Modelagem e Desenvolvimento de um Sistema *Help-Desk* para a Prefeitura Municipal de Lavras - MG

Gabriel O. T. Cavalari[1], Heitor A. X. Costa[2]

Universidade Federal de Lavras – Departamento de Ciência da Computação
Caixa Postal 3037 – Lavras – MG – Brasil
[1]calvalari@comp.ufla.br, [2]heitor@ufla.br

Resumo – Sistemas *Help-Desk* podem ser definidos como sendo um setor da empresa ao qual são endereçadas questões e onde são resolvidos problemas, tendo como principal característica a de ser um sistema facilitador de informações. A proposta deste trabalho é realizar a modelagem e o desenvolvimento de um sistema *Help-Desk* para a Prefeitura Municipal de Lavras. Neste trabalho, são apresentados conceitos e análises de ferramentas Help-Desk e discutidas a arquitetura e as tecnologias deste tipo de sistema. É exposta ainda, uma descrição da funcionalidade do sistema implementado, bem com a sua modelagem.

Palavras-chave: Gestão do Conhecimento, Raciocínio Baseado Regras, Raciocínio Baseado Em Casos, *Help-Desk*

Abstract *– Help-Desk systems can be defined as a department of a company where questions are addressed and problems are resolved, the main characteristic of this type of system is to be an information facilitator. The paper objective is to accomplish the modeling and the development of a Help-Desk System to the Lavras City Hall. In this paper, Help-Desk tools are presented and analyzed, and the architecture and the technologies of this type of system are discussed. A description of functionalities of implemented system is exposed as well as its modeling.*

Key-word: Knowledge Engineering, Rules-Based Intelligence, Case-Based Intelligence, HelpDesk.

1 Introdução

Sistemas *Help-Desk* podem ser definidos como sendo um setor da empresa ao qual são endereçadas questões e onde são resolvidos problemas. Entretanto, a concepção de *Help-Desk* vai além do conceito de suporte técnico. Isso ocorre pois, se até a década de 80 a informática tinha-se uma população de usuários elitizada, após os anos 90 os computadores passaram a atender a uma população mais preocupada em adquirir conhecimentos e informações.

Neste ambiente, onde o computador é a principal ferramenta de produção, é primordial que este esteja sempre em perfeito funcionamento. Essa tarefa, porém, mostrou-se bastante complexa, pois enquanto nos anos 80 os usuários eram especialistas em informática, na década de 90 esse quadro se inverteu radicalmente. A população de usuários de computadores passou a ser composta majoritariamente por pessoas oriundas de áreas onde a informática não era fundamental e, por essa razão, possuía praticamente nenhum conhecimento sobre o uso dos computadores.

Essa mudança proporcionou a criação de um tipo de suporte capaz de atender a demanda deste novo público, pois, o que antes era visto com um contado entre especialistas, passa a ser uma forma de contado empresa-consumidor. Além disso, as empresas perceberam que era importante armazenar todos os dados decorrentes dos problemas ocorridos, assim como a maneira de solucioná-los. Desta forma, o usuário teria um atendimento mais rápido e personalizado e a empresa teria um controle maior sobre quais são os problemas mais comuns e as melhores formas de solucioná-los.

Sendo assim, os sistemas *Help-Desk* têm hoje a principal característica de ser um facilitador de informações ao usuário, não importando se esta facilidade é ou não de natureza técnica computacional. Um sistema *Help-Desk* constitui um mecanismo computacional facilitador de informação do tipo *Help-Desk* = *<Pergunta, Resposta>,* sendo tanto a *Pergunta* em apreço se refere àquela de clientes e/ou usuários quaisquer; quanto a *Resposta* do sistema se refere a um apoio informacional bem definido em seu domínio [1].

O termo *Help-Desk* pode ser utilizado tanto para o sistema em si quanto para a ferramenta de *software* utilizada por este [2]. Um sistema *Help-Desk* é composto por três componentes básicos [2]:

- *software* (ferramenta) – controla o inventário tecnológico da empresa, revelando a repetição de problemas, os tempos médios de atendimento, as soluções para as áreas mais demandantes e a identificação da necessidade de treinamentos;
- equipe – atua em dois níveis: o campo e a retaguarda. A equipe de campo faz o atendimento aos problemas, solucionando-os quando possível ou acionando fornecedores internos e externos. A equipe de retaguarda é acionada quando um problema técnico supera a capacidade de resolução da equipe de campo. Os analistas de suporte de fornecedores funcionam como uma terceira camada e podem ser requisitados pela equipe de retaguarda, recebendo o problema identificado e mapeado;
- metodologia de serviço – contempla o posicionamento do sistema *Help-Desk* junto aos usuários, definindo quais são as estratégias de ação diante de determinado problema, apresentando os indicadores de desempenho e identificando novas oportunidades em um processo de melhoria contínua.

Um sistema *Help-Desk* é utilizado para melhorar o gerenciamento das soluções de atendimento. Através do *Help-Desk*, cria-se uma ampla base de dados para a empresa, a qual permite gerenciar os problemas, resolvê-los na sua raiz e diminuir custos operacionais. O *Help-Desk* pode centralizar uma diversidade de informações e áreas de atendimento, tornando-se assim um ponto chave na administração e na solução de problemas. Domínios atraentes para *Help-Desk* são relativos a: i) suporte à informática; ii) sac – serviço de atendimento ao consumidor (interno/externo); iii) controle de serviços/manutenção; e iv) centro de informações.

1.1 Motivação

A Prefeitura Municipal de Lavras, ciente da importância que a informática adquiriu nos últimos anos, tem se esforçado para modernizar-se, tendo iniciado o seu processo de informatização recentemente. Porém, como em todo processo de mudança, a prefeitura tem enfrentado problemas. A maioria de seus usuários não é habituada ao uso da informática. Isso ocasionou uma sobrecarga no Centro de Processamento de Dados (CPD) da Prefeitura, pois este passou a ser responsável também por todo setor de suporte. Essa sobrecarga fez com que a qualidade do atendimento baixasse e aumentou o custo operacional, pois áreas primordiais ficam paradas por falta de atendimento e manutenção preventiva. Além disso, houve um desgaste do CPD dentro da prefeitura. Entre os principais motivos que levam a Prefeitura Municipal de Lavras a estudar a implantação de um sistema *Help-Desk*, pode-se citar:

- chamados, pendências, reclamações e sugestões não são anotados e quando são não há controle sobre esses processos;
- o sistema de atendimento utilizado não atende todas as necessidades;
- não existe documentação das soluções de problemas resolvidos anteriormente;
- necessidade de diminuir os custos operacionais.

1.2 Objetivos

Este artigo tem como objetivos a modelagem do sistema utilizando a UML (*Unified Modeling Language*) [3], a modelagem de dados utilizando a notação UML [4] e o desenvolvimento do produto de *software Help-Desk* para a Prefeitura Municipal Lavras (PML). Esse *software* visa melhorar o suporte aos funcionários, que são os principais usuários da infra-estrutura tecnológica da prefeitura.

O *software Help-Desk* implementado dispõe-se não somente a facilitar a comunicação usuário-suporte, mas também a realizar o gerenciamento de inventário de *hardware* e de *software* da PML. Além disso, a ferramenta de *software* é capaz de gerar relatórios que auxiliam na tomada de decisões e na manutenção da infra-estrutura tecnológica. Desta forma, o CPD da PML passa a ter mais controle das atividades de suporte e presta um serviço de melhor qualidade aos seus usuários.

1.2.1 Metodologia do Desenvolvimento

O presente trabalho divide-se em duas fases. A primeira diz respeito à busca de informações e à análise das principais tecnologias e ferramentas de *software* empregadas em sistemas *Help-Desk*.

Durante essa etapa, o trabalho teve um cunho exploratório [5], tendo como objetivo proporcionar uma maior familiaridade com tema e o aprimoramento de idéias. Nesta fase, a pesquisa bibliográfica e documental foram as abordagens metodológicas utilizadas. Apesar de serem semelhantes no desenvolvimento, diferem na natureza das fontes [5]. A pesquisa bibliográfica utiliza fundamentalmente as contribuições dos diversos autores sobre determinados assuntos, sendo quase sempre constituída de material impresso localizado em bibliotecas. A pesquisa documental vale-se de matérias que ainda não tiveram tratamento analítico sendo muito mais diversificadas e dispersas, formada principalmente por jornais, revistas, *sites* da Internet, folhetos, manuais, entre outros.

A segunda etapa do trabalho consistiu na modelagem e no desenvolvimento do *Help-Desk* da PML. Durante essa fase do trabalho, foi utilizada a metodologia de pesquisa-ação [6]. A escolha dessa metodologia foi motivada pelo enfoque dado a este trabalho, onde foi desempenhado um papel ativo no equacionamento do problema, considerado como central na pesquisa, no acompanhamento e na avaliação das ações desencadeadas em função dos problemas, atuando para encontrar soluções.

1.2.2 Implementação

A ferramenta *Help-Desk* foi dividida em três partes:

- banco de dados: é responsável por armazenar todas as informações relevantes do sistema, como por exemplo: chamadas em aberto, soluções de problemas anteriores, quantidade de chamadas resolvidas em um mês, entre outras. Ele funciona como a memória do sistema. Essa parte da ferramenta foi implementada utilizando o sistema gerenciador de banco de dados (SGBD) MySql. A escolha do MySql se deve ao fato de ser confiável, robusto, multi-usuário, gratuito, portável, livre e de ter boa integração com aplicações *Web* [7];
- interface do usuário: é o meio pelo qual o funcionário da PML interage com o *Help-Desk*. É por essa interface que o usuário realiza as solicitações de suporte ao sistema e verifica o *status* de uma solicitação aberta. Essa parte da ferramenta foi implementada utilizando a tecnologia PHP (*Personal Home Page*), uma linguagem de script voltada para a construção de páginas *Web* dinâmicas o que facilita o acesso do usuário uma vez que a interface é visualizada em qualquer computador desde que tenha um *browser* e acesso a internet;
- interface do suporte: é por onde a equipe interage com o sistema, sendo a partir desta interface que os analistas têm acesso às chamadas realizadas pelos usuários, às soluções anteriores e aos dados estatísticos sobre usuários e equipamentos. É função da interface do suporte direcionar as novas solicitações ao analista pré-definido pelo administrador. Essa interface do sistema também foi desenvolvida utilizando a linguagem PHP.

No desenvolvimento do *Help-Desk*, foram utilizadas técnicas de engenharia de *software* objetivando conduzir o projeto dentro de um cronograma viável e adaptável. Para isso, foi utilizada a técnica de prototipação [8], pois melhor se enquadra ao projeto. Alguns motivos que levaram à sua escolha foram [8]:

- possibilita que o cliente possa acompanhar o desenvolvimento do *software*;
- mudanças durante a fase de implementação do projeto, não prejudicam significativamente sua construção;
- o *software* a ser desenvolvido é simples, não necessitando fazer uma análise de riscos como no modelo espiral.

O presente trabalho se encontra organizado em cinco seções. A seção 2 apresenta os conceitos envolvidos na utilização de ferramentas *Help-Desk*, bem como uma breve descrição e uma comparação de três ferramentas. A seção 3 discute as bases do sistema *Help-Desk*, tecnologias utilizadas no seu desenvolvimento e um exemplo de arquitetura. A seção 4 apresenta uma descrição sobre a funcionalidade do sistema implementado, bem com a sua modelagem e a sua arquitetura. A seção 5 apresenta as principais conclusões referentes ao trabalho, bem como algumas propostas de trabalhos futuros e contribuições.

2 Tecnologias Utilizadas

Esta seção define as bases tecnológicas e referenciais empregadas nos sistemas *Help-Desk*. Em termos mais amplos, pretende-se mostrar a integração destes sistemas ao contexto da área de Gestão do Conhecimento. Depois, é apresentado o conceito de Raciocínio Baseado em Regras, o primeiro modelo a ser utilizado na construção de sistemas *Help-Desk* Inteligentes. Em seguida, é mostrada a evolução desse conceito, o Raciocino Baseado em Casos (RBC), um paradigma para a construção desta classe de sistemas, principalmente aqueles baseados na Internet, chamados de *Web Help-Desk*. E, por último, é apresentado um modelo de arquitetura para sistemas *Web Help-Desk* utilizando Raciocínio Baseado em Casos e Regras.

2.1 Gestão do Conhecimento

Sistemas *Help-Desk*, juntamente com outras tecnologias computacionais crescentes, tais como *Data Warehouse*, *Intranet/Extranet*, *Groupware*, *Data Mining*, *Digital Whiteboards*, etc, não foram criados para existirem isoladamente, necessitam serem vistos como partes de um todo. Essas tecnologias necessitam de uma integração contextual para que os seus papéis possam ser devidamente valorizados. No caso de sistemas *Help-Desk*, a necessidade da sua integração à Gestão do Conhecimento decorre de sua própria natureza aglutinadora de informações. Um papel quase nunca explicitado.

Para entender o porquê dessa integração, é necessário que se defina Gestão do Conhecimento

e para tal é preciso antes compreender o que é conhecimento do ponto de vista computacional.

Conhecimento pode ser definido como sendo a combinação de experiência, valores, informação contextual, *insight* e intuições significativas que propiciam um ambiente e uma abordagem para novas informações e experiências [9].

Sendo assim, pode-se dizer que conhecimento é um sistema de dados e informações residente em memórias biológicas humanas. Nas organizações, o conhecimento pode tomar forma, não apenas de repositório, mas também de rotinas organizacionais, processos, práticas e normas. Este conhecimento pode ser classificado em dois tipos [9] e [10]:

- explícito: é o conhecimento que pode ser expresso em uma linguagem formal, em afirmações gramaticais, expressões matemáticas, especificações e manuais. Desta forma, este tipo de conhecimento formalizado pode ser apresentado e utilizado na resolução de problemas, definição de novos procedimentos e outras ações nas organizações.
- tácito: é o conhecimento que está relacionado a habilidades e competências pessoais e de difícil especificação. É o conhecimento pessoal incorporado à experiência individual e envolve fatores intangíveis. Consiste de modelos mentais que são usados para resolver problemas e que exercem influência sobre nossas ações e decisões.

A Gestão do Conhecimento pode ser vista como a formalização de conhecimentos para que estes se tornem acessíveis à organização, contribuindo para a melhoria do desempenho individual ou organizacional [11].

O sistema de Gestão do Conhecimento é uma ferramenta administrativa para representar, armazenar, compartilhar e distribuir conhecimentos com o objetivo de que a informação seja levada à pessoa certa no tempo certo. Para isso, utiliza-se de tecnologias de forma colaborativa e cooperativa [11].

A Gestão do Conhecimento significa adotar uma diversidade de fontes do conhecimento com a ajuda de um banco de dados corporativo e ambientes baseados em troca de informações via intranet e internet, onde funcionários e demais colaboradores poderão utilizar os recursos para descrever os modelos mentais que utilizam na resolução de problemas [11].

Sendo assim, pode-se definir Gestão do Conhecimento como um processo sistemático de: buscar, selecionar, organizar, filtrar e exibir informações, tendo em vista melhorar a compreensão de um usuário qualquer, em uma área específica de interesse [9].

A Gestão do Conhecimento não é, porém, só um problema de tecnologia; ela é igualmente um problema de gerenciamento. É importante salientar que:

- Gestão do Conhecimento não é Engenharia do Conhecimento;
- Gestão do Conhecimento diz respeito a processos, não somente a redes digitais;
- Gestão do Conhecimento não se refere à construção de Intranets inteligentes;
- Gestão do Conhecimento não se refere a investimentos pontuais;
- Gestão do Conhecimento não é simplesmente a integração de conglomerados empresariais;
- Gestão do Conhecimento não significa apenas a captura.

Antes de correlacionar *Help-Desk* como uma tecnologia de Gestão do Conhecimento, é importante enfatizar que sistemas facilitadores de informação podem ser modelados com o emprego de metodologias e ferramentas as mais variadas. Têm-se interesse, particularmente, os sistemas construídos com base no conhecimento sobre Problemas (a origem das Perguntas de um usuário) e sobre as suas Soluções (exibíveis em forma de Respostas, pelo sistema *Help-Desk*).

Um exemplo desse tipo de metodologia, conhecida como “cinco passos”, consiste em dividir a resolução de problemas nos seguintes estágios [11]:

- identificação do problema: o trabalhador, ou grupo de trabalhadores, define exatamente qual foi o problema ocorrido, procurando fornecer dados suficientes para uma análise posterior. Neste passo, desencadeia-se um processo de comunicação aos demais funcionários envolvidos que irá remetê-los ao segundo passo, ou seja, o estudo das prováveis causas;
- estudo de prováveis causas: conta-se com a participação de todos os trabalhadores envolvidos para uma avaliação das prováveis causas para o problema. Este passo é extremamente rico, pois possibilita a manifestação de todos os colaboradores, envolvidos diretamente ou não, para encontrar uma solução. Esta solução é definida para compor o terceiro;
- determinação da causa raiz do problema: é escolhida uma das causas, ou conjunto de causas, definida no segundo passo e, a partir desta definição, parte-se para o passo seguinte que é apresentação de ações corretivas;
- ações corretivas: as ações corretivas são alterações formalmente implementadas em processos, produtos, materiais, máquinas ou outros componentes envolvidos com o problema para eliminar a causa raiz específica. O

importante é a documentação dos procedimentos adotados e as instruções de trabalho, atualizando-as de maneira apropriada;

- eficácia da solução: os resultados alcançados são verificados. Coleta de dados, auditoria, estudos periódicos ou monitoramento são algumas das ações necessárias para a validação da solução raiz para o problema. Se as evidências indicarem que o problema foi solucionado, a empresa poderá armazenar todos estes dados com o status de problema solucionado. Caso contrário, o problema fica em aberto aguardando outras soluções como problema não solucionado. O importante é que existe uma documentação das ações desencadeadas e do insucesso alcançado. Isto irá evitar que outros trabalhadores, ou grupos de trabalhadores, invistam em possíveis soluções experimentadas anteriormente.

Pode-se então dizer que sistemas *Help-Desk* servem como um ponto único de intervenção para a solução de problemas enfrentados por usuários e onde a fonte das soluções computacionais propostas está na representação computacional de conhecimento [9].

Esta representação computacional sobre Problemas e Soluções dá origem a um importante repositório denominado base de conhecimento.

Uma base de conhecimento pode ser definida como sendo uma estrutura organizada de informação que facilita o armazenamento da inteligência do sistema com a finalidade de ser resgatada ou recuperada em apoio a uma demanda feita ao sistema *Help-Desk* [9].

Sendo assim, atender ao cliente/usuário consiste desta capacidade do sistema de oferecer uma determinada Resposta como resultado imediato de uma determinada Pergunta emitida por este usuário.

A base de conhecimento possui o papel de armazenar problemas soluções candidatos. Algum destes problemas e soluções armazenados haverão de coincidir com o problema que desafia o cliente/usuário, em um dado momento. É esta base de conhecimento que viabiliza, por exemplo, o apoio ao consumidor como um valioso recurso *on-line* capaz de oferecer respostas a problemas de forma rápida e correta. A base de conhecimento evita, assim, que se tenha de "reinventar a roda" a cada pedido de ajuda as empresas/organizações, por parte de clientes de produtos e/ou serviços.

O papel do sistema *Help-Desk* como um processo automático da Gestão do Conhecimento, no interior de uma organização, fica deste modo evidenciado. Pois, ambos [9]:

- lidam com processos envolvendo conhecimento;
- requerem a criação de bases de conhecimento nas organizações;
- dedicam-se à inteligência e à aprendizagem das organizações;
- estão totalmente a serviço da moderna economia baseada em conhecimento.

2.2 Raciocínio Baseado Regras

As funções de um sistema *Help-Desk*, envolvem tarefas de classificação, que se caracterizam pela necessidade de enquadrar um certo objeto, situação ou evento em uma dada categoria predeterminada.

Ao realizar classificações, os sistemas *Help-Desk* respondem indagações que podem vir a ser colocadas pelos usuários destes sistemas. Nos primeiros sistemas *Help-Desk*, todas as indagações eram respondidas pela equipe, cabendo as ferramentas de *software* apenas armazenar dados relativos a inventário, tempo de atendimento e geração de dados estatísticos. Nos sistemas atuais, a própria ferramenta de *software* é capaz de responder dúvidas dos usuários. Os sistemas *Help-Desk* dotados dessa característica são chamados de *Help-Desk* inteligente [9].

A construção de um sistema *Help-Desk* inteligente consiste em captar o conhecimento de um analista, representar este conhecimento em uma base e transmiti-lo ao usuário, permitindo-lhe obter respostas a perguntas relacionadas à base de conhecimento do sistema [12].

O uso de regras de produção é uma das maneiras mais utilizada para representação do conhecimento. Em sistemas baseados em regras, o conhecimento é representado por meio de pares condição-ação, onde as regras possuem duas partes: uma antecedente ("*IF*") e outra conseqüente ("*THEN*"), a esse tipo de regra dá-se o nome de *IF-THEN* [13].

A capacidade para definir as regras no formato *IF-THEN* tem diversas vantagens [13]:

- as regras são entendidas pelos programadores e pelos peritos de uma forma idêntica;
- as regras podem conter pequenos "pedaços" de conhecimento, que coletivamente podem modelar um problema bastante complexo;
- as regras são independentes entre si;
- as regras podem ser colocadas em qualquer ordem dentro do programa;
- as regras fazem parte da vida cotidiana e as pessoas estão familiarizadas com elas;
- a estrutura de controle assemelha-se a algumas estratégias humanas de resolução de problemas;
- a estrutura de controle é relativamente simples, podendo ser entendida pela maioria das pessoas.

Nos sistemas baseados em regras, o conhecimento é representado por fatos, relações entre fatos e regras para a manipulação desses fatos. Esta aparente simplicidade é dificultada pelas situações em que mais de uma regra podem ser aplicada ou quando a partir da aplicação de uma regra, outras também passam a ser aplicáveis. Deste modo, um programa baseado em regras necessita de uma estrutura de controle que permita determinar qual a próxima regra a ser aplicada e como encadear as regras. Considere as seguintes regras simples:

Regra 1: *IF* A *THEN* B
Regra 2: *IF* B *THEN* C
Regra 3: *IF* C *THEN* D

Ou seja, A é uma função que retorna o valor verdadeiro quando a função B também retornar verdadeiro. O resultado da função B é verdade quando a função C retornar o valor verdadeiro e esta só é verdade quando a função D também o for.

Sendo assim, pode-se inferir logicamente pela aplicação das três regras que, se A é verdadeiro então D também é verdadeiro. Isto pode ser representado esquematicamente como na Figura 1. Um conjunto de regras mais complexo, como:

Regra 1: *IF* A *THEN* B & C
Regra 2: *IF* B *THEN* D
Regra 3: *IF* C *THEN* E
Regra 4: *IF* D *THEN* G

pode ser representado como na Figura 2.

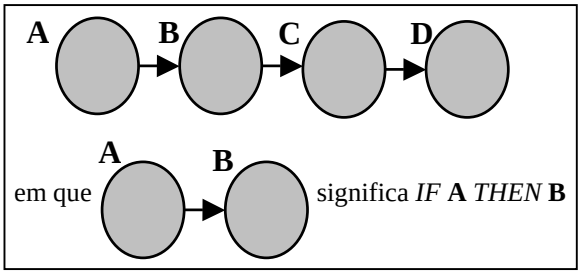

Figura 1 – Representação Gráfica de Regras [14]

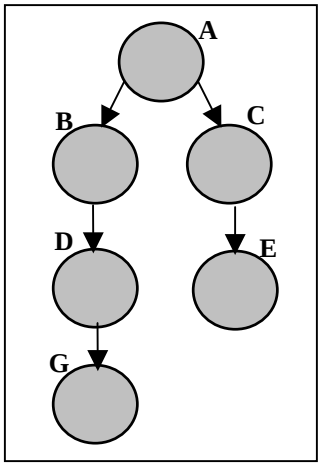

Figura 2 – Árvore de Regras [14]

O problema, nesta situação, é a ordem pela qual as regras serão aplicadas, pois, a menos que o computador possua processadores paralelos, é necessário definir qual a regra deve ser acionada primeiro. Se A for verdadeira, é melhor tornar B verdadeira antes de C ou o contrário? De fato, é tarefa do programador decidir, mas uma estrutura freqüentemente usada e denominada *depth-first search* faz com que inferências sejam feitas pela ordem indicada pelos números na Figura 3 [15].

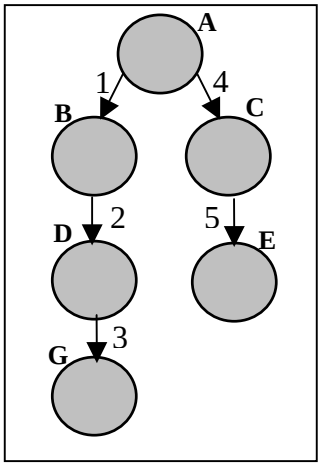

Figura 3 – *Depth-First Foward Chaining* [14]

Deste modo, o sistema infere que G é verdadeiro antes de inferir que E é verdadeiro. Na realidade, se G for a solução, pode não ser necessário inferir que E é também uma solução potencial. Esta estrutura de controle é mais corretamente denominada *depth-first forward chaining*. *Forward* significa que novos fatos são inferidos de fatos conhecidos. É também possível usar o mesmo conjunto de regras pela ordem inversa – *backward chaining* – para descobrir quais as condições que necessitam serem verdadeiras para que uma condição seja também verdadeira.

A estrutura *backward chaining* é freqüentemente usada para permitir que se teste uma hipótese. Isto "imita" as estratégias de resolução de problemas por parte dos humanos [15].

Os sistemas baseados em regras têm limitações significativas. Na maioria das situações, é extremamente difícil obter um conjunto correto de regras, porém o problema central aos sistemas baseados em regras é o próprio conhecimento. A abordagem baseada em regras assume que existe um corpo de conhecimento que a maioria dos especialistas na área usa e aprova. No entanto, em muitas áreas cotidianas não existem modelos casuais subjacentes nem princípios gerais aceitos pela maioria dos especialistas, para se produzir um modelo. Deste modo, dada a inexistência de um modelo explícito ou a extrema dificuldade em apreender o modelo, torna-se não trivial o

desenvolvimento de sistemas baseados em regras surgindo então o raciocínio baseado em casos [14].

2.3 Raciocínio Baseado Em Casos

Para a construção de um sistema baseado em regras, tem-se de conhecer previamente como resolver o problema, para então elaborar um conjunto de regras que resolva o problema cada vez que este ocorrer. Mas, para que resolver um problema cada vez que ele ocorre, se alguém o resolveu anteriormente? Não seria mais simples recordar apenas a solução? Além disso, por vezes, sabe-se apenas como se resolveu o problema em determinada circunstância, mas é difícil generalizar para quaisquer situações.

Pensando assim, foi desenvolvida uma outra metodologia de representação do conhecimento chamada de Raciocínio Baseado em Casos (RBC).

Os sistemas *Help-Desk* centrados no RBC oferecem respostas às perguntas procurando em uma base de casos (devidamente indexada), casos passados que se aplicam ao problema atual. Uma indexação coerente e o modo como os casos são representados facilitam sua recuperação correta.

É possível simplificar o processo mental de um sistema RBC como sendo cíclico e composto por quatro "R" [16] (Figura 4): i) recuperar o(s) caso(s) mais similar(es); ii) reusar o(s) caso(s) para resolver o problema; iii) revisar a solução proposta, se necessário; e iv) reter a nova solução incluindo-a na base de casos.

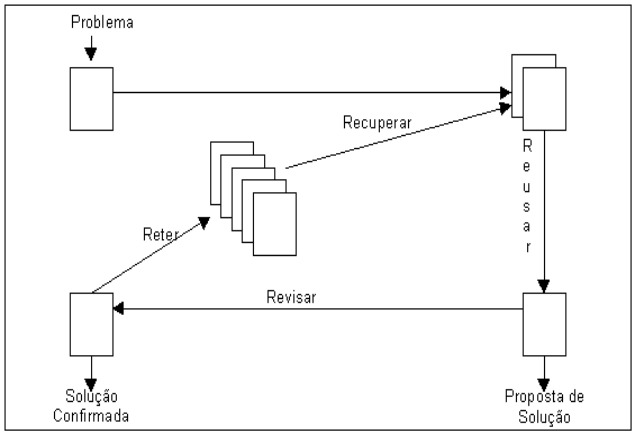

Figura 4 – O ciclo do RBC [10]

As aplicações RCB devem seguir os seguintes passos [9]: i) representação do caso; ii) entrada dos casos; iii) indexação dos casos; iv) recuperação dos casos; v) avaliação e comparação; e vi) adaptação.

A qualidade das soluções depende fundamentalmente de [9]: i) experiência; ii) habilidade de comparar e entender um novo caso em relação a um existente; iii) capacidade de adaptação; e iv) capacidade de avaliação.

A principal parte do conhecimento nos sistemas RBC é representada através de seus casos. Um caso pode ser entendido como a abstração de uma experiência descrita em termos de seu conteúdo e contexto, podendo assumir diferentes formas de representação. Todo caso é composto por um problema, que descreve o estado do mundo real onde o caso ocorre, e uma solução, que contém o estado das soluções derivadas para o problema. É importante salientar que um caso não é uma regra. A representação dos casos é uma tarefa complexa e importante para o sucesso do sistema RBC [9].

A representação dos problemas deve conter as metas a serem alcançadas na resolução do problema, as restrições e as características da situação e as relações entre suas partes. Quando montado um sistema, cada diferente solução ou interpretação do problema é um novo caso.

Na representação dos casos, cada característica tem uma certa importância. Baseado nisso, é feita a indexação dos casos, ou seja, associam-se rótulos de forma a caracterizá-los, para depois recuperá-los na base de casos. A escolha dos índices é de vital importância à recuperação dos casos, pois orienta a avaliação da similaridade. A indexação determina o que comparar entre os casos para determinar a sua similaridade. Assim, RBC, além de usar índices com os objetivos de facilidade e rapidez na recuperação, também os usa para realizar eficientemente a atribuição de similaridade entre os casos. Bons índices são abstratos o suficiente para fornecerem cobertura, mas concreto suficiente para serem reconhecíveis [9].

Uma das características importante dos sistemas de RBC é a capacidade de identificar, entre os casos da base de conhecimento, quais são os mais úteis para resolver o problema do usuário. Essa identificação dos casos ocorre através de procedimentos de comparação e medição de similaridades [17]. A determinação da medida de similaridade é um importante componente para determinar a utilidade do caso. Deve-se considerar também que o grau de utilidade de um caso depende dos propósitos a que ele se destina e quais dos seus aspectos foram relevantes no passado. Estas considerações habilitam os procedimentos de comparação a determinar em que dimensão é importante para um caso focá-lo no julgamento da similaridade [17].

Considerando que a definição dos índices retrata todos os aspectos a serem considerados na recuperação, os algoritmos de comparação podem usá-los para se orientar na busca, determinando quais características devem ser focadas no

julgamento da similaridade. Qualquer caso pode ser indexado de várias formas, entretanto o importante é que o algoritmo de comparação esteja apto a distinguir entre os aspectos relevantes em cada caso específico [9].

No processo de recuperação, não se pode falar apenas em comparação e medição da similaridade, precisa-se falar também sobre as estruturas que serão usadas pelos algoritmos para dirigir a busca. Na recuperação, são usadas duas técnicas de busca que são [15] e [18]:

- o vizinho-mais-próximo: os aspectos de definição e identificação dos índices é fator fundamental para uma recuperação de sucesso. Garantidos estes aspectos, a técnica de busca indica quais são os casos que apresentam um problema semelhante. O próximo passo é a comparação e a valorização da similaridade entre o contexto dos casos passados e o caso atual;
- indutiva: constroem-se árvores de decisão baseadas em dados de problemas passados. Em sistemas RBC, a base de casos é analisada por um algoritmo de indução, que cria a árvore de decisão classificando ou indexando os casos. Para que o algoritmo construa a árvore de decisão a partir dos casos da base de casos, é necessário passar-lhe os atributos que melhor identificam os casos. Encontrado o primeiro atributo, o algoritmo monta o primeiro nó da árvore. O passo seguinte é encontrar dois novos atributos que formem os próximos nós e assim por diante. Montada a árvore a partir da base de casos, o próximo passo é percorrer a árvore com o caso em questão. Quando chegar no último nó da árvore, têm-se os casos mais similares.

Ambas as técnicas são boas, porém cada uma delas tem características que se enquadram melhor para determinados tipos de problemas. A técnica do vizinho-mais-próximo é mais indicada para problemas com bases de casos pequenas e com poucos atributos indexados, devido ao volume de cálculos necessários para determinar cada um dos atributos indexados e cada um dos casos. A técnica indutiva, por sua vez, é bem mais rápida, somente ficando lenta para bases de casos muito grandes. O principal problema desta técnica é que casos inéditos não recuperarão nada [18].

As soluções passadas são adaptadas para solucionar novos problemas, pelo fato de nenhum dos problemas passados ser exatamente igual a um problema atual. A adaptação pode ser uma simples substituição de um atributo da solução por outro ou uma complexa e total modificação na estrutura da solução.

A adaptação pode ser feita de várias formas [18] pela inclusão de um novo comportamento à solução recuperada, pela eliminação de um comportamento da solução recuperada e pela substituição de parte de um comportamento. Podem ocorrer situações onde há a presença de mais de uma das formas.

Os métodos, para proceder à adaptação, podem ser classificados da seguinte forma [9]: i) substituição; ii) transformação; e iii) outros métodos.

Apesar da adaptação poder ser usada de várias formas e em várias situações, ela não é essencial. Muitos dos sistemas comerciais de RBC não implementam a adaptação. Eles simplesmente recuperam o caso mais similar e disponibilizam a solução para o usuário, deixando-o livre para proceder à adaptação. Isto acontece devido a grande complexidade da adaptação. Na maioria das vezes, para implementar a adaptação é necessário representar um volume muito grande de conhecimento.

A aprendizagem em um sistema de RBC acontece principalmente pelo acúmulo de novas experiências em sua memória e pela correta indexação dos problemas. Um sistema de RBC se torna eficiente quando estiver preparado para, a partir das experiências passadas e da correta indexação dos problemas, aprender [9].

A implementação da aprendizagem em um sistema RBC está baseada no fato de que, a partir das experiências passadas, o sistema está apto a analisar os efeitos da sua solução e a armazenar informações sobre o que deu certo, o que não deu certo e porque. Com estes procedimentos, o programa poderá montar melhores respostas. Isto torna o sistema RBC muito mais confiável, à medida que antecipa erros cometidos no passado.

Quanto mais casos houver na memória do RBC, maior o número de casos que, agrupados por famílias (conjuntos de atributos que definem a similaridade), poderão contribuir para solução de um novo caso. Casos que tiveram sucesso em recuperações anteriores e casos que não tiveram sucesso dão maior amplitude de cobertura ao problema. Novos índices dão a sintonia necessária para recuperar os casos em situações diversas mais apropriadas [9].

A maioria dos sistemas *Help-Desk*, atualmente em funcionamento, têm sido projetados com base nestes casos computacionais. Pesquisas recentes dão conta de que 58,5% de todas as aplicações da tecnologia de RBC estão concentradas no desenvolvimento de sistemas *Help-Desk* ou de apoio a clientes e usuários [18].

2.4 Arquitetura *Help-Desk* Baseada Em Casos e Regras

O uso de ferramentas *Help-Desk* inteligentes permite construir sistemas que, além das funções

usuais de controle e supervisão do fluxo de atendimento aos chamados, facilita a resolução técnica do problema.

Grande parte dos sistemas *Help-Desk*, que oferecem ferramentas inteligentes, utiliza o paradigma baseado em regras ou o baseado em casos para implementá-lo. Essa abordagem, onde apenas uma metodologia é utilizada, não é suficientemente boa, pois ambas possuem deficiências [9].

Mostra-se, como alternativa, uma arquitetura implementada através do emprego do Raciocínio Baseado em Casos acoplado ao uso de Regras. A Figura 5 apresenta um exemplo de arquitetura de um sistema *Help-Desk* inteligente [19].

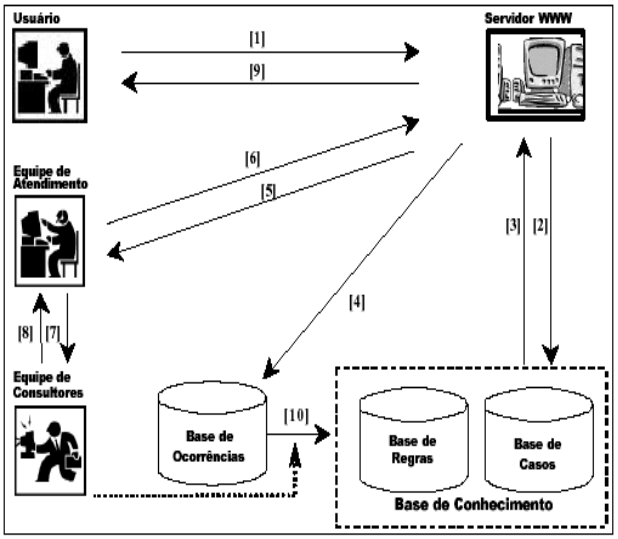

Figura 5 – Arquitetura *Web Help-Desk* Utilizando RCB e Regras [19]

Essa arquitetura é particularmente interessante, pois explora a modelagem da pergunta-resposta com apoio tanto de casos computacionais quanto de regras computacionais e ainda com o suporte da Internet. A arquitetura compreende uma base de conhecimento, uma base de ocorrências, um servidor *Web*, uma equipe de atendimento e uma equipe de consultores no domínio de aplicação. A idéia central da arquitetura é fazer com que somente problemas de extrema complexidade no domínio cheguem às mãos dos consultores.

A base de conhecimento é composta por uma base de regras e uma base de casos, sendo essa a principal diferença dessa arquitetura híbrida para as arquiteturas convencionais. Junto com a base de conhecimento, a base de ocorrências forma a memória do sistema. A base de ocorrências é responsável por registrar todas as solicitações que foram feitas, porém não foram resolvidas de forma automática. É a partir da base de ocorrências que a base de conhecimento é atualizada, seja pela inserção de novos casos, ou pela criação de novas regras.

Nessa arquitetura, o servidor *Web* é responsável por ser a interface entre o usuário e o suporte. O sistema funciona da seguinte forma, ao necessitar de suporte, o usuário se conecta a Internet e acessa o suporte (1) através do servidor *Web* da empresa. Em uma primeira instância, o sistema tentará responder diretamente à indagação do usuário através de busca à base de regras (2), que soluciona problemas que ocorrem com maior freqüência (3, 9). Quando o sistema não consegue uma resposta direta para o problema do usuário, isto significa que, em uma primeira instância, não existe qualquer regra na base de regras capaz de responder exatamente à indagação feita. O passo seguinte será tentar obter da base de casos um caso que seja semelhante ao problema trazido e cuja solução possa ser utilizada para a situação desse usuário. Assim, o sistema realiza uma busca na base de casos e recupera aqueles casos que mais se assemelhem ao problema trazido pelo usuário (3). Os casos resgatados são apresentados ao usuário, dispostos em ordem de semelhança e segundo algum critério de similaridade (9). Se, porém, o problema trazido também não estiver previsto na base de casos, este problema é inserido da base de ocorrências (4) e repassado para a equipe de atendimento (5). Ao resolver rapidamente o problema, uma resposta é enviada ao usuário (6, 9). Só então, quando a equipe de atendimento não puder resolver o problema, ela poderá acionar a equipe de consultores (7,8). Cabe ainda à equipe de consultores a responsabilidade de inclusão de novos casos a povoarem a base de casos (10) de modo a permitir a aprendizagem do sistema. Estes casos podem ser extraídos a partir de uma base de ocorrência (casos reais) ou hipoteticamente criados (casos abstratos).

3 Desenvolvimento do Sistema *Help-Desk*

Esta seção descreve a funcionalidade e a arquitetura do sistema *Help-Desk* desenvolvido. Além disso, são apresentadas a modelagem de dados e a modelagem do sistema.

3.1 Descrição do Sistema *Help-Desk*

Os objetivos do sistema *Help-Desk* desenvolvido é gerenciar e controlar a qualidade dos serviços oferecidos pelo setor de suporte em informática da PML. A base desse controle consiste na gestão dos equipamentos que compõe a infra-estrutura tecnológica da PML e no rastreamento dos chamados abertos pelos usuários.

Com o intuito de oferecer informações gerenciais, o *software* armazena dados cadastrais dos usuários, o registro dos equipamentos de informática e as suas características. Caso o equipamento esteja na garantia, ou coberto por um contrato de assistência técnica, terá associado o código da empresa responsável pela sua manutenção, visto que o sistema armazena informações relativas às empresas de assistência com as quais a PML trabalha.

Se o equipamento for um computador ou uma antena de internet a rádio, alguns dados adicionais sobre sua configuração são registrados. Além disso, o sistema armazena informações referentes aos produtos de *software* que a PML utiliza.

Quando um usuário necessita de assistência para a utilização de algum equipamento, ou mesmo de algum serviço do setor de informática, este deve acessar o *site* do *Help-Desk* e abrir um chamado, informando qual o tipo e a categoria do chamado.

Cada chamado de suporte está associado obrigatoriamente ao usuário que originou o chamado e será atendido de acordo com a prioridade associada ao setor que ele pertence. Dependendo da categoria do chamado, o usuário deverá informar qual é o equipamento ou *software* que apresenta problema. Cada chamado aberto é automaticamente associado a um analista que fica responsável pela resolução do problema.

Durante o atendimento de um chamado, podem ocorrer vários eventos tais como: i) equipamento pode ser enviado ou recebido da assistência técnica; ii) chamado pode ser transferido de um analista para outro; iii) atendimento pode ser temporariamente suspenso devido à falta de peças de reposição. O sistema mantém registro dos eventos ocorridos para cada chamado.

Através desses dados, é possível um maior controle sobre o serviço de suporte da PML, uma vez que o sistema fornece informações mais precisas e confiáveis sobre os chamados.

O sistema possui a possibilidade de geração de relatórios gerenciais que auxiliam a avaliação do desempenho dos técnicos, a tomada de decisões e a elaboração de material para treinamento dos usuários e do quadro técnico.

Além disso, as estatísticas que poderão ser extraídas do sistema tornarão as estimativas dos prazos mais precisas para a solução dos problemas que surgem com maior freqüência.

3.2 Arquitetura

Um dos requisitos fundamentais do sistema é ser descentralizado, ou seja, os usuários deveriam ser capazes de abrir chamados usando suas próprias estações de trabalho, sem a necessidade de ligar para o setor de suporte da PML.

Para obedecer a esse requisito, a arquitetura escolhida foi a *thin client* baseada em *Web*, pois as aplicações são executadas integralmente no servidor e a única função da estação cliente é a exibição da interface com o usuário [20].

Um sistema *thin client* baseado em *Web* consiste de um servidor *Web* central que hospeda o aplicativo, processa as solicitações e armazena informações no banco de dados (caso seja necessário o banco de dados da aplicação pode ser hospedado em um servidor separado) [20].

Os usuários finais do sistema podem acessar o aplicativo a partir de qualquer *desktop* que possua um navegador *Web* e acesso ao servidor onde a aplicação está hospedada (Figura 6).

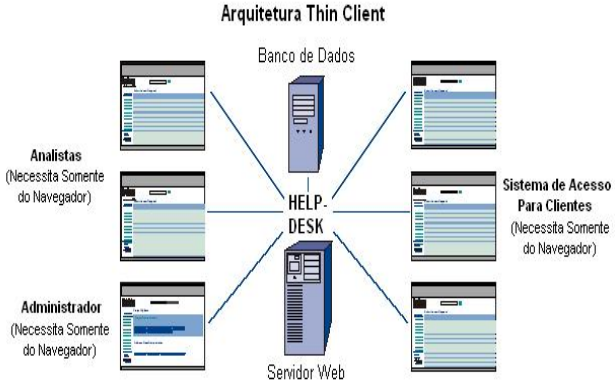

Figura 6 – Arquitetura *Thin Client* Baseada em *Web* [13]

A linguagem escolhida para implementação foi o PHP, por ser executado no servidor sendo enviado para o cliente apenas HTML puro, ou seja, a linguagem se enquadra perfeitamente na definição da arquitetura escolhida. O sistema gerenciador de banco de dados utilizado pelo sistema é o MySql. A sua escolha foi pelo fato de ter bom desempenho em aplicações *Web* e o PHP possuir suporte nativo a ele. Os requisitos para a utilização do *software* são acesso à Internet, browser *Web* e sistema operacional Microsoft Windows ou Linux.

3.3 Modelagem de Dados

O sistema mantém um banco de dados contendo informações, as quais a empresa necessita sobre os recursos de informática, usuários desses recursos e registro de ocorrência de chamados de suporte, suas causas e o tempo necessário para a solução dos problemas.

A abordagem escolhida para a representação do banco de dados é o paradigma relacional, para tal, a modelagem de dados será utilizada a UML [4].

3.4 Modelo de Dados

O poder de modelagem da linguagem UML [21] não se limita apenas ao desenvolvimento de *software* orientado a objetos. Cada vez mais, a UML está sendo aplicada em outras áreas de desenvolvimento de *software*. Um exemplo é o Diagrama de Modelagem de Dados, que é uma extensão da linguagem UML para que esta possa suportar a modelagem de bancos de dados relacionais. Essa ampliação inclui extensões especiais para tabelas, *schema* de banco de dados, chaves de tabelas, *triggers* e *constraints* [4].

O Diagrama de Modelagem de Dados UML pode ser usado para descrever o desenvolvimento completo de um banco de dados relacional ou objeto relacional, desde as exigências empresariais até o modelo de dados físico.

A Figura 7 apresenta o Diagrama de Modelagem de Dados UML para o banco de dados do sistema *Help-Desk* da PML.

3.5 Modelagem do Produto de Software

Nesta seção, são mostrados os aspectos de modelagem do sistema *Help-Desk*. Para isso, é utilizado um subconjunto dos artefatos de *software* (diagramas e conceitos) presentes na UWE [22].

A finalidade da modelagem é apresentar uma visão lógica do sistema, mostrando as suas características e a sua funcionalidade, facilitando a sua compreensão e o seu entendimento. Os artefatos de *software* utilizados na modelagem são: modelo de casos de uso, modelo conceitual e modelo navegacional.

3.5.1 Diagrama de Casos de Uso

O diagrama de casos de uso é utilizado para descrever a funcionalidade da aplicação e a sua interação com os usuários.

Os casos de uso são apresentados utilizando o Diagrama de Casos de Uso proposto pela UML, sendo escrito em termos de atores, casos de uso e o sistema a ser modelado. Os atores representam o papel de uma entidade externa como por exemplo um usuário, um *hardware* ou até mesmo um outro sistema. Os atores se comunicam com a aplicação através dos casos de uso, que representam uma seqüência de ações a serem executadas.

O Diagrama de Casos de Uso é central, pois seu conteúdo é base do desenvolvimento de outros diagramas e da implementação da aplicação (Figura 8 e Figura 9).

Para a utilização do sistema, há três tipos de atores:

- funcionário administrativo: realiza somente as funções de inclusão e consulta de chamado, além da operação de alteração de seus dados pessoais;
- analista: realiza todas as operações exceto as de gerenciamento de analista, de geração de relatórios gerenciais e de subfluxo exclusão nas operações de gerenciamento;
- administrado: realiza todas as operações, uma vez que cabe a esse ator a função de gerenciar o sistema.

3.5.2 Modelo Conceitual

O objetivo do *design* conceitual é construir um modelo do domínio da aplicação levando em conta as exigências observadas nos casos de uso. Nesse modelo, as classes e os objetos do sistema e a relações entre eles são modelados utilizando-se técnicas tradicionais de orientação a objeto. O modelo conceitual é representado por um Diagrama de Classes UML.

O sistema *Help-Desk* desenvolvido possui dezenove classes. Os atributos das classes foram omitidos, uma vez que, estes são iguais aos das tabelas mostradas no modelo de dados. A Figura 10 apresenta o modelo conceitual.

3.5.3 Modelo de Navegação

A modelagem de navegação de aplicações *Web* compreende a construção de dois modelos, o modelo do espaço de navegação e o modelo de estrutura de navegação [22].

O modelo do espaço de navegação é baseado no modelo conceitual e nos requisitos definidos nos casos de uso. A função desta modelagem é especificar quais classes do modelo conceitual serão visíveis ao usuário e quais serão os caminhos para se chegar a essas classes [22].

A Figura 11 apresenta o modelo do espaço de navegação para o sistema *Help-Desk* da PML. Os principais elementos deste modelo são as classes de navegação derivadas do modelo conceitual cujo estereotipo é *«navegation class»* e a associação entre as classes. Nesse modelo, as classes de navegação representam as páginas e as associações representam os *links*. Somente as classes do modelo conceitual que são pertinentes para navegação foram incluídas.

O modelo de estrutura de navegação é construído a partir do refinamento do espaço de navegação. Neste modelo, são adicionados elementos de acesso como índices, excursões, consultas e *menus* [22].

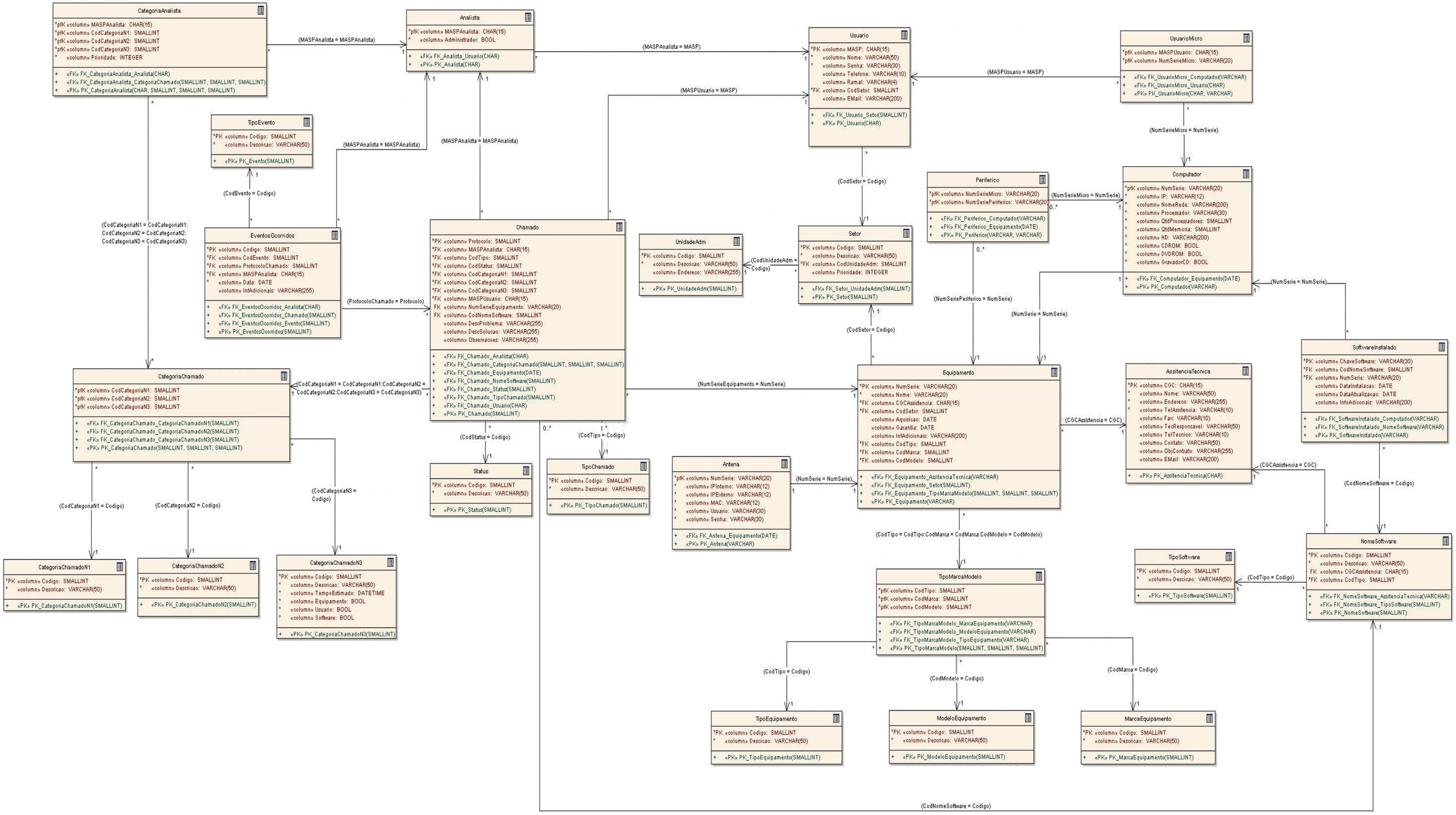

Figura 7 – Modelagem de Dados

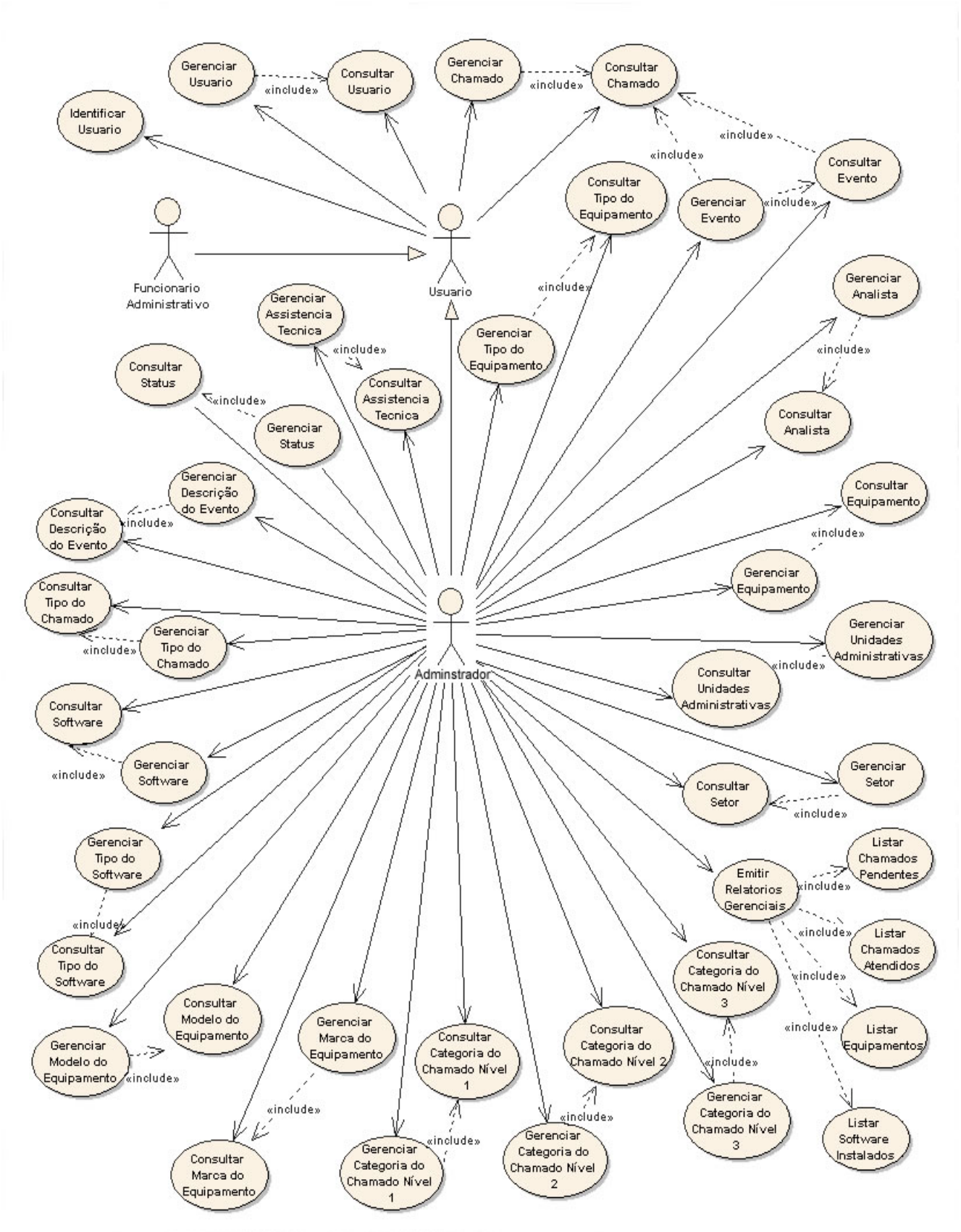

Figura 8 – Diagrama Casos de Uso (1)

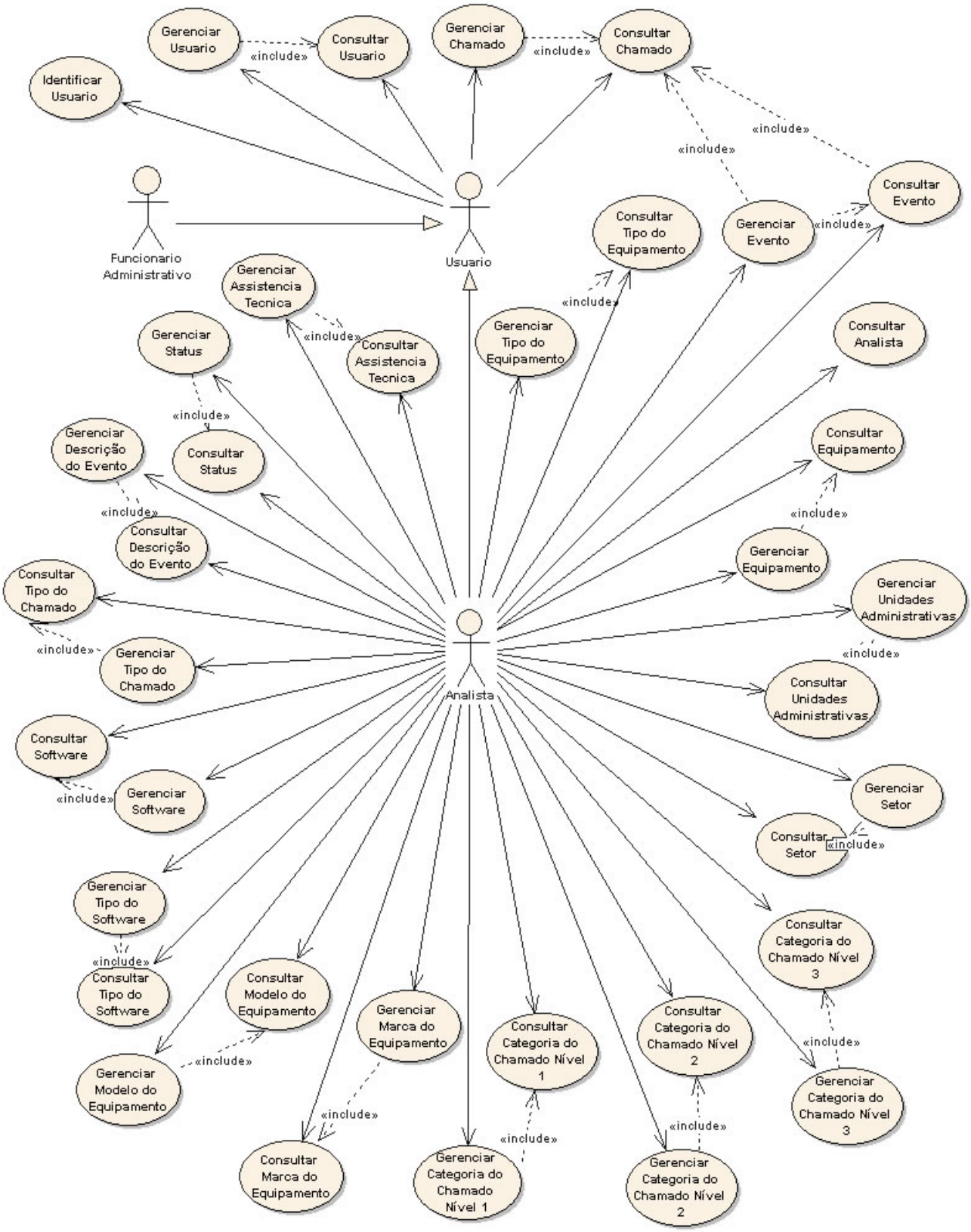

Figura 9 – Diagrama Casos de Uso (2)

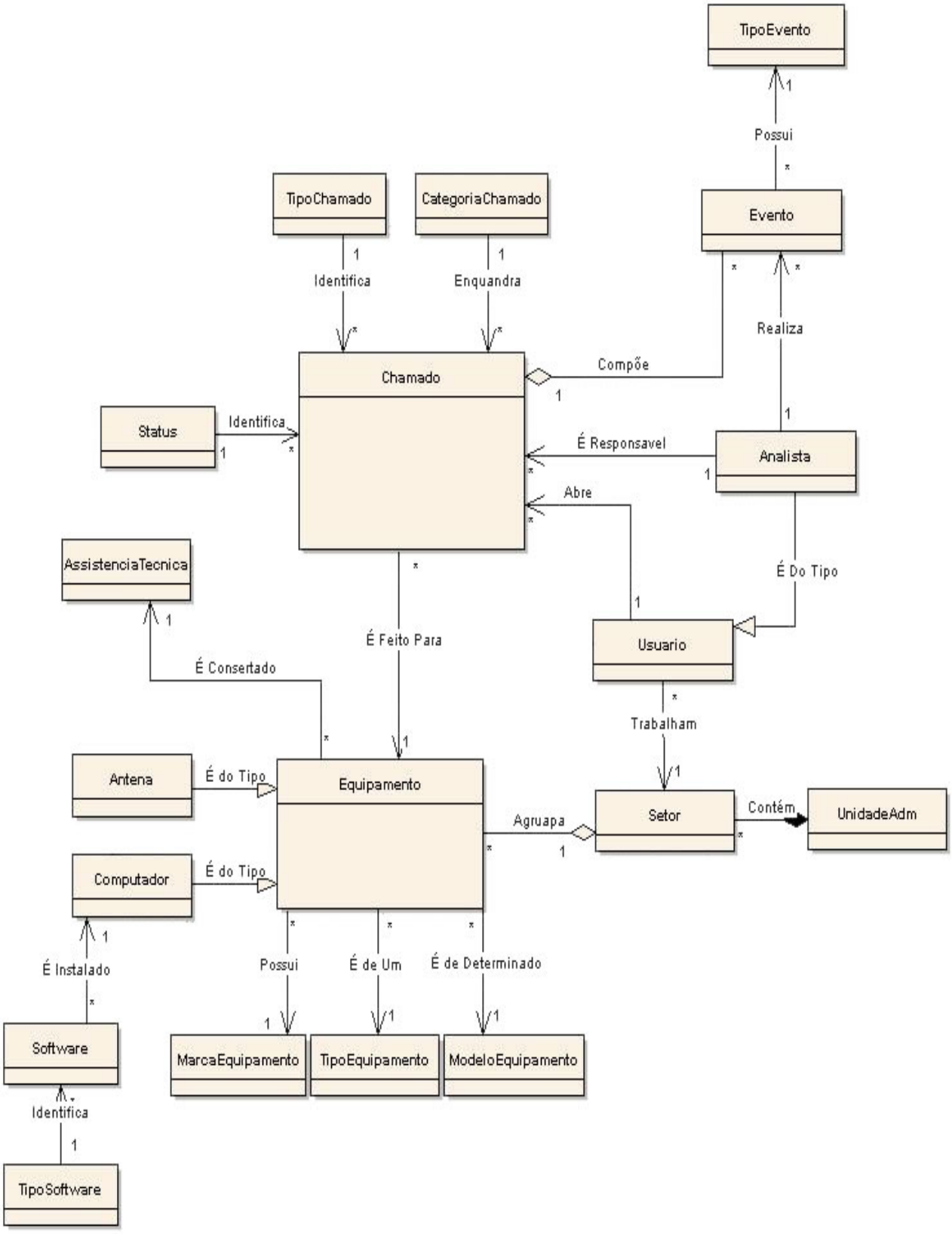

Figura 10 – Modelo Conceitual

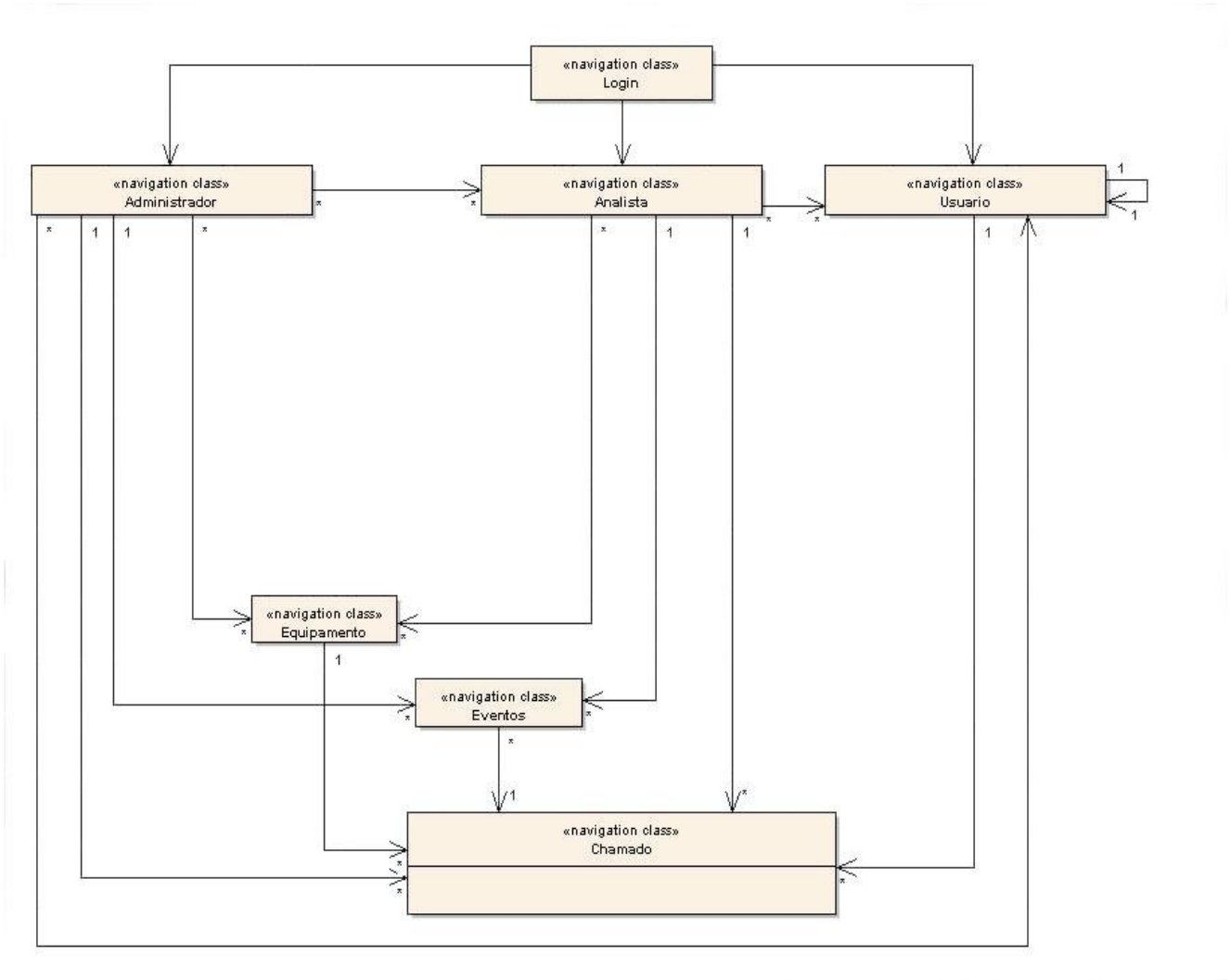

Figura 11 – Modelo do Espaço de Navegação

3.6 Funcionamento do Sistema

Nesta seção, são apresentadas algumas telas do sistema *Help-Desk*, explicando a sua funcionalidade e as suas características.

A primeira página exibida ao usuário é a tela de autenticação, isto ocorre para garantir que somente pessoas autorizadas acessem o *Help-Desk* e para identificar os três tipos de usuário: funcionário administrativo, analista e administrador, sendo os dois últimos da equipe de suporte (Figura 12).

Se o usuário é um funcionário administrativo, é exibida uma página contendo uma mensagem de boas vindas, um quadro com os chamados pendentes aberto por ele, um quadro de avisos e o *menu* de opções (Figura 13).

Caso o usuário queira ver os detalhes de algum chamado apresentado na tela principal, deve-se clicar no número do protocolo e uma tela contendo um detalhamento do chamado é apresentada (Figura 14).

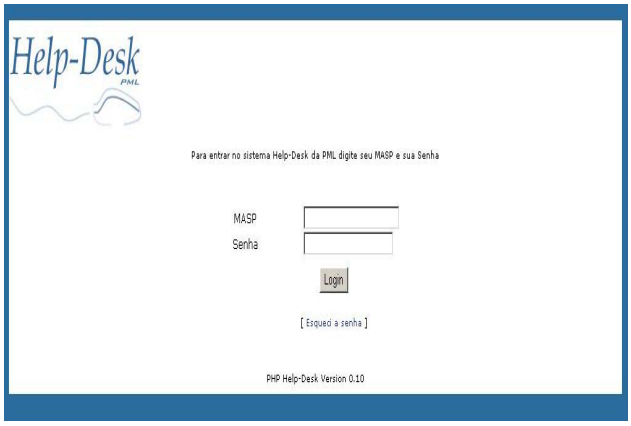

Figura 12 – Tela de Autenticação do Sistema

O *menu* de opções fica presente em qualquer página do sistema, isso ocorre para facilitar a navegação do usuário.

A Figura 15 exibe a tela mostrada a um usuário da equipe de suporte. Essa tela é muito

semelhante à mostrada ao funcionário administrativo, a diferença é na tela principal da equipe de suporte mostrar os chamados que estão sob a responsabilidade do analista ou do administrador e o quadro de avisos pode conter mensagens destinadas exclusivamente para a equipe de suporte. Além disso, o usuário pertencente à equipe de suporte poderá não apenas ver os detalhes de um chamado como também alterá-los, caso ele seja o seu responsável.

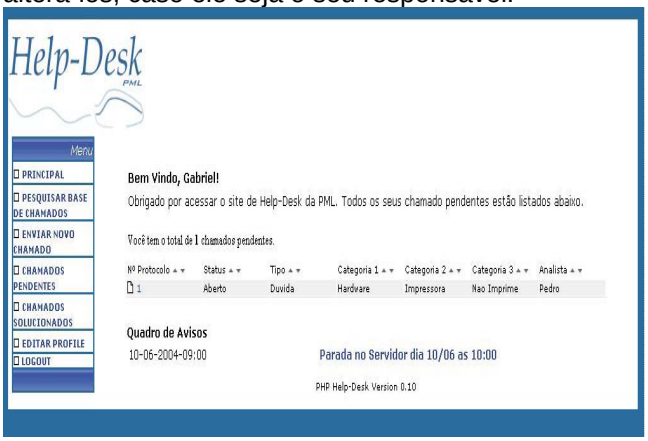

Figura 13 – Tela Principal – Funcionário Administrativo

Figura 14 – Tela Detalhamento do Chamado – Funcionário Administrativo

4 Considerações Finais

Nesta seção, são abordadas as conclusões, abrangendo os benefícios do uso do sistema *Help-Desk* e propostas para projetos futuros.

4.1 Conclusões

Este trabalho de pesquisa teve por objetivos realizar a modelagem e a implementação de um sistema *Help-Desk* para a PML.

Uma vez terminada a implementação do sistema, pôde-se notar a importância da modelagem no processo de confecção de um produto de *software*, pois além de servir de base para construção do sistema, a modelagem tem o intuito de servir também como documentação do *software* caso seja necessário realizar alguma alteração em seu código.

Figura 15 – Tela Principal – Equipe de Suporte

Como o sistema *Help-Desk* ainda está em fase de implantação, não é possível dizer com segurança que a ferramenta desenvolvida irá cumprir todos os objetivos propostos. Porém, pode-se concluir que, para o sistema funcionar com sucesso, é necessário alguns ajustes para ele se enquadrar totalmente na metodologia de serviços adotada pela PML.

Além destes ajustes, é necessário que a equipe de suporte habitue-se ao uso da ferramenta, pois quando os analistas estiverem totalmente familiarizados, o sistema *Help-Desk* será capaz de cumprir todos os seus benefícios.

4.2 Contribuições

Atualmente qualquer empresa, independente de seu porte, precisa garantir a disponibilidade de seus recursos de informática, gerenciar o pessoal de suporte responsável por esta tarefa e acompanhar o atendimento de solicitações dos usuários, solucionando o mais rapidamente possível as dificuldades encontradas.

A ausência de tal controle, ou um controle deficiente, implica em maiores despesas operacionais, desperdício de tempo e insatisfação dos usuários que dependem do funcionamento dos recursos de informática. Tudo isso se traduz em ineficiência e alto custo operacional, que precisam ser evitados pela organização.

Diante desse contexto, o sistema *Help-Desk* desenvolvido é de grande utilidade, uma vez que este oferece soluções gerenciais que auxiliam na prestação de um serviço técnico de informática com

qualidade e possibilita um melhor monitoramento das ocorrências diárias da PML.

Além do sistema desenvolvido, são contribuições deste projeto as pesquisas bibliográficas e documentais realizadas para este trabalho, uma vez que estas podem servir como fonte de pesquisa sobre: Gestão do Conhecimento, Sistemas *Help-Desk*, Raciocínio Baseado em Casos e Raciocínio Baseado em Regras.

Além disso, toda a documentação sobre o desenvolvimento do sistema poderá servir de base à construção de outras ferramentas *Help-Desk* baseadas na arquitetura *thin client*.

4.3 Trabalhos Futuros

O sistema *Help-Desk* gerado por essa pesquisa foi desenvolvido especificamente para a realidade e os problemas da PML. Sendo assim, várias funções que são desejáveis a um sistema *Help-Desk* comercial não foram implementadas, pois a prefeitura tinha necessidade por um sistema mais simples. Entre as características que poderiam ser incluídas no sistema destacam-se: i) suporte a SLA; ii) respostas Automáticas; iii) base de Conhecimento; iv) FAQ; e v) fórum de discussão.

5 Referências Bibliográficas

[1] Martins, A., Computação Baseada em Casos: Contribuições Metodológicas aos Modelos de Indexação, Avaliação, Ranking, e Similaridade de Casos. Tese de Doutorado, COPELE/CCT/UFPB, Campina Grande, 2000.

[2] G&P – Help-Desk, Disponível em: http://www.gpnet.com.br/help_desk/help_desk.shtml, Consultado em 17/03/2004.

[3] Introduction to OMG's Unified Modeling Language™ (UML®), Disponível em http://www.omg.org/gettingstarted/what_is_uml.htm, Consultado em 17/03/2004.

[4] Gornik, Davor, UML Data Modeling Profile, Disponível em http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/tp162.pdf, Consultado em 17/03/2004.

[5] Gil, A. Carlos, Como Elaborar Projetos de Pesquisa, Ed. Atlas, São Paulo, 1991.

[6] Thiollent, M, Pesquisa-Ação nas organizações, Ed. Atlas, São Paulo, Atlas, 1997.

[7] MySql, Disponível em http://www.mysql.com/, Consultado em 17/03/2004.

[8] Pressman, R.S., Engenharia de software. Ed. Makron Books, São Paulo, 2002.

[9] Coelho, Á.V. Souza; Ferneda, E.; Martins, A.; Barros, M. Alves & Gorgônio, F. Luz e, "Help Desk Inteligente em Gestão do Conhecimento: Um Tratamento Integrador de Paradigmas", Disponível em http://www.exercito.gov.br/06OMs/gabcmtex/PEG-EB/Noticias/artigo.htm, Consultado em 17/03/2004.

[10] Moresi, E.A.D., Gestão da informação e do conhecimento, Inteligência organizacional e competitiva, Editora UnB, Brasília, 2001.

[11] Junior, K. Schlünzen & Shimabukuro, M.H., Introdução a Sistemas de Gestão do Conhecimento Corporativo, XXII JAI, Campinas, 2003.

[12] Hall, L. O'Higgins & Kandel, A., Designing Fuzzy Expert Systems,. Ed. Verlang, Rheinland, 1986.

[13] Site de Suporte ao curso de Inteligência Artificial do DCC/UA, Disponível em http://www.dcc.fua.br/~dcc_ia/, Consultado em 17/03/2004.

[14] Simões, Carlo, CBR – Raciocínio Baseado em Regras, Notas de Aula, 1999.

[15] Rich, E. & Knight, K., Inteligência Artificial, Ed. Makron Books, São Paulo, 1993.

[16] Aamodt, A. & Plaza, E., Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, AI Communications, 7(i), pp.39-59, 1994.

[17] Leake, D. B., Case-Based Reasoning Experiences, Lessons & Future Directions, AAAI Press/The MIT Press, Menlo Park, 1996.

[18] Watson, I., Applying Case-Based Reasoning: Techniques for Enterprise Systems, Morgan Kaufmann Pub. Inc., San Francisco, CA, 1997.

[19] Gorgônio, F. Luz e, Uma Arquitetura de Sistemas Inteligentes de Apoio ao Usuário, Dissertação de Mestrado em Informática, UFPB, Campina Grande, 1999.

[20] O Service Desk Baseada na Web, Disponível em http://www.mmbrasil.com/solucoes/footprints/docs/features_port.pdf, Consultado em 17/03/2004.

[21] Booch, G., Rumbaugh, J. & Jacobson, I, The Unified Modeling Language User Guide, Addison-Wesley, 1999.

[22] Koch, Nora Parcus, I., Software Engineering for Adaptive Hypermedia Systems, Ludwig-Maximilians-Universität München, 2001.