, incluindo um sistema de permissões por grupo. Terminada a análise dos requisitos foram definidas orientações para o desenvolvimento do projeto. Elas são apresentadas na forma de diretrizes de desenvolvimento, a saber:

- Todo o código gerado deve ser distribuído de acordo com a licença GNU *General Public License*, da mesma forma que a documentação gerada deve ser distribuída de acordo com a licença GNU *Free Documentation License*, segundo os conceitos e liberdades apresentdas no Capítulo 4.

- Os métodos que retornam valores dos atributos são iniciados por “get” e os métodos que alteram os valores dos atributos são iniciados por “set”.

- Os métodos devem conter verbos no infinitivo, lembrando as ações relacionadas com o método em questão. Ex.: “validar”.

- Todo método ou atributo deve ser precedido de comentário para sua identificação. Também devem ser incluidos comentários no interior do código para fácil entendimemto de partes relevantes e de sua estrutura.

- O nome dos identificadores deve seguir as seguintes diretrizes:

[8] *Computer Aided Software Engeneering* ou Engenharia de *software* auxiliada por computador

  - O nome do identificador deve sempre começar com letra minúscula, exceto as constantes, que tem todas as letras maiúsculas.
  - Quando o identificador tiver mais de uma palavra, a primeira letra de cada palavra, a partir da segunda, deve ser maiúscula. Deve ser evitado o uso de '_', '.' ou '-' como separador de palavras.

- Todos os nomes de classes devem ser iniciados com "cl". Ex.: "clUsuario".
- Os nomes dos arquivos devem conter somente letras minúsculas, sem acentos, pontuação, ou 'ç'.
- O arquivo que define uma classe deve ter o nome da classe, com extensão ".php". Ex.: "clusuario.php".
- Todo arquivo de código fonte deve conter comentários iniciais sobre sua função, licença e data de criação.
- Todo código-fonte produzido deveser desenvolvido utilizando orientação à objetos, conforme os conceitos apresentados no Capítulo 5

Além disso ficou clara a necessidade de dividir o programa em três camadas básicas, para que fossem alcançadas as metas de portabilidade. São elas:

- Camada de dados, ou banco de dados;
- Camada de aplicação, ou regra do negócio;
- Camada de apresentação, ou *interface*.

# Capítulo 7

# Modelagem do sistema

Neste capítulo serão apresentados o diagrama de caso de uso, diagrama de classe, diagrama de navegação e modelo do banco de dados gerados a partir dos requisitos levantados de acordo com o descrito no capítulo 6.

## 7.1 Diagramas de caso de uso

Diagrama de casos de uso é um diagrama que mostra a interação de um ou mais usuários com um ou mais sistemas (mais informações em [Fowler e Scott 2003]). Foi modelado o diagrama casos de uso, que foi fracionado em três, para melhor detalhamento. A Figura 7.1 mostra a interação do usuário da rede com o sistema. Já a Figura 7.2 modela a interação do membro da equipe técnica com a ferramenta. Finalmente a Figura 7.3 mostra como os administradores interagem com o sistema. Foram modelador três atores para o diagrama:

- **Usuário:** é o usuário comum, que acessa a ferramenta para basicamente fazer um chamado técnico e consultar chamados anteriores.
- **Técnico:** membro da equipe de suporte, têm como função atender os chamados e gerenciar os equipamentos. Alguns deles tem acesso ao gerenciamento de servidores e grupos.
- **Administrador:** tem como papel administrar os usuários, grupos, habilidades e as permissões. Tem poder total sobre o sistema.

### 7.1.1 Casos de uso envolvendo o ator “usuário”

O ator “usuário” pode realizar os seguintes casos de uso (Figura 7.1):

- **Fazer novo chamado:** o usuário está com dificuldades e acessa a ferramenta para fazer um chamado técnico ou solicitação de serviço à equipe de suporte. Este caso de uso inclui implicitamente o caso de uso ”Atribuir chamado”;

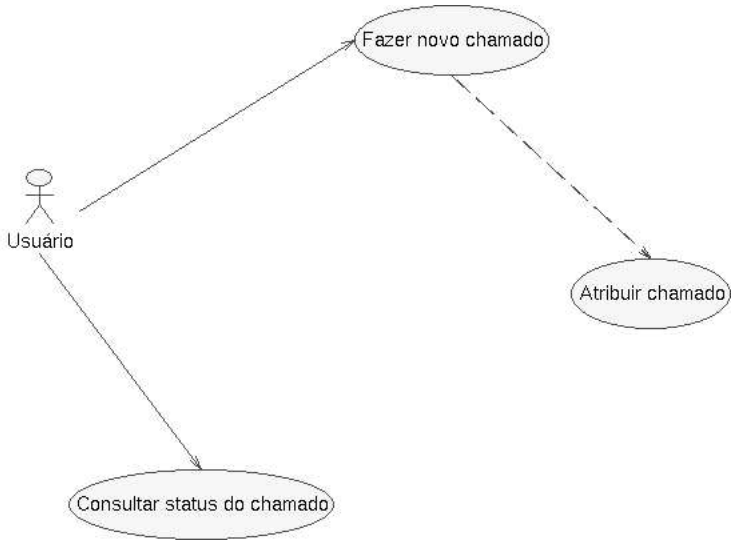


**Figura 7.1:** Diagrama de caso de uso com ator ”usuário”.

- **Atribuir chamado:** o sistema deve, a cada novo chamado, atribui-lo automaticamente a um dos técnicos, de acordo com a habilidade que está associada com cada um dos técnicos no sistema. Isso visa evitar que chamados sejam desconsiderados pelos técnicos, ou atribuido a alguém que não possa fazê-lo;

- **Consultar chamado:** O usuário pode, a qualquer momento, consultar a situação de um chamado através do identificador fornecido na hora da sua criação no sistema.

### 7.1.2 Casos de uso envolvendo o ator “técnico”

O ator “técnico” pode realizar os seguintes casos de uso (Figura 7.2):

- **Atender chamado:** o técnico deve atender os chamados a ele atribuídos pelo sistema. É muito importante que ele registre todas as informações relevantes do atendimento no sistema, e mantenha atualizado o *status* do chamado;

- **Gerenciar computadores:** o técnico pode alterar as características de um computador, componente ou periférico, de forma a manter as informações contidas no sistema atualizadas em relação à situação real dos equipamentos;

- **Gerenciar servidores:** o técnico pode atualizar ou visualizar informações sobre os servidores, e também visualizar ou incluir registros no histórico

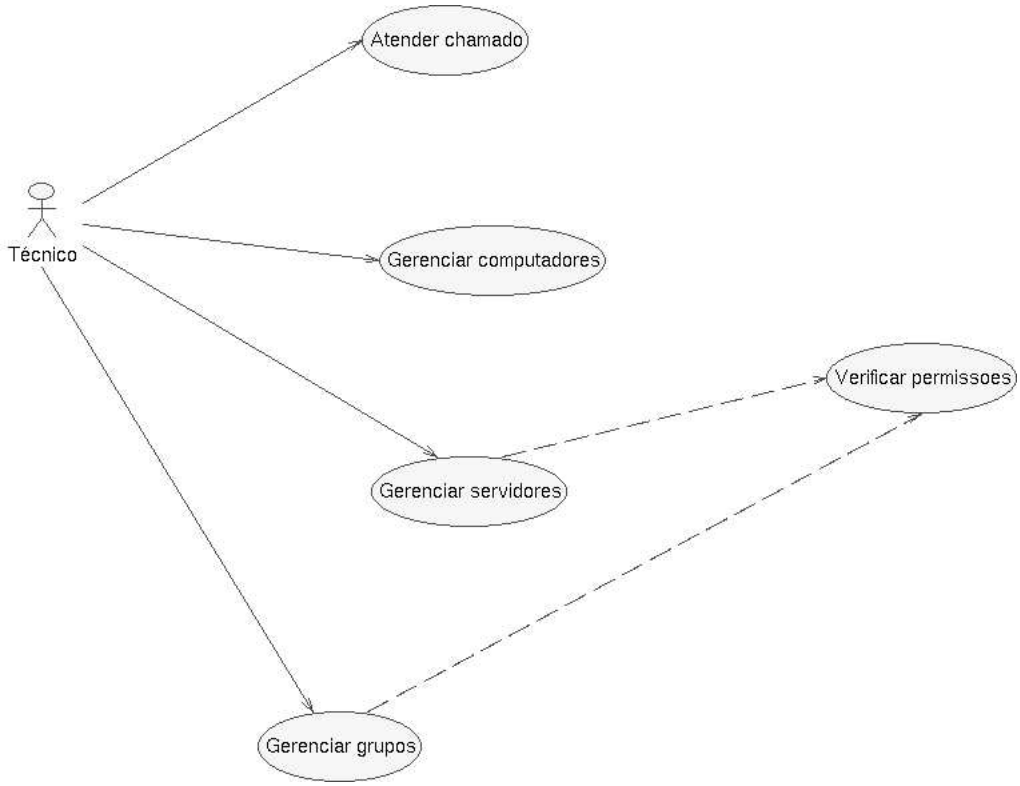


**Figura 7.2:** Diagrama de caso de uso com ator "técnico".

de cada um deles. Esse caso de uso inclui implicitamente o caso de uso "Verificar permissões";

- **Gerenciar grupos:** o técnico pode alterar as propriedades de grupos, de acordo com suas atribuições. Esse caso de uso inclui implicitamente o caso de uso "Verificar permissões".
- **Verificar permissões:** o sistema deve, antes de conceder permissão para a execução para uma determinada ação, verificar se o usuário tem permissão de executar tal ação. As tentativas de acesso não-autorizado devem ser reportadas ao administrador.

Os casos de uso "Atender chamados" e "Gerenciar computadores" não incluem o caso de uso "Verificar permissões" porque todos os técnicos têm permissão para executar os casos de uso "Gerenciar computadores" e "Atender chamados".

### 7.1.3 Casos de uso envolvendo o ator "administrador"

O ator "administrador" pode realizar os seguintes casos de uso (Figura 7.3):

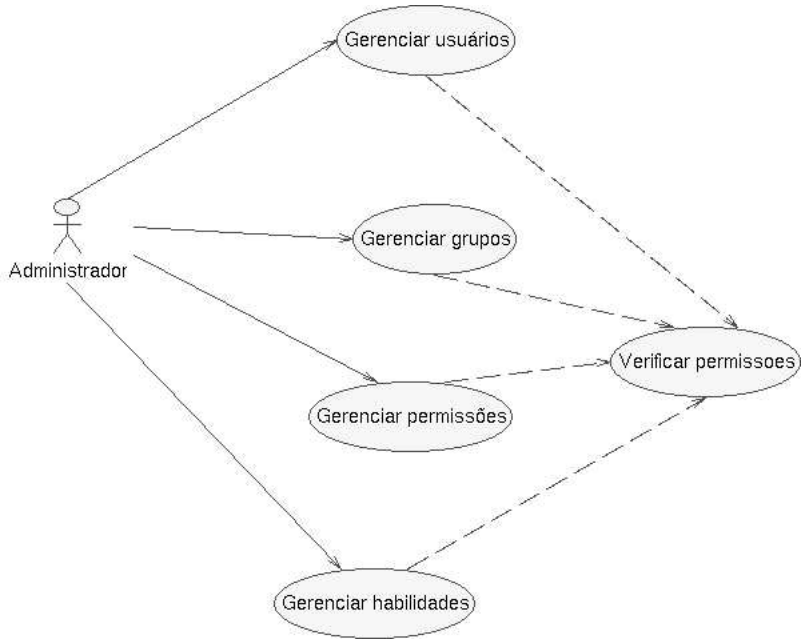


**Figura 7.3:** Diagrama de caso de uso com ator "administrador".

- **Gerenciar usuários:** o administrador pode criar, remover ou modificar contas de técnicos no sistema. Esse caso de uso inclui implicitamente o caso de uso "Verificar permissões".
- **Gerenciar grupos:** o administrador pode alterar as propriedades de todos os grupos. Esse caso de uso inclui implicitamente o caso de uso "Verificar permissões";
- **Gerenciar permissões:** é tarefa exclusiva do administrador do sistema gerenciar as permissões concedidas aos técnicos e grupos cadastrados no sistema. Esse caso de uso inclui implicitamente o caso de uso "Verificar permissões";
- **Gerenciar habilidades:** o administrador deve administrar as habilidades, ou conjuntos de itens que caracterizam um técnico ou grupo de técnicos. Esse caso de uso inclui implicitamente o caso de uso "Verificar permissões";
- **Verificar permissões:** A mesma definição da Seção 7.1.2.

## 7.2 Diagrama de classes

Após serem definidos os casos de uso, as classes do sistema foram modeladas. A Figura 7.4 exibe as classes e os relacionamentos. Nessa Figura não são exibidas

as informações sobre os atributos e os métodos de cada uma das classes. Isso foi feito para tornar mais visíveis os relacionementos. No Apêndice A as classes são mostradas em detalhes. Segundo [Fowler e Scott 2003] um diagrama de classe descreve os tipos de objetos no sistema e os vários relacionamentos estáticos entre eles. Esse diagrama corresponde à camada de aplicação do sistema.

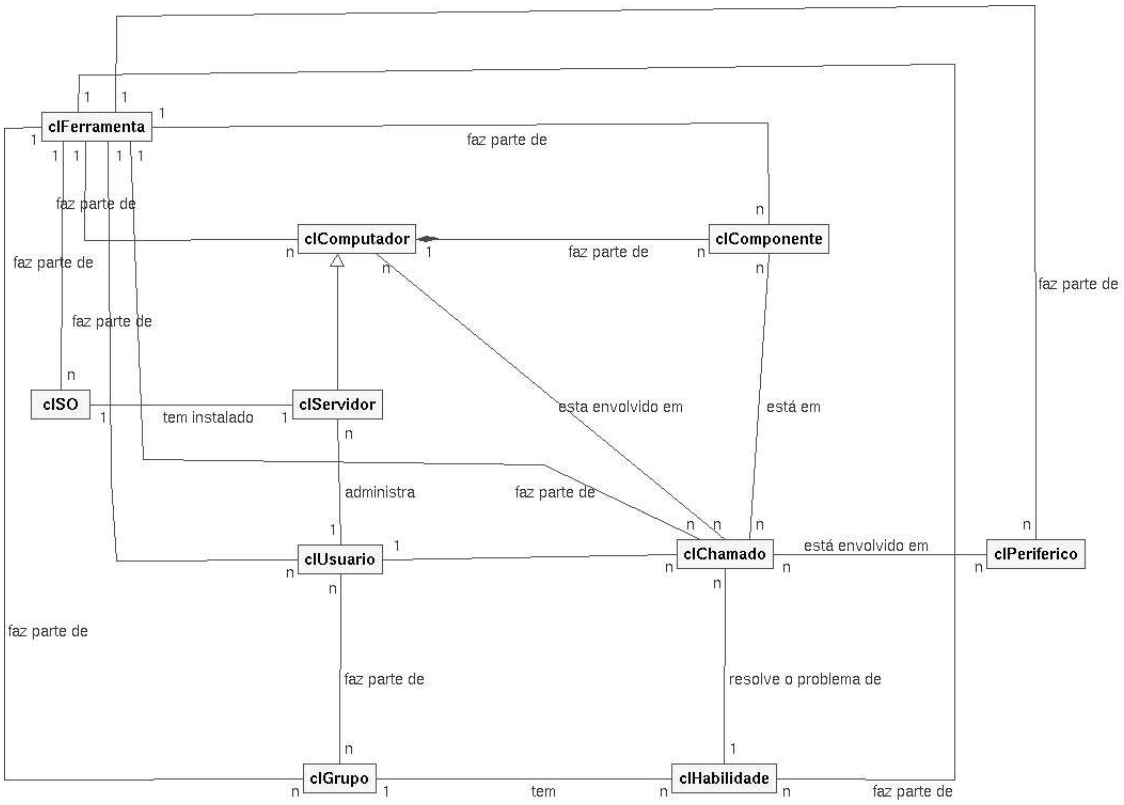


**Figura 7.4:** Diagrama de classe do sistema.

## 7.3 Modelagem da base de dados

A modelagem da base de dados foi feita utilizando a ferramenta definida no Capítulo 6. Os relacionamentos **1:n** foram representados por chaves estrangeiras. Já os relacionamentos **m:n** foram representados utilizando uma entidade fraca, cujo nome foi padronizado "*tabela1_has_tabela2*". Essa entidade fraca tem um relacionamento **1:n** e **1:m** com as duas entidades fortes. A Figura 7.5 mostra o modelo relacional da camada de dados do sistema. Os losangos que estão nos relacionamentos simbolizam a cardinalidade da relação. A cor preta significa *n* e a cor branca 1.

## 7.4 Diagramas de navegação

Neste trabalho, ao invés dos diagramas de sequência, foram feitos diagramas de navegação, como proposto por [Dias 2003], por serem mais adequados a aplicações voltadas à *web*, devido à sua simplicidade e facilidade de entendimento. Nesses diagramas cada caixa é uma página, e cada uma das setas determina um dos possíveis destinos a partir daquele ponto. Foram feitos dois diagramas de navegação. O diagrama da Figura 7.6 mostra o modelo de navegação para o usuário da rede, que pode consultar ou fazer chamados técnicos. O diagrama da Figura 7.7 mostra o modelo de navegação para a área de acesso restrito aos técnicos e administradores do sistema.

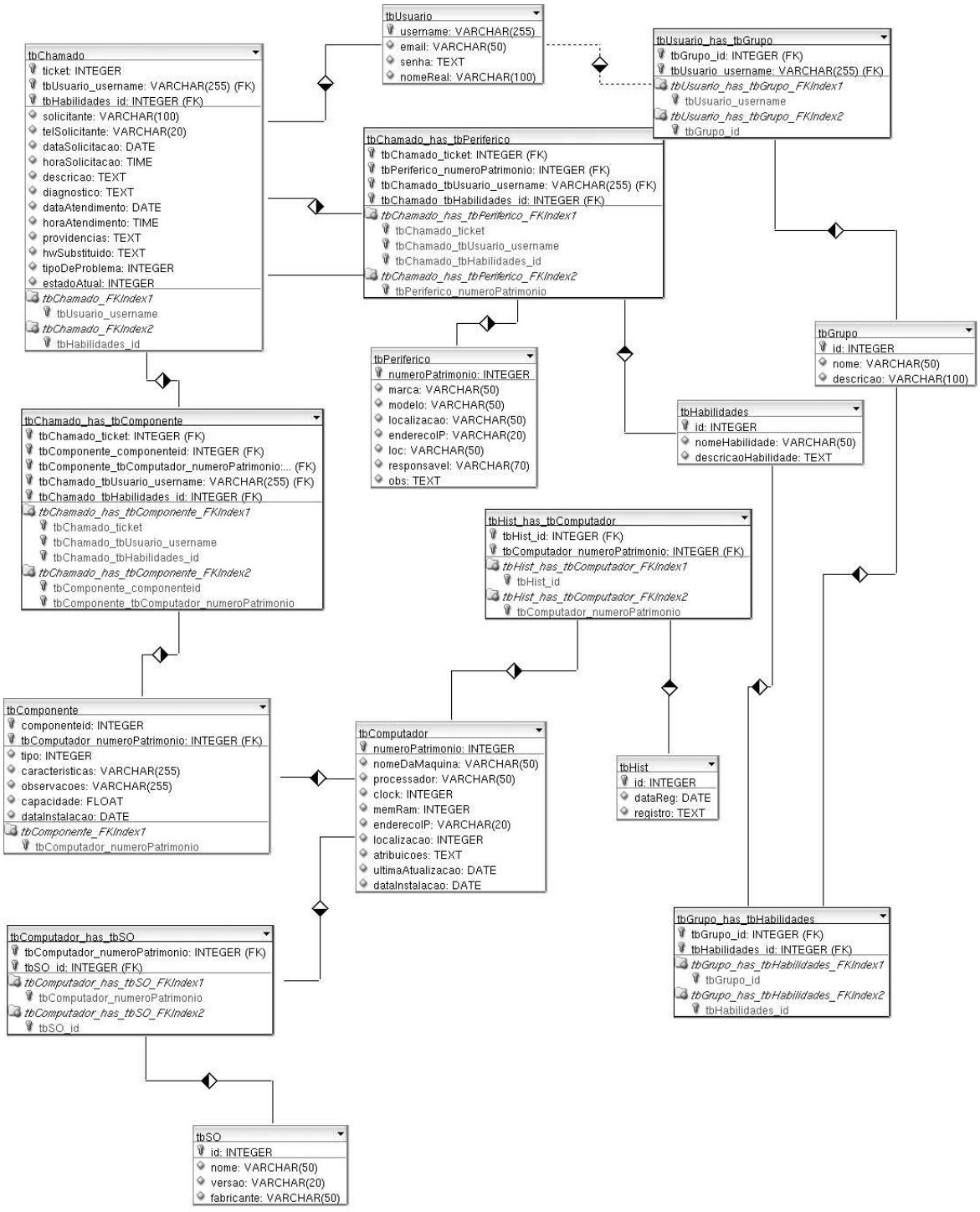


**Figura 7.5:** Modelo de dados do sistema.

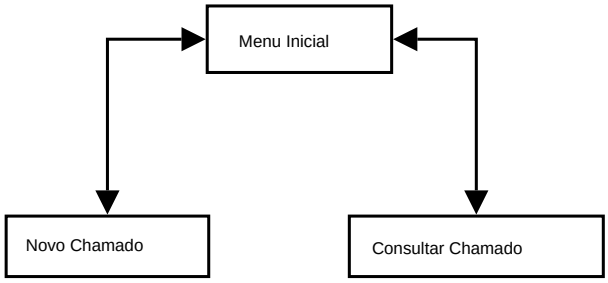


**Figura 7.6:** Diagrama de navegação da seção aberta aos usuários.

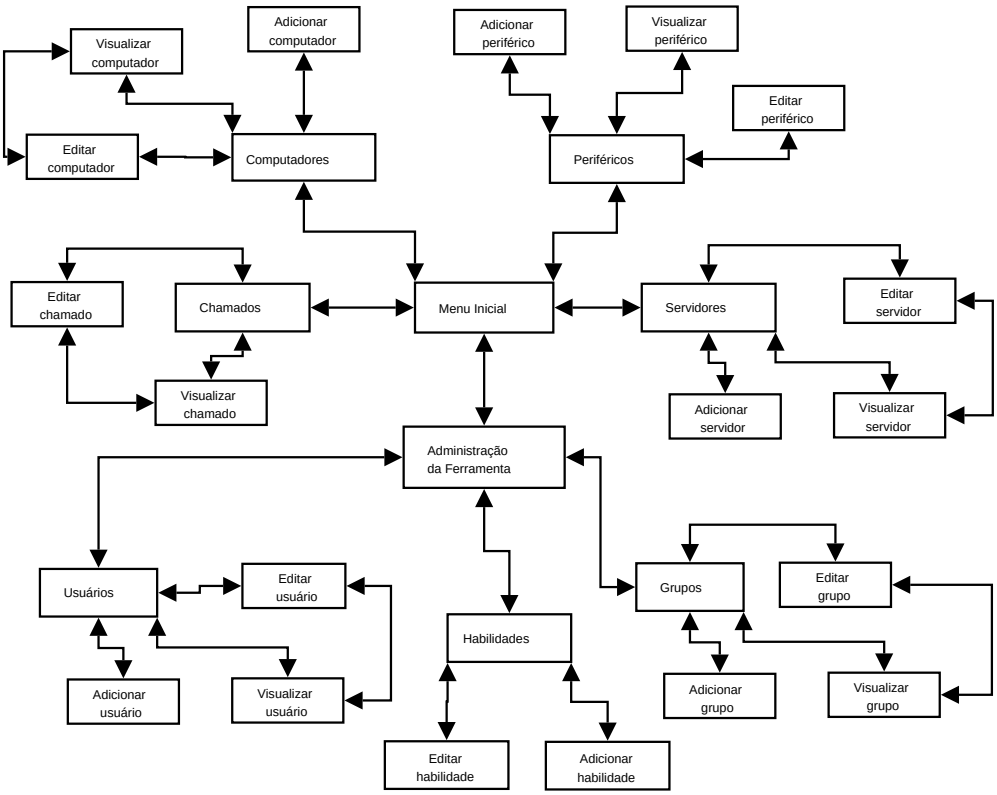


**Figura 7.7:** Diagrama de navegação da área restrita a técnicos e administradores.

# Capítulo 8

# Resultados e discussão

Este capítulo destina-se a mostrar os resultados obtidos ao final do trabalho. Foi dada prioridade à parte do *software* que cuida do atendimento aos chamados, por ser o cerne de todo o sistema. É importante salientar que foi privilegiado o desenvolvimento da camada de aplicação em detrimento à camada de apresentação devido ao fato de a *interface* do programa, mesmo sendo fundamental, pode ser implementada com um grau maior de simplicidade sem afetar o funcionamento da aplicação.

O sistema está dividido em duas partes: uma que tem acesso público, onde os usuários podem fazer e consultar seus chamados, e outra de acesso restrito a membros da equipe técnica, onde são gerenciados os chamados, os equipamentos e a própria ferramenta. Na seção aberta aos usuários, existe uma página inicial, como mostra a Figura 8.1, onde estão os *links* para as seções onde é possível fazer um novo chamado e verificar a situação de um chamado. A página onde é possível fazer um novo chamado tem basicamente os campos que o usuário deve preencher e algumas pequenas instruções de preenchimento, como é possível observar na Figura 8.2. Após enviar o chamado, o usuário recebe um *ticket*, um identificador único de cada chamado. A página onde são consultados os chamados tem apenas o campo onde deve ser inserido o *ticket* e o resultado é exibido conforme a Figura 8.3.

Já na região reservada aos técnicos, logo de início, é solicitado o nome de usuário e a senha a quem está acessando a ferramenta. Após o *login* o técnico vê o menu principal da ferramenta, como mostrado na Figura 8.4. Desse menu é possível escolher entre chamados, computadores, servidores, periféricos, e área administrativa. Ao escolher chamados, serão exibidos os chamados atribuídos ao técnico, conforme Figura 8.5. O técnico pode abrir um chamado para completar as informações relativas ao atendimento e alterar o seu *status*, sendo vedada a alteração dos dados preenchidos pelo usuário que efetuou o chamado. Ao acessar a seção computadores, será exibida uma lista com todos os computadores cadastrados, conforme exibido na Figura 8.6. O tecnico pode cadastrar um novo computador

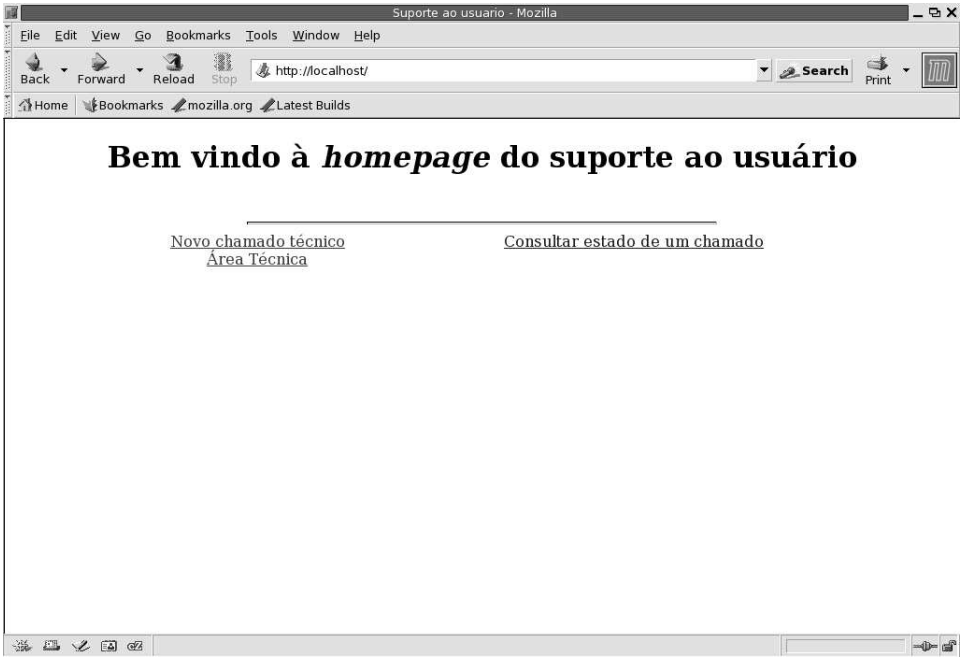


**Figura 8.1:** Página inicial da seção aberta aos usuários.

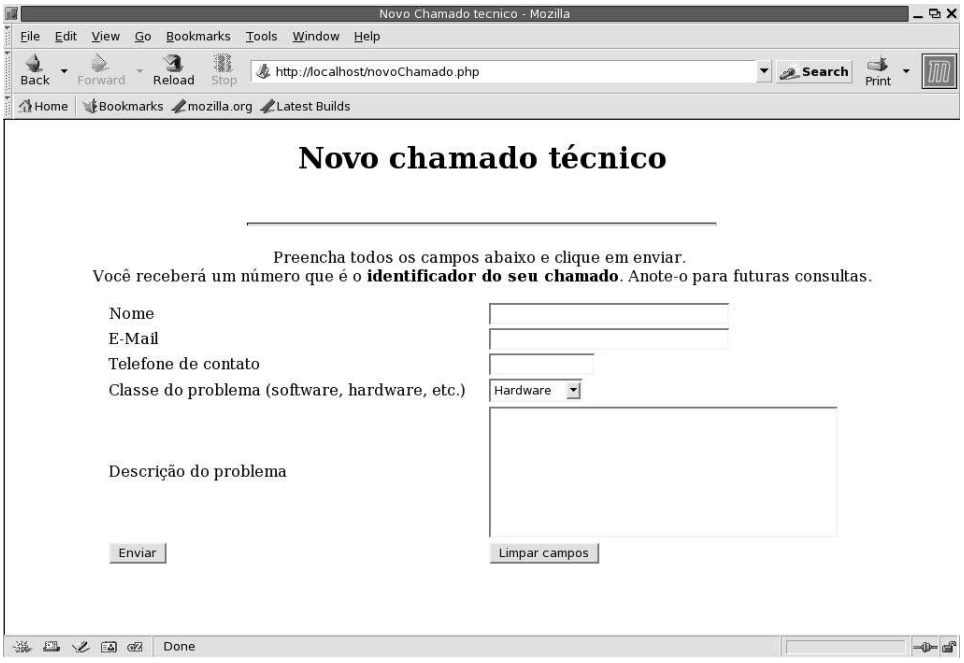


**Figura 8.2:** Página onde são cadastrados novos chamados.

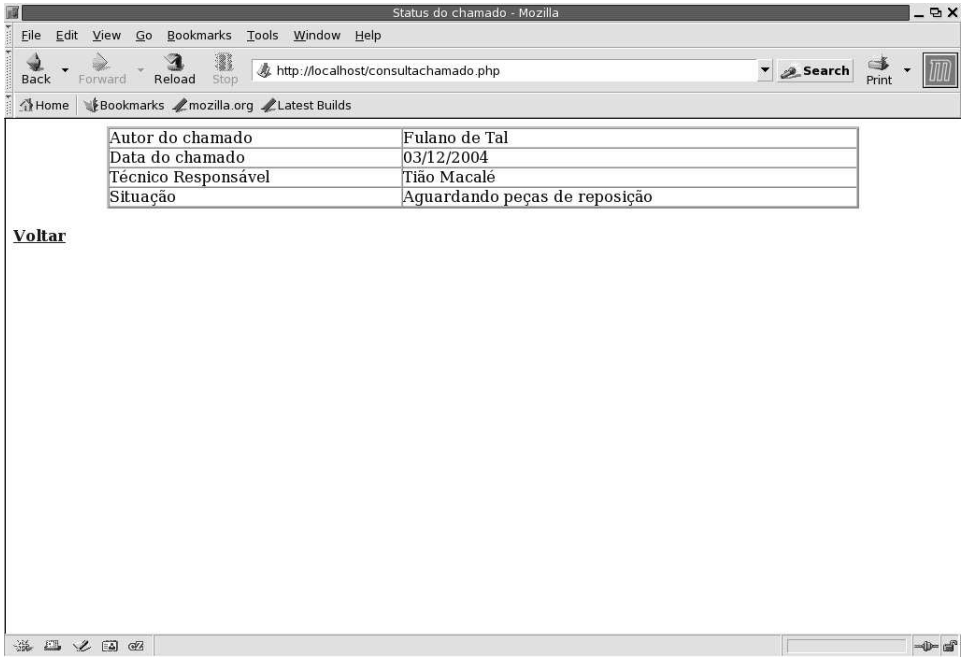


**Figura 8.3:** Resultado da consulta ao *status* de um chamado.

ou acessar qualquer computador da lista, para fazer alterações e até exclui-lo. Ao acessar a seção servidores, a navegação é muito parecida com a da seção computadores, porém com a diferença que somente os técnicos que fazem parte do grupo "servidor" terão acesso a essa seção. Outra característica da seção servidores é o histórico onde o técnico deve relatar todas as ações tomadas em relação a cada servidor. A seção periféricos trata dos periféricos, como monitores, impressoras e *scanners* que estão cadastrados no sistema. Da mesma forma que nas outras seções, é possivel modificar, criar e excluir periféricos do sistema. A seção administração é de acesso restrito aos membros do grupo "admin". Nessa seção, é possivel alterar o cadastro de técnicos, como mostra a Figura 8.8, o cadastro de grupos e o cadastro de habilidades, além de informações gerais sobre o sistema. O único relatório implementado é o de chamados em aberto, que lista aos administradores informações sobre os chamados que ainda não foram finalizados (Figura 8.7).

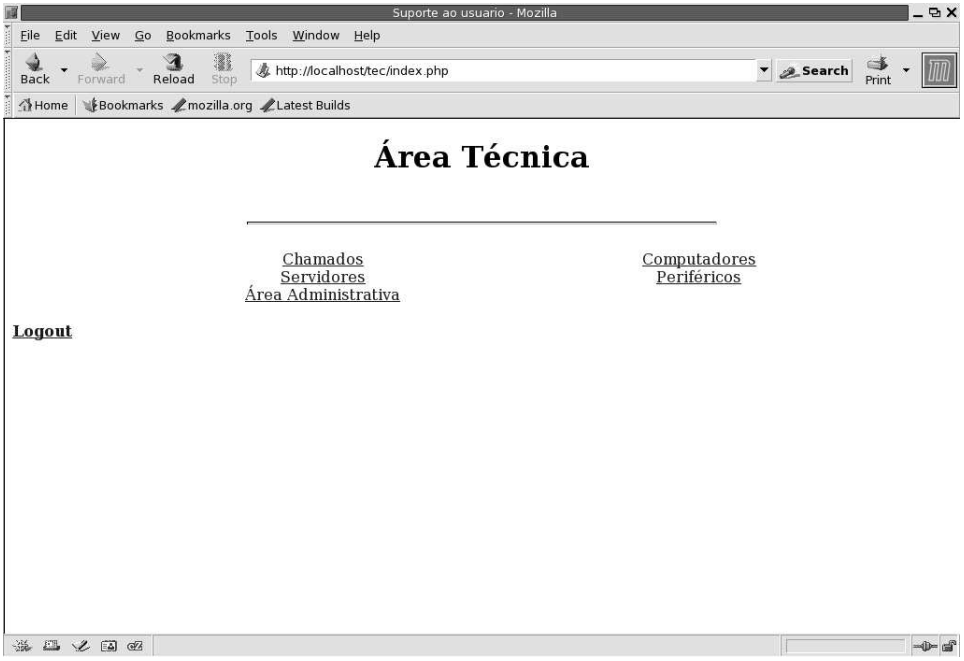


**Figura 8.4:** Página inicial da área técnica.

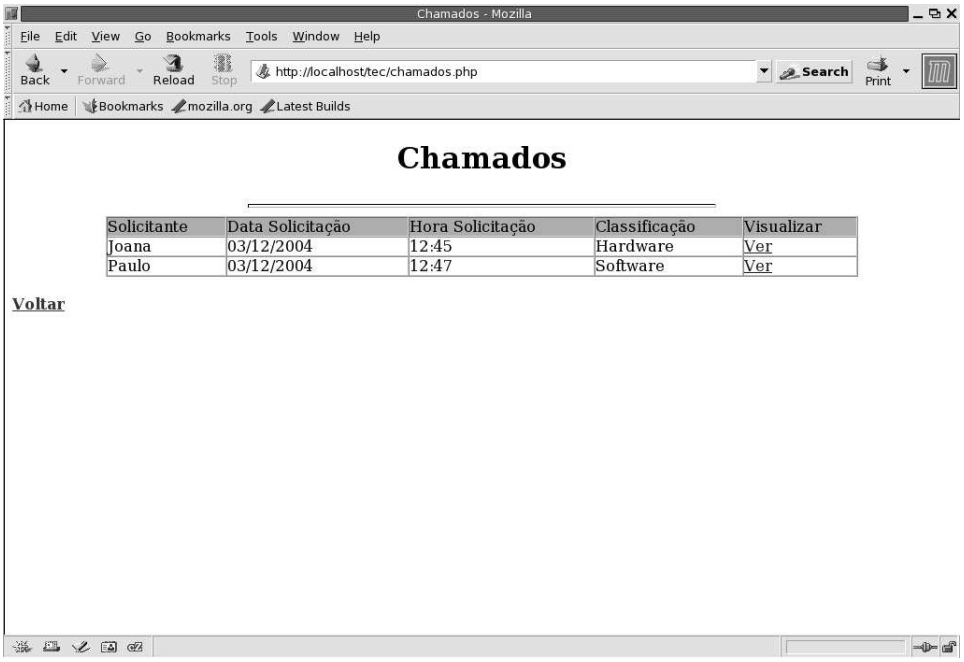


**Figura 8.5:** Lista dos chamados atribuidos a um técnico.

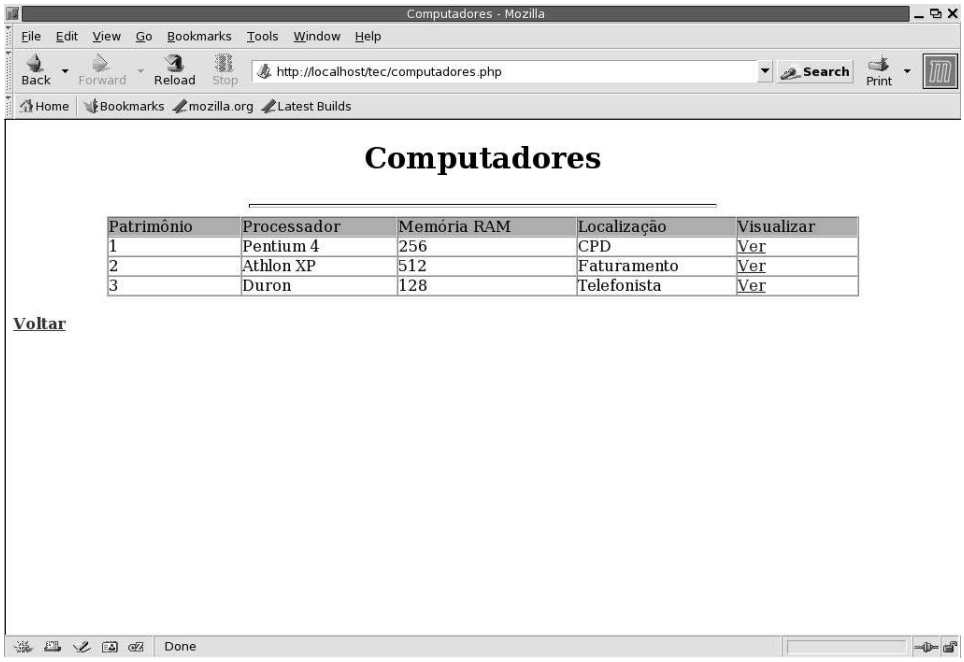


**Figura 8.6:** Lista dos computadores cadastrados na ferramenta.

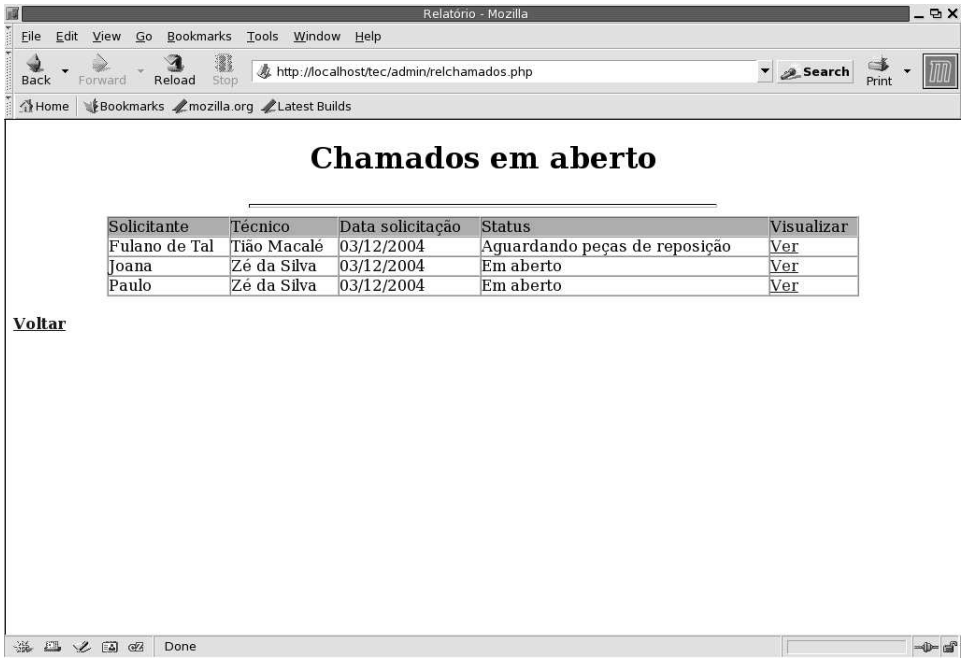


**Figura 8.7:** Lista dos computadores cadastrados na ferramenta.

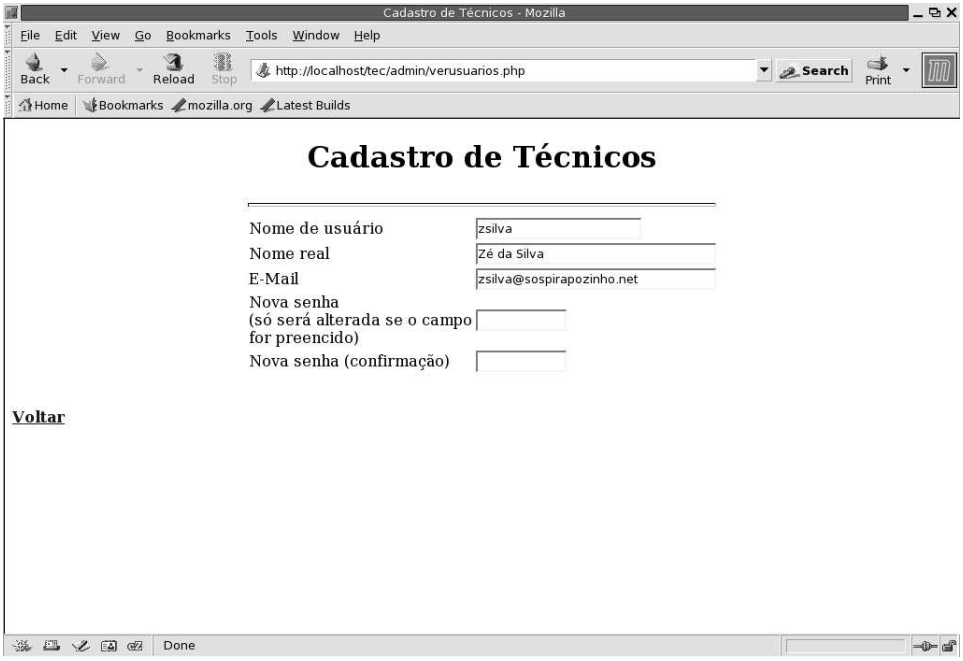


**Figura 8.8:** Cadastro de técnicos.

# Capítulo 9

# Conclusões

Este trabalho não tem a pretensão de ser a solução final para o problema do suporte, mesmo porque é composto de vários outros fatores além do gerenciamento de chamados técnicos e equipamentos. Uma boa solução para o suporte depende de comprometimento da equipe envolvida no processo, da colaboração dos usuários e de investimentos por parte da direção da instituição, dentre outros. Porém uma ferramenta que organiza e documenta o atendimento aos usários, além de fazer o inventário de equipamentos é de grande utilidade, principalmente para obter medidas da qualidade do atendimento. Com essas medidas em mãos é possível buscar a melhoria contínua dos processos de atendimento. Nesse aspecto a ferramenta atinge boa parte de seus objetivos, principalmente os expostos na Seção 1.2.

As necessidades dos usuários mudam muito rapidamente, principalmente depois do surgimento da *internet*. Não é viável, com essa velocidade toda, manter o suporte totalmente atualizado, mas cursos de reciclagem, oficinas, visitas a congressos, feiras e exposições ajudam as equipes a se manterem sempre inseridas no contexto da tecnologia da informação. Isso faz com que eles tenham mais facilidade para prever mudanças nos usuários, sistemas e demais componentes do ambiente. Também é importante investir em treinamento para os usuários. Vários casos de problemas são causados por imperícia do usuário. Treinamentos sobre a mecânica de utilização e boas práticas ao computador, netiqueta por exemplo, reduziria o número de ocorrências.

Outro ponto importante é a inserção de novas ferramentas de trabalho no dia-a-dia da equipe de suporte. Não é coerente que sejam utilizados recursos como caderno ou bloco de anotações para gerenciar o atendimento em um ambiente onde as pessoas conhecem e tem domínio sobre tecnologias de informação. Poderiam ser utilizados, por exemplo, computadores de mão ou até celulares para acessar informações sobre os equipamentos, os chamados e a rede. Isso tornaria o suporte mais ágil, otimizando o tempo para realizar as tarefas, sem desperdícios. Outro ponto seria a utilização de métricas do atendimento, como tempo médio de aten-

dimento. Com esses dados podem ser feitos estudos sobre a qualidade do serviço prestado, e estabelecer metas buscando melhorias.

Enfim, suporte é uma área delicada, que exige atenção e dedicação, principalmente de e para quem trabalha nessa área. O ponto mais importante é manter a estrutura de suporte sempre atualizada. Seja com novas tecnologias e ferramentas ou conhecimeto. O suporte deve sempre estar apto a acompanhar as mudanças, e, principalmente, trazê-las para o dia-a-dia do trabalho.

## 9.1 Trabalhos futuros

Deve ser feita a tradução de todo o código fonte, comentários, e documentos gerados durante o desenvolvimento do sistema para o inglês. Isso facilitaria a criação de um grupo de desenvolvimento mais amplo, caso o projeto evolua e seja disponibilizado para a comunidade de *software* livre contribuir. É muito importante também reestudar os casos de uso, para assegurar que o desenvolvimento do sistema siga uma linha coerente com a sua proposta. Uma mudança interessante seria o desmembramento da camada de apresentação em duas: uma camada para tratar contextos, que são as estruturas do documento, e outra que cuidaria somente da apresentação. Existe a possibilidade de melhoria na atribuição automática de chamados aos técnicos que hoje é feita escolhendo aleatoriamente um técnico dentre os que estão nos grupos que têm determinada habilidade. Poderia ser utilizado um critério de balanceamento, para evitar técnicos sobrecarregados. Outra melhora de grande importância seria a criação de mais relatórios e estabelecer uma política de acesso basado no sistema de permissão implementado.

Pode-se ainda expandir a automação, criando um sistema especialista em resolução de problemas. Com esse sistema o susário pode navegar por um conjunto de perguntas que o leve à resolução do problema. Isso evitaria que situações já conhecidas cheguem à equipe de suporte e o usuário torne-se mais independente e auto-confiante na utilização dos equipamentos e do *software*. Além disso, situações mais comuns podem ser convertidas em FAQs do sistema.

# Referências Bibliográficas


[Ambler 1998]AMBLER, S. W. *Análise e Projeto Orientado a Objeto, vol.2*. [S.l.]: IBPI Press, 1998.

[Bakken et al. 2004]BAKKEN, S. S. et al. *PHP Manual*. 2004. Visitado em 12 de Junho de 2004. Disponível em: <http://www.php.net/manual/pt_BR>.

[Cardoso 2003]CARDOSO, C. *UML na prática: Do problema ao sistema*. [S.l.]: Editora Ciência Moderna, 2003.

[Cordani e Nagel 1995]CORDANI, J. R.; NAGEL, K. P. Redefining user support: Shifting paradigms insted of blame. *ACM Winning the Network Game*, 1995.

[Coventry e Kane 1993]COVENTRY, L. M.; KANE, T. B. The automation of helpdesks. *ACM - Inteligent User Interfaces'93*, 1993.

[Diagle 2003]DIAGLE, R. Student workers - the heart of help desk. *ACM - SIGUCCS'03*, 2003.

[Dias 2003]DIAS, S. P. *E-DUC@RE:Proposta de Ambiente de Aprendizagem Suportado pela Web para Cursos de Nível Superior Oferecidos a Distância*. 2003.

[Diem 1998]DIEM, R. E. Using the technology you support - automating the help desk. *ACM - SIGUCCS XXVI*, 1998.

[Fowler e Scott 2003]FOWLER, M.; SCOTT, K. *UML Essencial*. [S.l.]: Editora Bookman, 2003.

[FSF 2003]FSF. *O que é Software Livre?* 2003. Visitado em 26 de Maio de 2004. Disponível em: <http://www.gnu.org/philosophy/free-sw.pt.html>.

[Ghezzi, Jahzayeri e Mandrioli 1991]GHEZZI, C.; JAHZAYERI, M.; MANDRIOLI, D. *Fundamentals of Software Engeneering*. [S.l.]: Prentice Hall, 1991.

[Govindarajulu 2002]GOVINDARAJULU, C. The status of helpdesk support. *Communications of the ACM*, 2002.

[Link 2002]LINK, M. Transforming support: From helpdesk to information center. *ACM - SIGUCCS'02*, 2002.

[McRitchie 2003]MCRITCHIE, K. Student staff n. an energetic, brilliant, creative, talented potential rolled into a food-driven, sleep-deprived, unmotivated creature who wears dirty clothes. *ACM - SIGUCCS'03*, 2003.

[Nemeth et al. 1995]NEMETH, E. et al. *Unix System Administrator Handbook*. Second edition. [S.l.]: Prentice Hall, 1995.

[OSI 2004]OSI. *Frequently Asked Questions*. 2004. Visitado em 27 de Maio de 2004. Disponível em: <http://www.opensource.org/advocacy/faq.php>.

[OSI 2004]OSI. *The Open Source Case for Hackers*. 2004. Visitado em 27 de Maio de 2004. Disponível em: <http://www.opensource.org/advocacy/case_for_hackers.php>.

[OSI 2004]OSI. *The Open Source Definition, version 1.9*. 2004. Visitado em 27 de Maio de 2004. Disponível em: <http://www.opensource.org/docs/definition.php>.

[Pear 1988]PEAR, M. R. A new model for user services: Distributed support. *ACM - SIGUCCS XVI*, 1988.

[Ragget 2002]RAGGET, D. *Getting started with HTML*. 2002. Visitado em 13 de Junho de 2004. Disponível em: <http://http://www.w3c.org/MarkUp/Guide/>.

[Saini e Lavoie 1991]SAINI, S.; LAVOIE, L. Provinding user support: a response to economic restraint. *ACM - SIGUCCS XIX*, 1991.

[Saul, Black e Larsson 2000]SAUL, J.; BLACK, B.; LARSSON, E. Helpdesk.drew.edu: Home growing a helpdesk solution using open-source technology. *ACM - SIGUCCS'2000*, 2000.

[Thompson 1994]THOMPSON, M. Extending support services to antecipate growing user needs. *ACM - SIGUCCS XXII*, 1994.

[W3C 1999]W3C. *HTML 4.01 Specification*. 1999. Visitado em 13 de Junho de 2004. Disponível em: <http://www.w3.org/TR/html401/>.

[Zandstra 2000]ZANDSTRA, M. *Teach yourself PHP in 24 hours*. [S.l.]: Sams Publishing, 2000.

# Apêndice A

# Detalhamento das classes do sistema

Este apêndice tem como objetivo mostrar as classes em detalhes. A Figura A.1 mostra a classe `clUsuario`, que modela um usuário da ferramenta (técnico). A Figura A.2 mostra a classe `clGrupo`, que modela os grupos de técnicos. A Figura A.3 mostra a classe `clHabilidade`, que modela as habilidades, ou conjuntos de habilidades, que cada grupo de técnicos têm. As Figuras A.4,A.5 e A.6 mostram a classe `clComputador`, que modela um computador, a classe `clComponente`, que modela um componente de um computador e a `clServidor`, que modela um servidor de rede, respectivamente A Figura A.7 mostra a classe clPeriferico, que modela os periféricos de um computador, como *scanners* e impressoras, e a Figura A.8 mostra a classe clChamado, que modela um chamado técnico. Finalmente, a Figura A.9 mostra a classe que modela o funcionamento de toda a ferramenta.

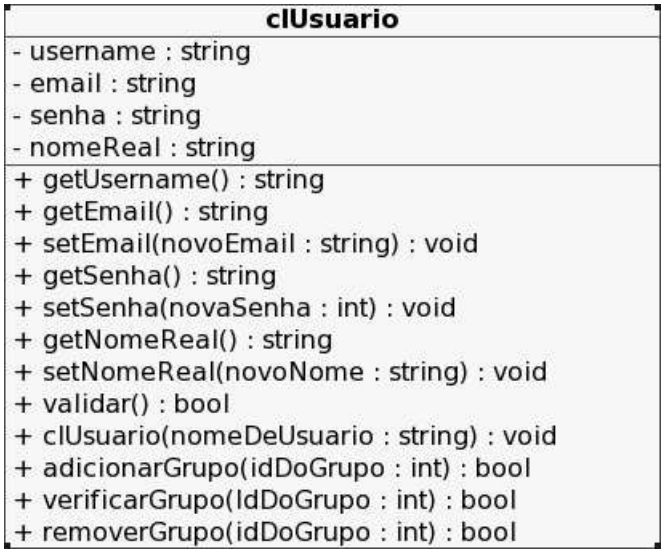


**Figura A.1:** Classe clUsuario.

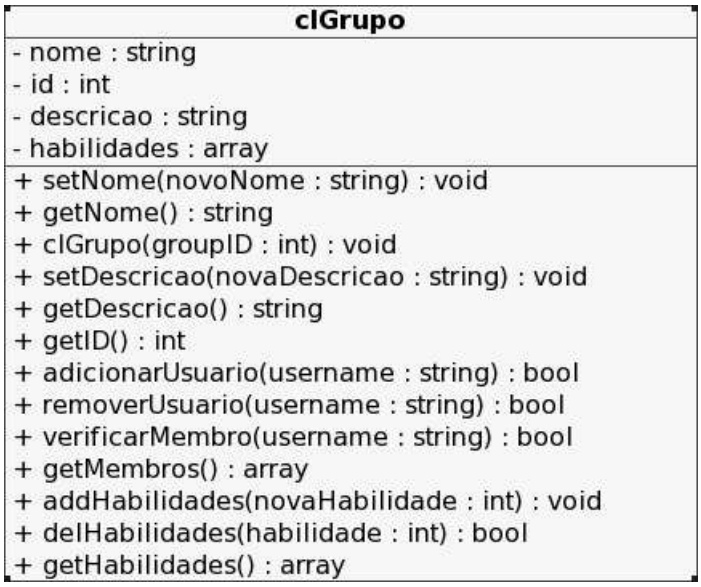


**Figura A.2:** Classe clGrupo.

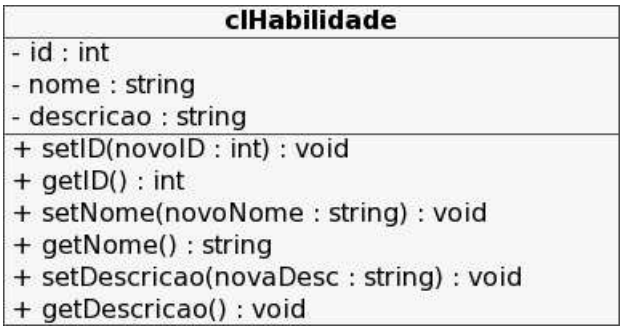


**Figura A.3:** Classe clHabilidade.

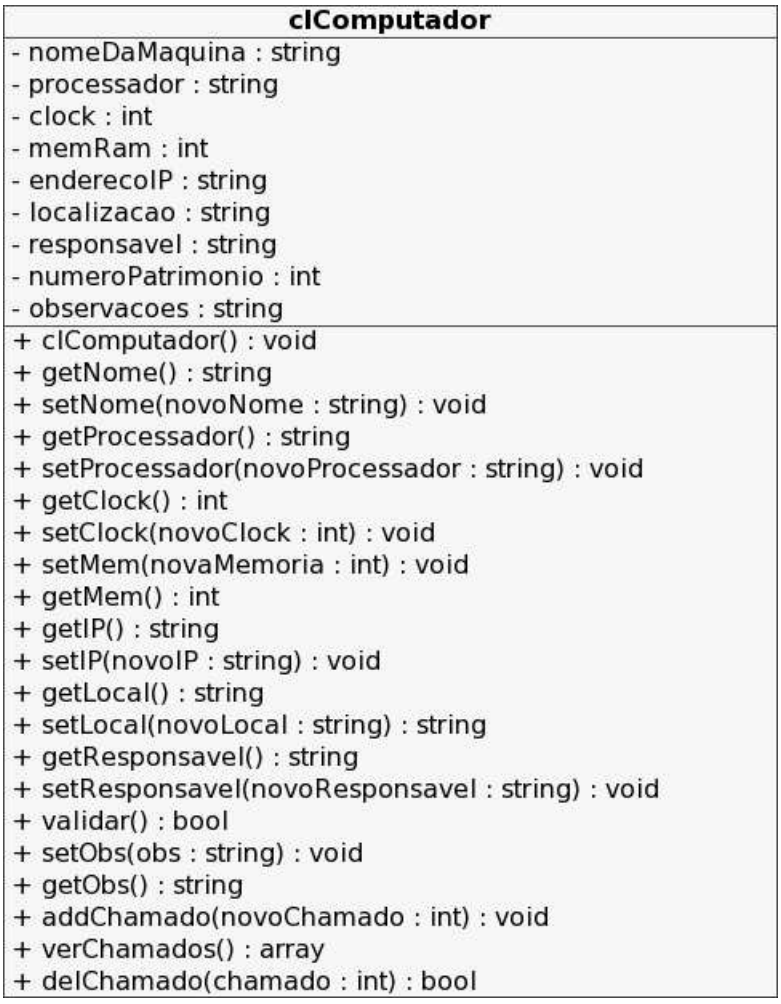


**Figura A.4:** Classe clComputador.

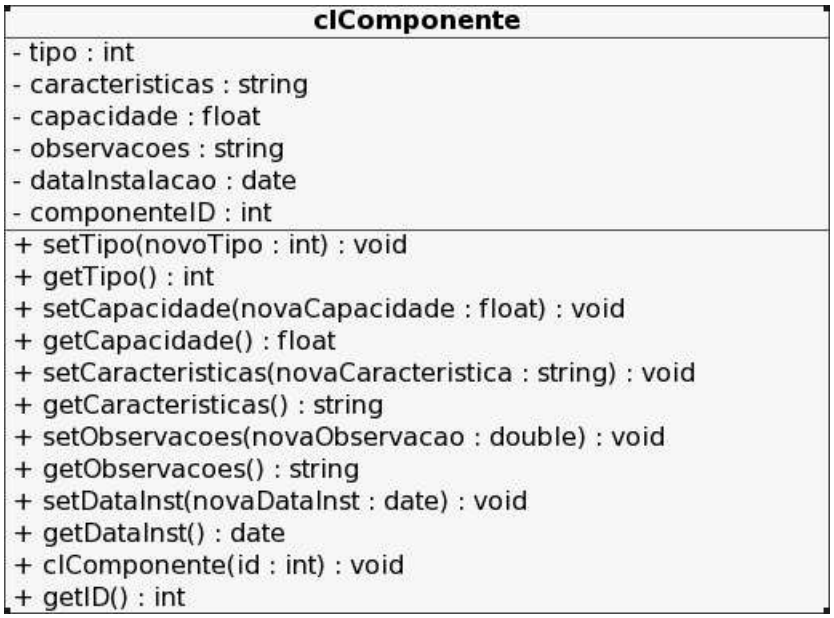


**Figura A.5:** Classe clComponente.

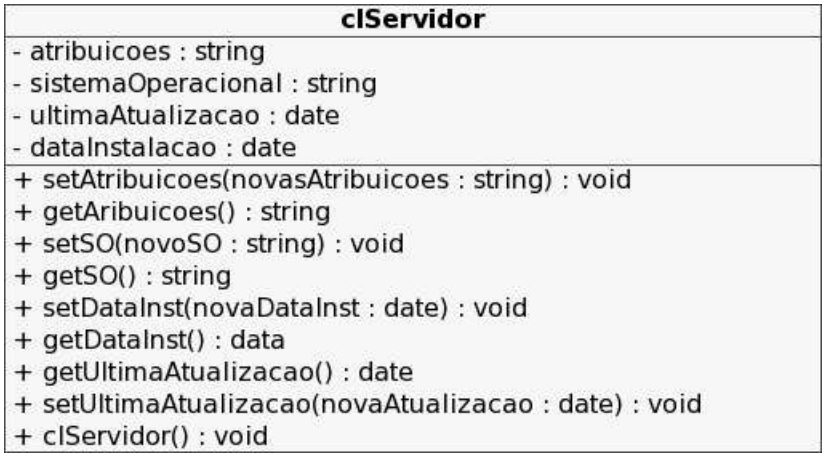


**Figura A.6:** Classe clServidor.

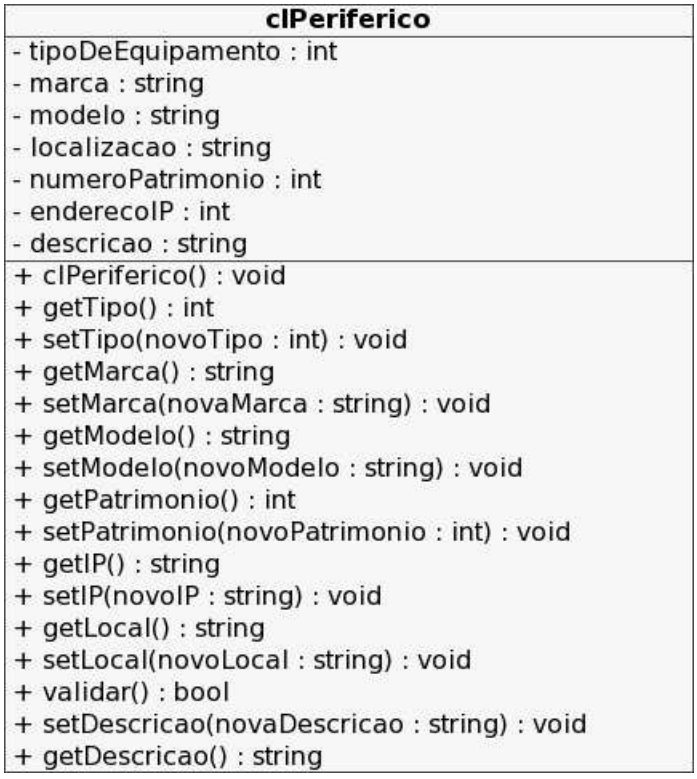


**Figura A.7:** Classe clPeriferico.

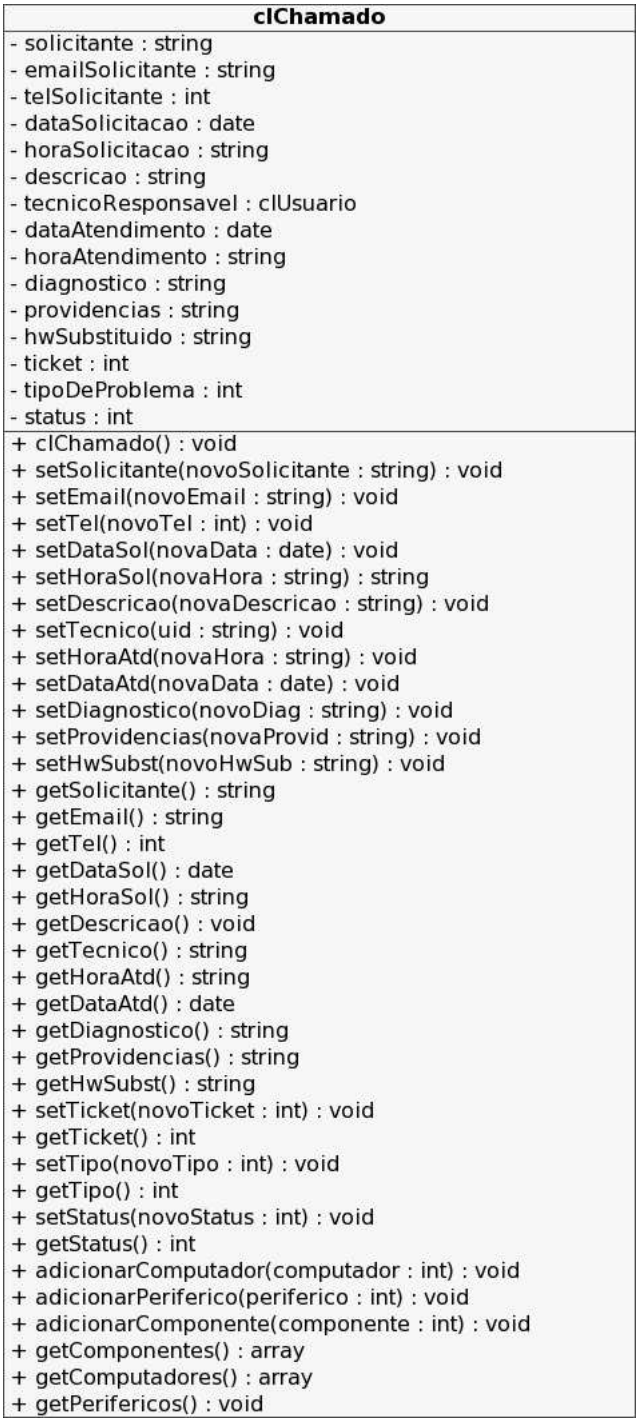


**Figura A.8:** Classe clChamado.


**Figura A.9:** Classe clFerramenta.