BRUNO ADALBERTO DOS SANTOS

SDS – SERVICE DESK SOLUTION: UTILIZANDO OS PROCESSOS E CONCEITOS DA ITIL PARA GESTÃO DE SERVIÇOS

ASSIS

2014

BRUNO ADALBERTO DOS SANTOS

SDS – SERVICE DESK SOLUTION: UTILIZANDO OS PROCESSOS E CONCEITOS DA ITIL PARA GESTÃO DE SERVIÇOS

Trabalho de Conclusão de Curso (TCC) apresentado ao Instituto Municipal de Ensino Superior de Assis, IMESA como requisito do Curso de Análise e desenvolvimento de sistemas.

Orientador: Luiz Carlos Begosso

Área de Atuação: Desenvolvimento de Sistema

ASSIS

2014

FICHA CATALOGRÁFICA

SANTOS, Bruno Adalberto.

SDS – SERVICE DESK SOLUTION: UTILIZANDO OS PROCESSOS E CONCEITOS DA ITIL PARA GESTÃO DE SERVIÇOS/ Bruno Adalberto dos Santos. Fundação Educacional do Município de Assis – FEMA – Assis, 2014.

65 pag.

Orientador: Luiz Carlos Begosso

Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de Assis – IMESA

1. ITIL. 2. Central de serviços

CDD: 001.61

Biblioteca da FEMA

BRUNO ADALBERTO DOS SANTOS

SDS – SERVICE DESK SOLUTION: UTILIZANDO OS PROCESSOS E CONCEITOS DA ITIL PARA GESTÃO DE SERVIÇOS

Trabalho de Conclusão de Curso (TCC) apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de graduação, analisado pela seguinte comissão examinadora:

Orientador: LUIZ CARLOS BEGOSSO

Analisador: Felipe Alexandre Cardoso Pazinatto

ASSIS

2014

RESUMO

Devido ao grande crescimento dos fornecedores de softwares no país, tem aumentado a procura por sistemas de gestão de empresas de T.I, que sejam de confiança e que promovam maior apoio para as mesmas.

Atualmente podemos acompanhar a dificuldade que as empresas de T.I possuem para fazer acordos com seus clientes, estabelecer serviços e prestar suporte. Com base nestas críticas, foi elaborado um sistema que estabeleça a união destas duas partes (empresa x clientes) no ambiente do suporte, para que as empresas consigam, de maneira eficaz, fazer acordos com seus clientes, estimar tempo de execução, ouvir solicitações, prestar serviços e auxiliar os clientes em dúvidas e problemas.

Palavras Chave: ITIL; Central de serviços; Java; chamados;

ABSTRACT

Due to the large growth of software vendors in the country, has increased the demand for management of IT companies that are trustworthy and that promote increased support for the same systems.

Currently we can monitor the difficulty that IT companies have to make arrangements with their customers, establish and provide support services. Based on these reviews, a system that establishes the union of these two parties (company x clients) in support of the environment was elaborated, so that companies are able to, effectively, make arrangements with their clients, estimating runtime, hearing requests, provide services and assist customers on questions and problems.

Key words: ITIL Service Desk, Java, called;

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1. INTRODUÇÃO

Service Desk, ou Suporte, popularmente dito, é responsável pelo tratamento direto com o cliente ou grupo de clientes, fornecendo uma assistência técnica, intelectual e material sobre questões de um determinado produto ou serviço. Esta função tem por finalidade basicamente tirar dúvidas e encontrar soluções aos problemas e questões. O Suporte é a imagem e qualidade dos produtos/serviços fornecidos por ser o setor que faz a ligação entre a empresa e o cliente, por isso esta área deve ser rigorosamente controlada para que o negócio não perca sua credibilidade.

Segundo *IT Governance Institute (1998, p.134)*

> *"*O gerenciamento do processo de ***"gerenciar a central de serviço e os incidentes"*** que satisfaça ao requisito do negócio para a TI de ***"permitir o uso eficaz dos sistemas de TI através da análise e resolução de consultas, solicitações e incidentes"*** é: Inexistente quando não há suporte para resolver temas e questões de usuários. Há uma completa falta de processo de gerenciamento de incidente. A organização não reconhece que há uma questão a ser tratada*".*

Uma empresa que não gerencia uma central de serviços de maneira correta, ou seja, não reconhece que esta questão deve ser tratada com muita presteza, esta sucinta ao fracasso no mercado de trabalho. Visando a solução destes fatores, este projeto tem por finalidade, construir uma ferramenta que aperfeiçoe um atendimento ao cliente, garantindo ao mesmo um rápido esclarecimento sobre dúvidas e total orientação quanto aos problemas relatados, buscando sempre uma posição sólida no primeiro atendimento. Caso não seja possível sanar o problema do cliente num primeiro momento, o sistema deverá acompanhar o desenvolvimento do chamado

aberto, para que o cliente esteja sempre informado e possa realizar interações quanto ao andamento do mesmo. Para a construção deste projeto, Serão utilizadas algumas características do modelo de referência da ITIL *(Information Technology Infrastructure Library)* que tem por objetivo otimizar e melhorar os processos da área de Suporte *.*

Segundo Moraes e Mariano (2008):

> A ITIL foi desenvolvida pela CCTA (*Central Computer and Telecommunication Agency*), atualmente chamada OGC (*Office of Government Commerce*), do Reino Unido, no final dos anos 80, sendo documentada em um conjunto de livros que descrevem um modelo de referência com as melhores práticas para um efetivo Gerenciamento dos Serviços de TI.

A ITIL foi criada para melhorar os processos dos departamentos de TI do governo Britânico. Em 1990 a ITIL acabou se tornando um padrão de fato para todo o mundo, sendo posteriormente adaptado pelas maiores empresas de TI do mundo.

Para Sodré & Souza (2007) A ITIL é baseada na necessidade de se prestar serviços de alta qualidade, com ênfase nas relações com os clientes. A organização de TI deverá cumprir os acordos com o cliente, o que significa manter boas relações com os clientes e parceiros, tais como fornecedores.

De acordo com Lima (2007) um dos principais fatores do crescente sucesso da ITIL é a sua flexibilidade, pois deve ser implementado como parte de uma metodologia de negócios que envolvem os processos de gerenciamento de serviços. Por isso, pode ser utilizado por várias aplicações gerenciais, pois não é fixo a uma regra de negócio, e sim abrangente a todo e qualquer tipo de aplicação gerencial.

Portanto podemos dizer que a ITIL não é o negócio, propriamente dito, e sim o conjunto de boas práticas que compõe a fundamentação do negócio.

O presente sistema será desenvolvido utilizando a tecnologia Java e será denominado *SDS – Service Desk Solutions*. O sistema será uma aplicação gerencial interna para a empresa, não disponível para os seus clientes. A aplicação será utilizada também pelos programadores e responsáveis por testes do sistema, porém, o foco principal do SDS é o Analista de Suporte.

1.1 OBJETIVOS

O presente trabalho tem como objetivo geral o desenvolvimento de um sistema capaz de controlar os processos do setor de suporte, seguindo as métricas de gestão do ITIL. O software será capaz de produzir informações que auxiliarão os gestores na tomada de decisão, a fim de estabelecer um controle especifico sobre o setor.

Como objetivo específico, serão desenvolvidos os módulos de suporte a atendimento ao cliente, gestão de configurações e solicitações, gerenciamento de chamados e registros de problemas, além de análises e levantamentos de desempenho do setor e dos analistas.

1.2 JUSTIFICATIVAS

Devido ao conhecimento no ambiente de suporte, espera-se auxiliar na boa comunicação entre os clientes e a empresa, provendo uma ferramenta que auxilie o atendente em suas tomadas de decisões de maneira simples e rápida. O sistema permitirá o compartilhamento de soluções para diversos tipos de ocasiões, sendo elas para simples dúvidas ou até problemas complexos, além de manter configurações internas como controladores de versões e mudanças.

1.3 MOTIVAÇÃO

A principal motivação para o desenvolvimento do SDS é que ele poderá ajudar os gestores de Suporte, entre outros departamentos envolvidos em uma empresa, a garantir um atendimento rápido e eficaz aos clientes de maneira inteligente e estruturada, oque resultará em satisfação pelos serviços prestados.

Outro fator que contribuiu na criação deste projeto foi o estudo da biblioteca ITIL, que faz com que o sistema seja um diferencial no mercado, por adotar estes padrões de gestão, que por sinal são respeitados mundialmente pelas maiores empresas da área de TI.

1.4 PÚBLICO ALVO

O presente trabalho almeja contribuir para o bom funcionamento e desempenho dos atendimentos na área de suporte, garantindo agilidade, fonte de raciocínio e opções de solução. Espera-se que o sistema a ser desenvolvido possa atender as expectativas da empresa e do cliente proporcionando um atendimento rápido e eficaz.

O SDS pode ser utilizado por toda e qualquer empresa que possui uma área de suporte, ou até mesmo telemarketing e pós-vendas.

2. REFERENCIAL TEÓRICO

O projeto utiliza várias ferramentas em sua composição. Foi necessário realizar diversas pesquisas sobre *frameworks* que são utilizados na linguagem Java para Web. Os mecanismos escolhidos foram embasados nas realidades das empresas de T.I na atualidade, e que podem, de certa forma, comportar a regra de negócio de maneira eficaz e ágil.

2.1 ITIL

2.1.1 O que é?

Segundo Moraes e Mariano (2008):

> A ITIL foi desenvolvida pela CCTA (*Central Computer and Telecommunication Agency*), atualmente chamada OGC (*Office of Government Commerce*), do Reino Unido, no final dos anos 80, sendo documentada em um conjunto de livros que descrevem um modelo de referência com as melhores práticas para um efetivo Gerenciamento dos Serviços de TI.

Para Sodré & Souza (2007) A ITIL é baseado na necessidade de se prestar serviços de alta qualidade, com ênfase nas relações com os clientes. A organização de TI deverá cumprir os acordos com o cliente, o que significa manter boas relações com os clientes e parceiros, tais como fornecedores.

2.1.2 Como pode ser utilizado?

De acordo com Lima (2007) um dos principais fatores do crescente sucesso da ITIL é a sua flexibilidade, pois deve ser implementado como parte de uma metodologia de negócios que envolvem os processos de gerenciamento de serviços. Por isso, pode ser utilizado por várias aplicações gerenciais, pois não é fixo a uma

regra de negócio, e sim abrangente a todo e qualquer tipo de aplicação gerencial. Portanto podemos dizer que a ITIL não é o negócio, propriamente dito, e sim que o conjunto de boas práticas que compõe a fundamentação do negócio. Para o desenvolvimento do sistema, os módulos gerenciadores foram definidos de acordo com a seguinte imagem:

Figura 1 Representação dos processos envolvidos em uma central de serviços, segundo as regulamentações da ITIL.

2.1.3 Help Desk

HelpDesk, ou Suporte, popularmente dito, é responsável pelo tratamento direto com o cliente ou grupo de clientes, fornecendo uma assistência técnica, intelectual e material sobre questões de um determinado produto ou serviço. Esta função tem por finalidade basicamente tirar dúvidas e encontrar soluções aos problemas e questões.

Esta função tem por finalidade atender o cliente em seu primeiro nível de atendimento, é a equipe que faz a ligação entre o provedor de serviço e o cliente e é o ponto principal que o cliente possui para solicitar serviços e relatar incidentes. O Help Desk está na linha direta de frente para qualquer impacto dos acordos de níveis de serviço, bom como é responsável pelo rápido fluxo dessa informação.

As atividades realizadas pelo Help Desk são:

- Atender chamadas
- Registrar incidentes
- Suporte inicial e classificação do nível do atendimento
- Atribuir os incidentes após a confirmação do Cliente.
- Ser responsável pelos incidentes, monitoramento, acompanhamento e comunicação com clientes e também executores.
- Atender as solicitações de serviço e alteração de padrão.
- Manter os clientes informados quanto ao status e progressos das suas solicitações, registrando as interações devidas.
- Fornecer informação de gerenciamento e sugestões de melhoria de serviços
- Detectar necessidades de capacitação
- Contribuir para a identificação de problemas, pelos registros de incidentes
- Identificar as oportunidades de apoio ao negócio.

2.1.4 central de serviços

A central de serviços ou *Service Desk* em inglês, é responsável por ser o ambiente de trabalho do Help Desk, ou equipe de suporte. Ela é responsável por organizar os níveis de atendimentos, quem irá atender aos usuários e quem irá resolver problemas e solicitações. Ter uma área especifica para o suporte garante ao cliente um rápido atendimento, com responsáveis e experientes no assunto. Isso garante também um trabalho adequado a equipe de suporte, pois o responsável pelos problemas não será interrompido por chamadas diretas dos usuários

A construção da central de serviços tem como objetivo:

- Funcionar como ponto central de contato (SPOC – *Single point of contact*) entre os usuários e a empresa de T.I. A central de serviços funciona como o primeiro nível de atendimento aos clientes.
- Restaurar os serviços sempre que possível. A equipe deve estar sempre equipada com tecnologias que facilitem o acesso a informações, como erros conhecidos para que possam obter uma solução o mais breve possível.
- Prover suporte com qualidade a respeito do negócio. É de fator primordial que a equipe esteja bem treinada quanto a todas as áreas de atuação do sistema no qual presta o suporte.
- Gerenciar todos os incidentes até o encerramento do mesmo, afim de gerenciador todos os acordos de níveis de serviço.
- Dar suporte a mudanças constantes do sistema, em um web site informativo, ou bases de conhecimento.
- Maximizar a disponibilidade do serviço.

A seguinte figura representa a ação da central de serviços e todos os níveis de serviços que um cliente pode se submeter, de acordo com o incidente reportado:

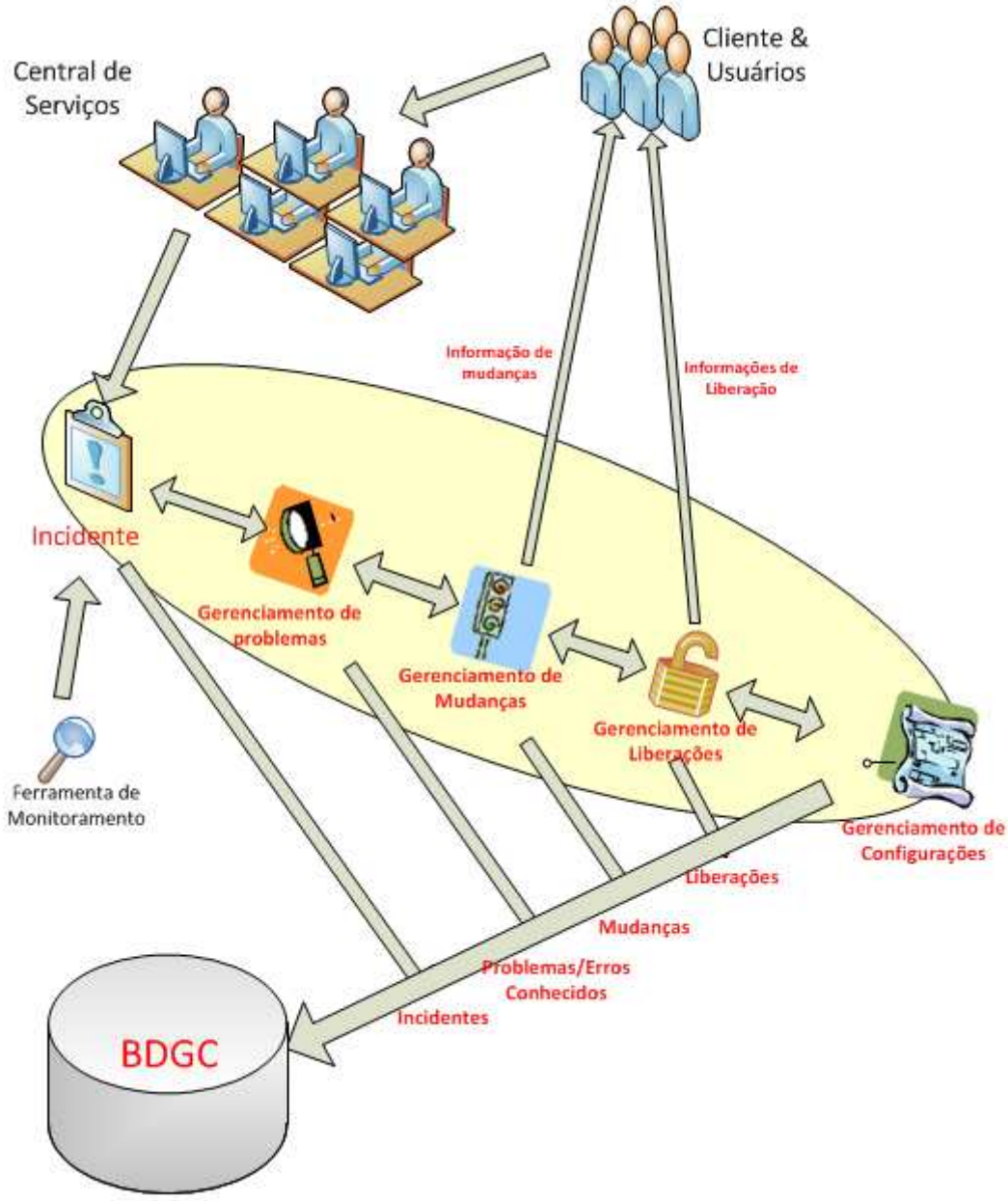


Figura 2 – Central de Serviços, todos os níveis de um incidente gerenciados pela central de serviços.

No sistema SDS será construída uma central de serviços, a central de serviços é uma função das funcionalidades padrões da ITIL, conforme ilustrada na imagem acima.

2.1.4.1 gestão de incidentes

Existe uma estreita relação entre os processos de Gerenciamento de Incidentes, Gerenciamento de Problemas e Gerenciamento de Alterações, assim como com a função de Help Desk. Caso esta relação não seja controlada, as alterações implantadas podem gerar novos incidentes, motivo pelo qual é necessário o seu monitoramento. É recomendável que os registros dos incidentes estejam armazenados na mesma CMDB, bem como os registros dos problemas, erros conhecidos e alterações, para melhorar as interfaces.

A gestão de incidentes, serve para relatar tudo oque foi abordado no primeiro decorrer do atendimento, colocando palavras chaves para que os gestores de T.I possam analisar oque mais gerou incidentes para tomadas de decisões.

2.1.4.2 gestão de problemas

Este processo tem como objetivo encaminhar os incidentes que foram taxados como sem solução para que sejam analisados e processados.

Erros conhecidos, soluções de contornos/definitivas, *quick fixes (*reparos rápidos*)* são fornecidos ao gerenciador de incidentes pelo gerenciador de problemas.

Uma funcionalidade importante do gerenciador de problemas é gerar chamados automáticos para os clientes, quando registrado ou vinculado um problema ao seu atendimento, para que o mesmo tenha acesso as interações. Um problema, gera uma tarefa, em paralelo é gerado um chamado também, assim que a tarefa é processada e o problema é resolvido, são disparadas mensagens a todos

os clientes que tiveram aquele problema, para que realizem atualizações ou soluções.

Segundo a OGC Service Support (2003), o Gerenciamento de Problemas tem aspectos reativos e proativos. O aspecto reativo é concentrado com a resolução de problemas em resposta a um ou mais incidentes. O Gerenciamento de Problemas proativo é concentrado em identificar e resolver problemas e erros conhecidos antes da ocorrência de incidentes.

2.1.4.3 gestão de chamados

Um chamado é um serviço que esta além do Help Desk, ou seja, é uma equipe com foco em resolver problemas maiores. Os chamados são acordados com os clientes de acordo com o nível de urgência ou prioridades de execução.

O chamado pode gerar uma taxa, de acordo com o tipo do serviço prestado, porém o cliente precisa estar ciente da taxa antes da abertura do chamado, para que não haja desacordos no momento da realização. Pode ser considerado também um serviço que esta além das atribuições da empresa, como por exemplo instalação de sistemas que trabalham em conjunto, exemplo, um sistema de vendas, necessita de um TEF (maquina de cartão), a empresa de T.I tem a responsabilidade de disponibilizar o recurso para a implantação do TEF, porém, não tem a obrigação de instalar e configurar o mesmo para o cliente, caso solicitado, pode ser aberto um chamado com uma taxa, para que seja configurado o TEF no sistema.

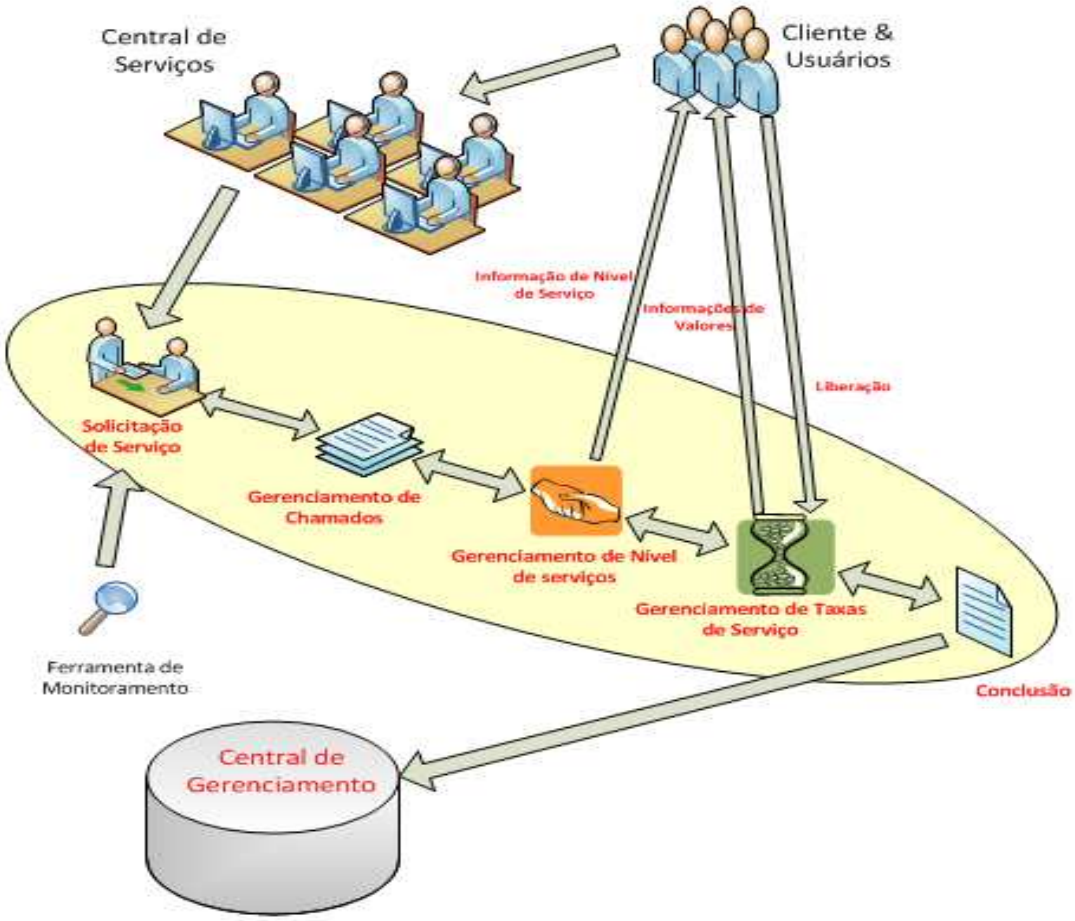


Figura 3 – Acordos de níveis de serviços. Gestão e todos os procedimentos envolvidos na abertura de um chamado (solicitação de serviço).

2.2 ANÁLISE E PROGRAMAÇÃO ORIENTADA A OBJETOS

A fim de estabelecer uma visão real do caso, a análise orientada a objetos nos provê facilidade para transpor um objeto do mundo real em representações como um modelo. Utilizando esta facilidade conseguimos criar classes e objetos que interagem entre si, além de métodos e conceitos que nos da facilidade para “reaproveitar” o código.

É um paradigma para o desenvolvimento de software que se baseia na utilização de componentes individuais (objetos) que interagem entre si para a elaboração de sistemas mais complexos.

2.2.1 Classes E Objetos

2.2.1.1 classes

Segundo Ambler (1997, p.121), “A Definição de uma classe descreve a organização incluindo-se os dados e a funcionalidade dos objetos que serão criados a partir dela.”.

Uma classe também pode ser caracterizada como um tipo, onde cada uma possui suas particularidades e características, como por exemplo, uma pessoa, todas as pessoas possuem nome, endereço, data de nascimento entre outras características, logo podemos concluir que Pessoa é uma classe, pois, possui atributos que são comuns para muitas pessoas (João, Maria, José, Bruno...).

2.2.1.2 objetos

De acordo com Ambler (1997, p.125), “Os objetos são entidades do mundo real, e é por esta razão que necessitamos do conceito de objeto”.

Pode-se dizer que, objetos são materializações de classes, ou seja, algo que possui as características e ações de uma classe, como no exemplo acima, João é uma pessoa, logo foi materializado com os moldes de sua respectiva classe.

2.2.2 atributos e métodos

2.2.2.1 atributos

Atributos correspondem aos conhecimentos de uma classe, podem ser considerados como características da classe, como por exemplo, nome, idade, endereço e etc.

2.2.2.2 métodos

Métodos são as ações de uma classe, todos os comportamentos e reações que um objeto desta classe pode ter, seguindo o exemplo de pessoa, seus métodos seriam basicamente, andar, correr, falar, comer, dormir entre outros.

2.2.3 abstração, encapsulamento e ocultamento de informações

2.2.3.1 abstração

Como estamos em um mundo repleto de coisas, fica um tanto complicado criar classes, porém podemos utilizar uma maneira generalizada de ver as coisas. Estas formas podem variar de negócio para negócio, por exemplo, podemos dizer que para uma universidade uma pessoa possui nome, endereço, telefone, e grau de

escolaridade. Já para uma vaga de emprego, uma pessoa precisa ter, nome, endereço, telefone, certificações, conhecimento na área, experiências dentre outras.

Segundo Ambler (1997, p.129), “Abstração é basicamente uma questão de análise, pois lida com os conhecimentos e as ações subjacentes a uma classe”.

2.2.3.2 encapsulamento

Encapsulamento é a forma de modificar os comportamentos de características ou ações de uma classe (atributos e métodos).

De acordo com Ambler (1997, p.130):

> O encapsulamento é basicamente um aspecto do projeto que discute como as funções serão compartimentalizadas em um sistema. Ninguém precisa saber como uma função é implementada para pode utilizá-la. A implicação do encapsulamento é que você pode construir os componentes da maneira que desejar, e mais tarde modificar implementação sem que isto afete outras partes do sistema (contanto que a interface para estes componentes não seja alterada).

2.2.3.3 ocultamento de informações

Ocultamento de informações, como o nome já diz, é a maneira de restringir acessos a determinados atributos e métodos que o programador define. Esta técnica serve basicamente de manter a integridade das informações presentes em uma classe. Esta forma de pensar foi basicamente interpretada nas atitudes do mundo

real, por exemplo, quando preciso saber o nome de uma pessoa, vou até ela e pergunto. O conceito de ocultamento de informações é a mesma coisa, para ter acesso a tais atributos deve solicitar uma autorização, ou pedir permissão.

2.2.4 herança e polimorfismo

2.2.4.1 herança

É comum encontrar similaridades entre classes, ou seja, várias classes como muitos atributos e métodos em comum, como, por exemplo, uma classe Aluno e Professor, ambos comem, bebem, dormem, falam e também possuem um nome, idade, endereço e etc. Oque diferencia uma classe da outra são os atributos/métodos específicos de cada uma, como por exemplo, Aluno assiste Aula, e professor prepara a aula. De acordo com a técnica de herança, podemos criar um tipo em comum, ou seja, uma abstração, neste caso “Pessoa”. Toda pessoa possui, nome, endereço, idade e etc. Um aluno é uma pessoa, assim como um professor também, então podemos atribuir a eles todas as características de pessoas e adicionar apenas os “extras” de cada classe. Este procedimento nos da a facilidade de não precisar repetir diversas vezes um código idêntico para classes em comum, chamada técnica de reaproveitamento de código.

Segundo Ambler(1997, p.133), a definição de herança é: “Permitir tirar proveito das similaridades entre as classes representadas por relacionamentos do tipo “é” ou “é Semelhante””.

2.2.4.2 Polimorfismo

Polimorfismo é a técnica de atribuir comportamentos diferentes a um mesmo objeto. Quando utilizamos herança, podemos fazer com que um objetos se comporte de várias formas, por exemplo, podemos dizer que um objeto do tipo pessoa, pode ser instanciado por um do tipo Aluno, ou Professor.

Para Ambler (1997, p.158), “O polimorfismo estabelece que um determinado objeto pode assumir variadas formas, e que os outros objetos podem interagir com ele sem se preocuparem com a sua forma especifica num dado momento”. Desde que o objeto seja de um tipo como Pessoa, por exemplo, não fará diferença para aplicação saber se ele é um Aluno, professor, diretor, apenas que ele é uma pessoa.

3. TECNOLOGIAS UTILIZADAS PARA O DESENVOLVIMENTO DO SDS

3.1 TECNOLOGIA JAVA

3.1.1 História do Java

Em 1991, durante a produção de microprocessadores que estavam tornando possível o desenvolvimento de aplicações. A empresa Sun MicroSystems, em 1991, financiou um projeto de pesquisa corporativo afim de criar uma nova linguagem de programação, baseada em c++.

Porém, o mercado de computadores para uso pessoal, não estava se desenvolvendo direito, a Sun acreditava que o projeto não havia dado certo, no

entanto, por uma feliz causalidade, a World Wide Web (WWW.) explodiu em popularidade e a equipe da Sun viu a possibilidade de adicionar serviços dinâmicos, como imagens, interatividade com as paginas da WEB, dando assim, uma nova vida ao projeto.

O nome sugeriu, pois a linguagem originada do projeto Green, chamou – se OaK, porém foi descoberto que já havia uma linguagem de programação com este mesmo nome, sendo assim, quando uma equipe da Sun, visitou uma cafeteria local, o nome Java (cidade de origem de um tipo de café importado) foi sugerido, e nunca mais foi alterado (DEITEL, 2005, p.6).

3.1.2 JVM (Java Virtual Machine) e JDK (Java Development Kit)

A execução de um programa escrito na linguagem Java acontece da seguinte forma: O Código fonte (.java) é compilado pelo compilador Java, e esta compilação gera bytecodes (.class). A JVM transforma estes bytecodes em linguagem de máquina para que o hardware execute a aplicação traduzida.

De acordo com o escopo acima, podemos observar que a JVM é um intermediador entre a aplicação java e o computador, pois sem a ela o computador não é capaz de traduzir sozinho os arquivos .class em linguagem de máquina. Os passos descritos também evidenciam que a portabilidade da linguagem Java depende inteiramente da JVM, pois, para que uma aplicação java seja iniciada, basta que o computador tenha a mesma instalada (DEITEL, 2005 p.10).

O JDK é o pacote de desenvolvimento Java, que pode ser adquirido gratuitamente. Este pacote inclui as API’s, o Compilador e também a JVM, ou seja, tudo oque é necessário para o desenvolvimento e execução de uma aplicação Java.

3.1.3 JAVA SERVER FACES (JSF) e PRIME FACES

Java Server Faces (JSF) é uma tecnologia que nos permite criar aplicações Java Web utilizando componentes visuais, fazendo com que o desenvolvedor não se preocupe com JavaScript e HTML. Basta adicionar os componentes que eles serão renderizados e exibidos no formato HTML.

Os módulos que compõe o JSF são: Componentes, eventos, validações e conversões, navegabilidade e backBeans. BackBeans são classes simples, não herdam de ninguém e também não há necessidades de implementar interfaces.

As vantagens de se utilizar o JSF são: MVC para aplicações Web, padrão *model, view e controller,* “Fácil” de usar, componentes extensíveis, boa demanda no mercado dos desenvolvedores e de código aberto.

Prime Faces é um componente mais sofisticado, é uma extensão do JSF, para criação de aplicações mais ricas visualmente. Seus componentes foram construídos para trabalhar com o Ajax por padrão, isto é, não é necessário nenhum tipo de esforço extra do programador, para realizar chamadas assíncronas ao servidor. Além disso, o Prime Faces da suporte a criação de funcionalidades que fazer o uso do *Ajax Push* e permite aplicação de temas para mudar todo o visual da página.

3.2. padrão de desenvolvimento - mvc

O modelo MVC é uma forma de desenvolvimento que ajuda na manutenção do código, tendo em vista que é dividido por partes, e cada parte tem seu real

sentido na aplicação, além de ser um padrão muito aceito no desenvolvimento de aplicações em todo o mundo. Segundo Gonçalves (2008, p.141), “MVC é um conceito (paradigma) de desenvolvimento e design que tenta separar uma aplicação WEB em três partes distintas”. Estas partes são respectivamente:

3.2.1. model

Model é a camada de modelo da aplicação, onde ficam representados os dados do programa. Este modelo não tem conhecimento específico, das demais camadas, nem sequer contém referencia delas. Normalmente as classes de modelo, são o espelho das tabelas do banco de dados. Segundo Gonçalves(2008, p.141), “Model, são as classes que trabalham no armazenamento e busca de dados”.

3.2.2 view

View é a apresentação, ou seja, o contato do usuário com as informações. A view, reflete inteiramente em suas “telas”, os dados a serem inseridos em um modelo, como por exemplo um formulário de cadastro de clientes, os campos a serem preenchidos na view são exatamente os de armazenamento no model.

3.2.3 controller

Controller são os controladores, a camada que faz o controle e fluxo das informações, ele responde as requisições prestadas pelo usuário, validando assim, se é possível retornar informações. É o responsável por gerenciar as regras de negócio.

3.3. MAPEAMENTO OBJETO/RELACIONAL E HIBERNATE

3.3.1 mapeamento objeto/relacional

Segundo Bauer e King (2007, p.25),

> “Mapeamento objeto/relacional é a persistência automatizada (e transparente) dos objetos em uma aplicação Java para as tabelas de um banco de dados relacional, utilizando metadados que descrevem o mapeamento entre os objetos e o banco de dados”.

Mapeamento objeto/relacional é o conceito de “transformar” um banco de dados relacional, em orientado a objetos, fazendo com que o banco se adapte facilmente com o código orientado a objetos.

3.3.1.1 VANTAGENS DE ORM (MAPEAMENTO OBJETO/RELACIONAL)

Uma das vantagens de utilizar ORM é proteger os desenvolvedores dos SQL’s confusos, difíceis de resolver e de prestar manutenções. Porém, é de extrema importância que um desenvolvedor ORM saiba muito bem gerenciador SQL’s sem a ferramenta, para que o mesmo entenda todos os procedimentos que estão sendo gerados através da ferramenta ORM.

A utilização do conceito de ORM, pode aumentar o empenho e diminuir o tempo de entrega de um projeto, pois livra o programador de executar certos comandos SQL’s e focar apenas nas regras de negócio e na elaboração do sistema Bauer e King (2007, p.29).

3.3.1.1.1 PRODUTIVIDADE

O código relacionado a banco de dados (SQL’s) em toda linguagem de programação torna-se um tanto “pesado” para o programador. O ORM ajudará o programador a diminuir o tempo de trabalho com ferramentas de persistência que podem realizar operações diversas (rotineiras) como CRUD, transações, pesquisa entre classes e etc.

3.3.1.1.2 MANUTENIBILIDADE

De acordo com Bauer e King (2007, p.29):

> “Menos linhas de código tornam o sistema mais legível (de mais fácil entendimento), pois enfatiza na lógica de negócio ao invés do “encanamento” (termo técnico para o código que gerenciam o acesso aos dados, persistência, e assim por diante). E o mais importante, um sistema com menos código é mais fácil de refatorar.”.

3.3.1.1.3 DESEMPENHO

Uma questão discutida é em relação a velocidade de consultas, realizadas manualmente (SQL’s) ou automaticamente. Claro que um procedimento feito diretamente no banco de dados torna-se mais rápido, porém o tempo de codificação pode não compensar. Contudo, com os processamentos atuais, a diferença acaba sendo despercebida.

3.3.1.1.4 INDEPENDÊNCIA DE FORNECEDOR

Uma ferramenta ORM garante ao programador a facilidade de escolher o banco de dados no qual deseja trabalhar, pois a maioria das ferramentas possuem vários dialetos (linguagens de banco) em suas implementações. Isso garante ao programador a disponibilidade e portabilidade de sua aplicação a qualquer tipo de banco de dados, sem precisar fazer alterações no código da aplicação. Porém, isso

também não garante que a implementação feita fique disponível a todos os bancos de dados, pois a competência de bancos de dados difere.

3.3.2 Hibernate

Segundo Bauer e King (2007, p.29), “O Hibernate é uma ferramenta de mapeamento objeto/relacional completa que fornece todos os benefícios de um ORM”

É considerado um framework de persistência de dados muito usado na atualidade pois provê ao programador maior facilidade de comunicação com os bancos de dados, e garante ao mesmo uma maior portabilidade do sistema, pois para a troca de um banco de dados é necessário apenas a mudança do dialeto, ou seja, da forma que o hibernate responderá aos comandos.

3.4. BANCO DE DADOS

3.4.1 mysql

3.4.1.1 Oque é o MySQL?

Segundo Milani (2006, p.22):

> “O MySQL é um servidor e gerenciador de banco de dados (SGBD) relacional, de licença dupla (sendo uma delas de software livre), projetado inicialmente para trabalhar com aplicações de pequeno e médio portes, mas

> hoje atendendo a aplicações de grande porte e com mais vantagens que seus concorrentes.”

Devido a estes conceitos, a aplicação será desenvolvida utilizando como base o banco de dados MySQL, que além de possuir uma licença gratuita, é um sistema que contempla, também, sistemas de grande porte. Um dos fatores que chama a atenção no MySQL é a simplicidade de utilização, a criação de SQL’s , manutenções do banco e também a velocidade de processamento dos comandos.

3.4.1.2 Breve História

O MySQL foi fundado quando os desenvolvedores David Axmark, Allan Larsson e Michael “Monty” Widenius, na década de 90, precisavam de um sistema que fosse compatível com suas rotinas ISAM, utilizadas em suas aplicações e tabelas. Depois de algumas tentativas de utilizações de API’s, passaram a usar mSQL, contudo, não era rápida como eles precisavam. Com base na mSQL, criaram em C/C++ uma nova API, chamada MySQL. (MILLANI 2006, p.23)

Com o ótimo resultado gerado pela API, o MySQL tornou-se popularmente conhecido por suas características de fácil acesso e cada vez mais utilizados.

4. FERRAMENTAS UTILIZADAS

4.1 ECLIPSE KEPLER

A ferramenta de desenvolvimento Eclipse é uma IDE (*integrated development environment*). Diferente de uma RAD, que é uma ferramenta de seleção de componentes através do arrastar e soltar do mouse. O Eclipse é a IDE líder no mercado dos desenvolvedores. Foi planejada por um consórcio liderado pela IBM, e é de código livre.

Uma IDE tem como objetivo principal, auxiliar o desenvolvedor na criação dos códigos da linguagem a qual ela representa, dando ao programador, facilidade de digitação, auxilio em importações de bibliotecas e etc.

Por estes motivos, a versão escolhida foi a Version: Kepler Service Release 1.

4.2 NAVICAT PREMMIUM

O software de gestão de banco de dados Navicat foi utilizado neste trabalho por ser uma ferramenta de fácil manipulação e que auxilia bastante na parte de modelagem do banco de dados. Para o projeto, foi utilizada a versão 10.1.1 – Premium, esta versão nos permite ter visões gráficas do banco de dados e do relacionamento das tabelas, para ter uma visão detalhada dos relacionamentos.

4.3 ASTAH PROFESSIONAL

O Astah, anteriormente chamado JUDE, é um software para modelagem UML.É desenvolvido na plataforma Java, oque garante uma grande portabilidade do software por ser interpretado pela JVM. A versão que foi utilizada no projeto foi professional 6.1 (Model version: 32)

4.4 MICROSOFT PROJECT 2013

O MS Project é um software baseado em tarefas e processos. É um sistema desenvolvido pela Microsoft, capaz de gerar gráfico temporal de acordo com tarefas e processos criados sobre ele. Foi utilizado para criação do calculo de tempo para o desenvolvimento e geração do cronograma.

4.5 MICROSOFT VISIO 2010

O MS Visio é um aplicativo de elaboração de diagramas diversos e além da UML. Ele foi utilizado para gerar diagramas da ITIL e também o WBS.

5. PLANEJAMENTO DO SISTEMA

Para inicio do desenvolvimento foram traçados objetivos e metas para o desenvolvimento do produto. Afim de, ilustrar o projeto do sistema, foram criados alguns diagramas e cronogramas.

5.1 ESTRUTURA ANALÍTICA DO PROJETO – WBS

WBS (*Work breakdown structure*) é o diagrama responsável por mostrar analiticamente todos os processos que estão envolvidos no desenvolvimento do produto, é também um auxiliar para a estimativa de tempo que levará o decorrer do projeto.

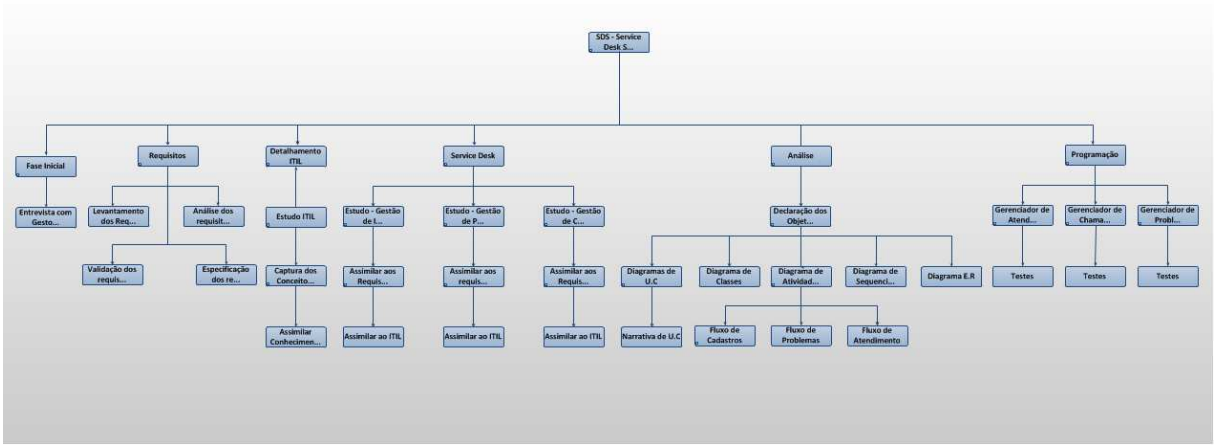


Figura 4 - WBS, representação dos processos envolvidos desde a fase inicial até a implantação do sistema.

5.2 REPRESENTAÇÃO DAS ATIVIDADES

Com base no WBS, podemos construir uma estimativa de tempo, dividindo as tarefas em partes, e também gerando Sub-tarefas, para que o projeto tenha melhor aproveitamento de tempo, para que possa reduzir também, o prazo de entrega.

Nome da tarefa	Duração	Início	Término
Fase Inicial	**1 dia**	**Sex 01/11/13**	**Sex 01/11/13**
Entrevista com Gestores e Analistas	1 dia	Sex 01/11/13	Sex 01/11/13
Requisitos	**5 dias**	**Sáb 02/11/13**	**Qui 07/11/13**
Levantamento dos Requisitos	1 dia	Sáb 02/11/13	Sáb 02/11/13
Análise dos Requisitos	1 dia	Dom 03/11/13	Dom 03/11/13
Validação dos requisitos	2 dias	Seg 04/11/13	Ter 05/11/13
Especificação dos Requisitos	2 dias	Qua 06/11/13	Qui 07/11/13
Detalhamento do ITIL	**11 dias**	**Sex 08/11/13**	**Sex 22/11/13**
Estudo do ITIL	4 dias	Sex 08/11/13	Qua 13/11/13
Captura dos conceitos para a área de atuação do sistema	3 dias	Qui 14/11/13	Seg 18/11/13
Assimilar conhecimentos as requisitos do projeto	4 dias	Ter 19/11/13	Sex 22/11/13
Service Desk	**11 dias**	**Sáb 23/11/13**	**Sex 06/12/13**
Estudo Sobre Gerenciamento de Requisitos	2 dias	Sáb 23/11/13	Seg 25/11/13
Estudo Sobre Gerenciamento de Problemas	4 dias	Ter 26/11/13	Sex 29/11/13
Estudo Sobre Gerenciamento de Chamados	2 dias	Sáb 30/11/13	Seg 02/12/13
Assimilar os Gerenciadores aos Requisitos	4 dias	Ter 03/12/13	Sex 06/12/13
Análise	**11 dias**	**Sáb 07/12/13**	**Sex 20/12/13**
Declaração Dos Objetivos	3 dias	Sáb 07/12/13	Ter 10/12/13
Diagramas de Casos de Uso	3 dias	Qua 11/12/13	Sex 13/12/13
Diagrama de Classes	3 dias	Sáb 14/12/13	Ter 17/12/13
Diagrama de Atividades	**3 dias**	**Qua 18/12/13**	**Sex 20/12/13**
Programação	**110 dias**	**Sáb 21/12/13**	**Qui 22/05/14**
Módulo - Gerenciador de Chamados	30 dias	Sáb 21/12/13	Qui 30/01/14
Módulo - Gerenciador de Problemas	30 dias	Sex 31/01/14	Qui 13/03/14
Módulo - Gerenciador de Atendimentos	30 dias	Sex 14/03/14	Qui 24/04/14
Testes	20 dias	Sex 25/04/14	Qui 22/05/14

Figura 5 – Representação das atividades e subatividades definidas para a construção do documento e sistema

Com base nas atividades e no WBS, podemos gerar um cronograma do projeto

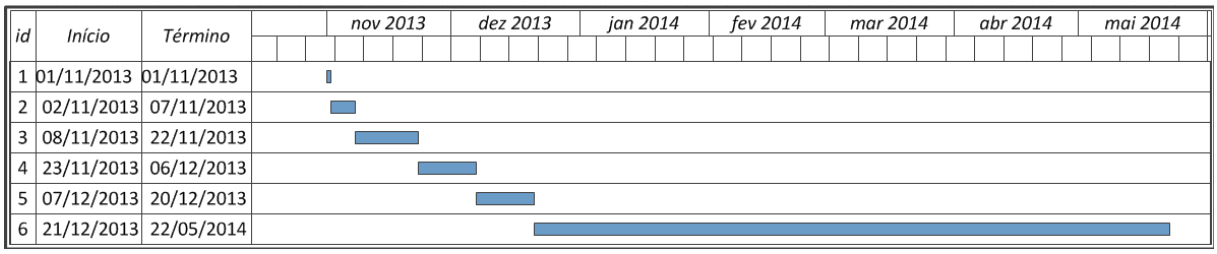


Figura 6 – Cronograma de desenvolvimento do SDS

6. MODELAGEM DO SISTEMA

Neste capítulo serão levantadas as especificações e a construção do modelo proposto no trabalho. O problema que fora abordado consistiu em desenvolver um sistema que gerencie uma central de serviços, capaz de assegurar o bom fluxo de atendimentos ao cliente de maneira mais rápida e eficaz, através de meios de consulta diversos que garantem ao cliente um posicionamento concreto e organizado com relação à questão levantada no inicio do atendimento.

6.1 DIAGRAMAS

6.1.1 casos de uso

Segundo Guedes(2011, p. 30):

> “O diagrama de caso de uso é o diagrama mais geral e informal da UML, utilizado normalmente nas fases de levantamento e análise dos requisitos do sistema, embora venha a ser consultado durante todo o processo de modelagem e possa servir de base para outros diagramas. Apresenta uma linguagem simples e de fácil compreensão para que os usuários possam ter uma ideia geral de como o sistema irá se comportar.”

O Diagrama de caso de uso tem por objetivo apresentar uma imagem das ações de um usuário em um sistema, todas as suas rotinas e permissões dentro do mesmo. É um digrama que facilita a visualização do sistema, sem mesmo tê-lo construído.

Seguindo estes conceitos, foram elaborados alguns casos de uso com as características principais dos fluxos do sistema.

6.1.1.1. caso de uso: atendimento

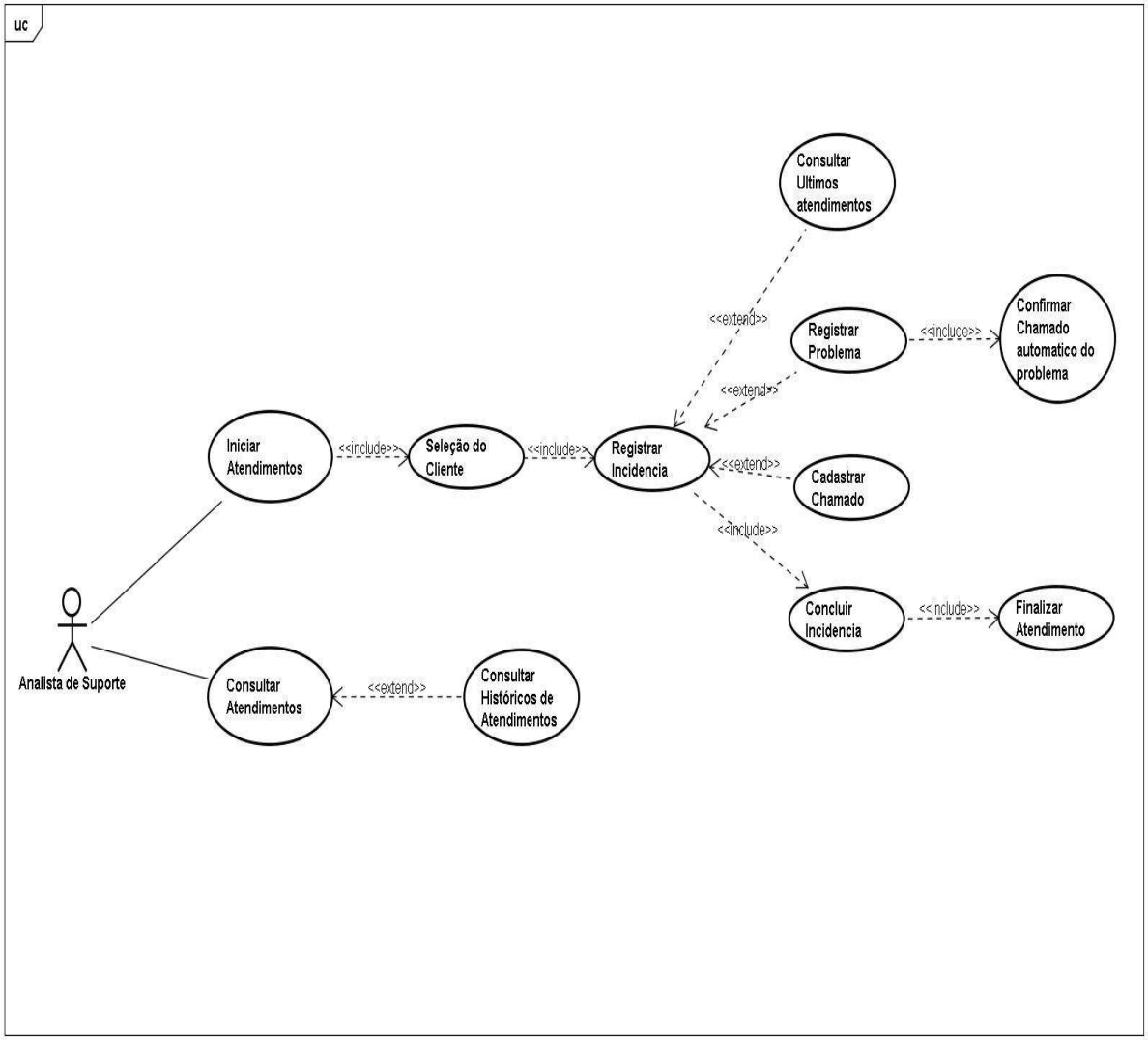


Figura 7 - Apresenta o Fluxo principal de um atendimento ao cliente, ou seja, o primeiro contato do cliente com a central de serviços (Help Desk).

6.1.1.1.1. Narrativa de caso do caso de uso: Atendimento

Nome do Caso de Uso	**Fluxo Principal - Iniciar Atendimentos. N**
Ator Principal	Analista de Suporte técnico
Resumo	Este Caso de Uso descreve as etapas de um fluxo comum de atendimento, podendo dele gerar problemas ou também chamados
Pré-Condições	Colaborador Cadastrado, Cliente cadastrado
Pós-Condições	
Fluxo Principal	
Ações do Ator	**Ações do Sistema**
1.Atender um Telefonema	
	2. Exibir a tela inicial de Atendimento
3. Informar Cliente	
	4. Buscar Cliente por CNPJ/ Razão Social/Série/Fantasia
5. Analisar Incidência	
	6. Registrar Incidência
7. Finalizar Atendimento	
Fluxo de Alternativas	
5. Caso o analista precise de um auxilio na tomada de decisões, poderá Executar o Caso de Uso de Problemas	5. Caso não for possível sanar o problema, executar caso de uso Abrir Chamado
Fluxo de Exceções	
4. Cliente não cadastrado, Gerar pré-cadastro de Clientes. Código do erro: 001	

Narrativa do caso de uso da figura 7.

6.1.1.2 caso de uso: gerenciamento de chamados

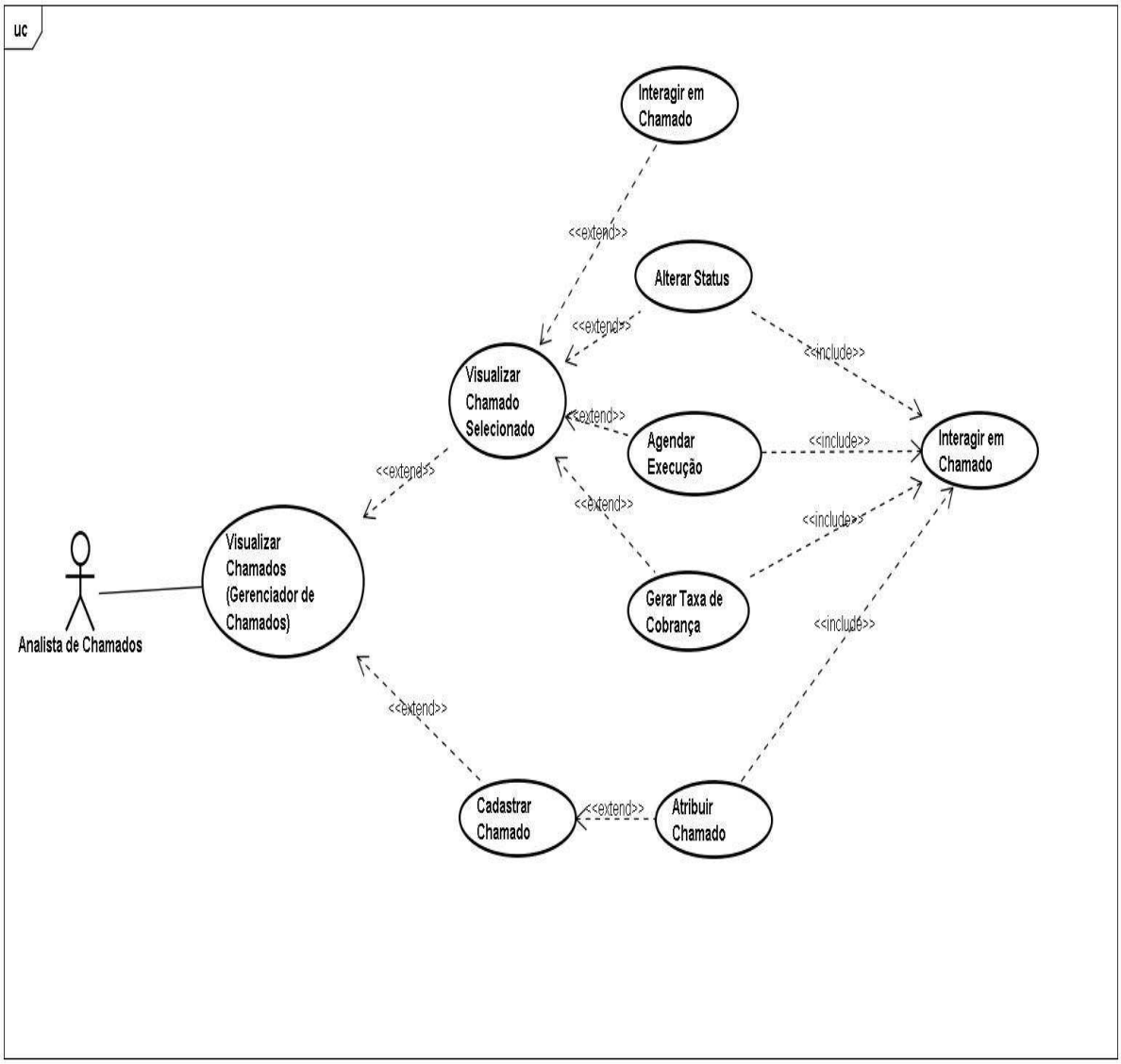


Figura 8 – Gerenciamento de Chamados. Esta ferramenta dará o rumo ao atendimento não solucionado, ou também a solicitações de clientes.

6.1.1.2.1 Narrativa de caso de uso

Colunas1	Colunas2
Nome do Caso de Uso	**Fluxo Principal - Acesso ao Gerenciador de Chamados**
Ator Principal	Analista de Suporte técnico
Resumo	Com base nas fundamentações da ITIL, se por acaso o cliente ficar sem resolução do seu atendimento no primeiro nível, deverá ser aberto um chamado para que outra equipe(com mais esperiencia se for o caso) entre em contato com uma melhor posição sobre o incidente.
Pré-Condições	Ter um Cliente para a abertura do chamado
Pós-Condições	
Fluxo Principal	
Ações do Ator	**Ações do Sistema**
1. Acessar o Gerenciador de Chamados	
	2.Exibição da tela de Gerenciamento de Chamados
3. Cadastrar um chamado	
	4.exibir a tela de cadastro de problemas, Solicita o preenchimento das informações e solicita a atribuição do chamado para o setor de destino.
5. Realizar interação em chamados	
	6. Recebe a mensagem de interação e atualiza o chamado
7.Agendar a execução de um Chamado (Somente para Executores)	
	8.Exibir agenda de chamados (Somente para executores)
9. Adicionar taxa de Execução do Procedimento	
	10.Vincular taxa ao chamado, gerar cobrança. Sistema exibe tela de interação

Fluxo de Alternativas	
3. Cancelar a abertura do Chamado	Ao cancelar sistema retorna ao passo 1.
5. Cancelar a interação	Ao cancelar sistema retorna ao passo 3, sem a confirmação do chamado.
7. Cancelar o Agendamento de Execução	Sistema fecha a agenda e retorna ao passo 1.
9. Cancelar a inserção da taxa	Sistema fecha a tela de inserção e inicia o passo 1.
Fluxo de Exceções	
4. Falha ao registrar salvar Chamado "Dados inválidos ou não preenchidos"	Código do erro: 003
6.Não foi enviada nenhuma mensagem "Chamado não pode ser Alterado sem antes ter uma interação"	Código do erro: 004
8.Colaborador não cadastrado para acessar agenda.	Código do erro: 005
10.Colaborador não autorizado a realizar o procedimento "Acesso Negado"	Código do erro: 006

Narrativa do caso de uso 02 – Gestão de Chamados.

6.1.1.3. caso de uso: gerenciamento de problemas (analista de suporte)

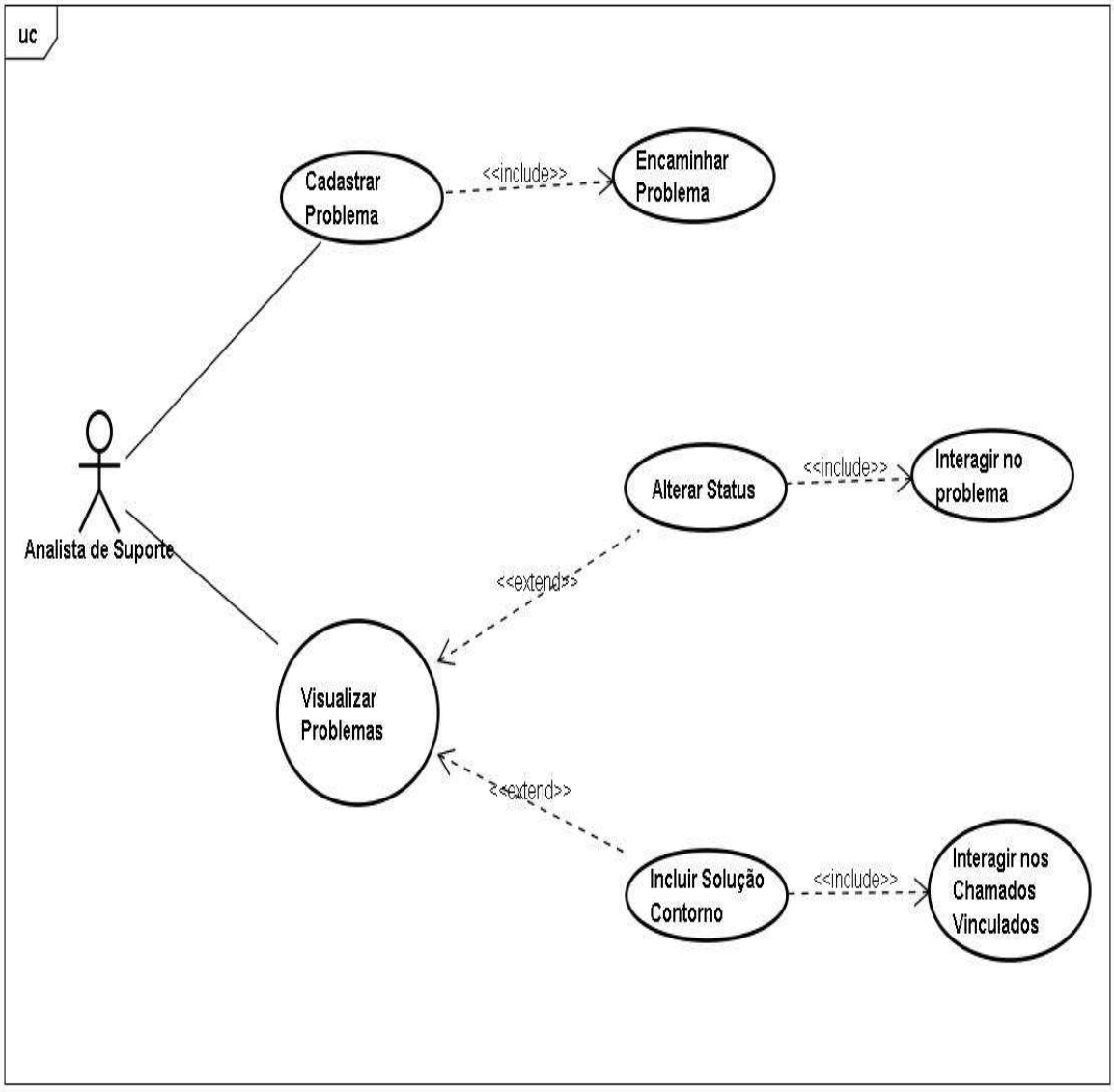


Figura 9 – Gestão de problemas. Este módulo dará ênfase em tarefas e processos para melhorias do sistema, alimentado pelos problemas reportados por clientes e até mesmo colaboradores.

6.1.1.3.1 narrativa do caso de uso

Colunas1	Colunas2
Nome do Caso de Uso	**Fluxo Principal - Acesso ao Gerenciador de Problemas**
Ator Principal	Analista de Suporte técnico
Resumo	Com base no problema reportado pelo cliente, o analista terá uma fonte de informações geradas por problemas de outros atendimentos no qual o cliente pode se enquadrar e aplicar suas soluções sendo elas, definitivas e/ou de contorno.
Pré-Condições	Permissão de Acesso ao módulo.
Pós-Condições	
Fluxo Principal	
Ações do Ator	**Ações do Sistema**
1. Acessar o Gerenciador de Problemas	
	2.Exibição da tela de Gerenciamento de problemas
3.Cadastrar problema	
	4.exibir a tela de cadastro de problemas, Solicita o encaminhamento do mesmo
5. Alterar do Status do problema	
	6. Solicitar a interação referente a alteração
7.Definir uma solução contorno	
	8.Solicitação de senha para inclusão de Solução contorno.
Fluxo de Alternativas	
3. Cancelar a abertura do problema	Ao cancelar sistema retorna ao passo 2.
5. Cancelar a alteração do Status	Ao cancelar sistema retorna ao passo 4.

Fluxo de Exceções	
2. Acesso Negado ! Colaborador não tem permissão para acessar o gerenciador	Código do erro: 002

6.1.1.4 caso de uso: gerenciamento de clientes

O gerenciador de Clientes fará o cadastro de todos os clientes de uma empresa já em funcionamento com uma rotina de leitura de arquivo de dados, para que uma empresa que tem muitos clientes não precisem realizar todos os cadastros de seus clientes. Será possível também o cadastro, alteração, leitura dos clientes.

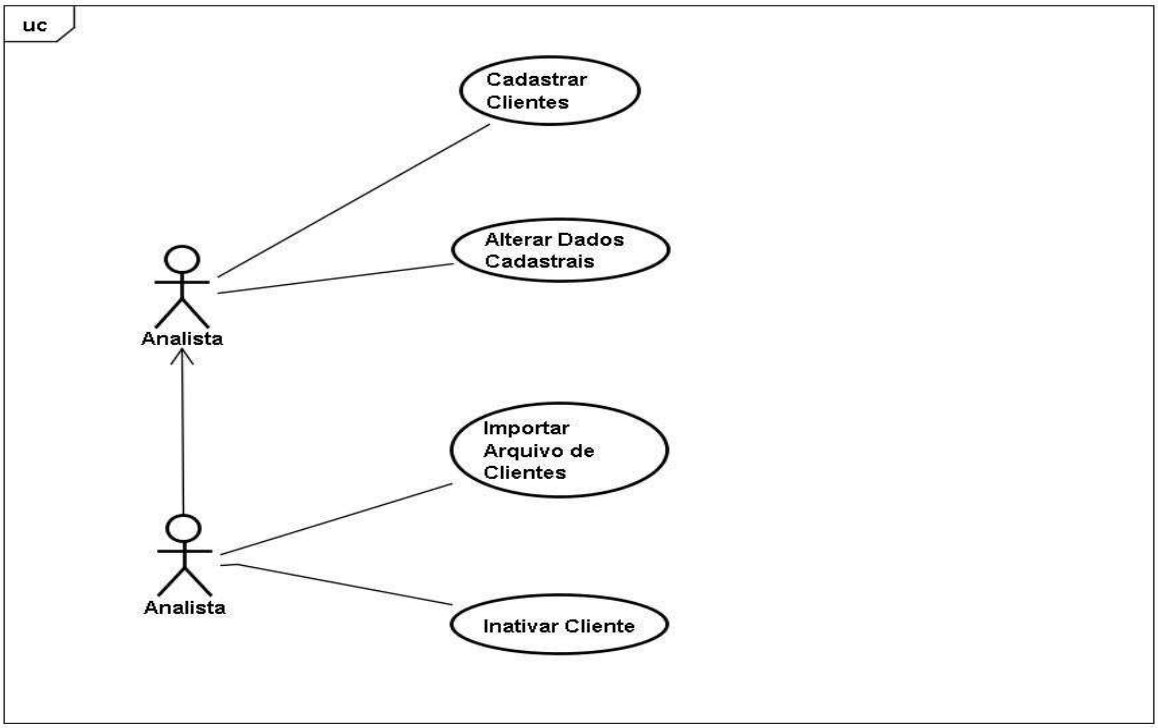


Figura 10 – Gestão de clientes pelo SDS

6.1.1.4.1. narrativa do caso de uso: gerenciador de clientes

Colunas1	Colunas2
Nome do Caso de Uso	**Fluxo Principal - Gerenciador de Clientes**
Ator Principal	Gerente de negócios, Administrador de Negócios
Resumo	Este Caso de Uso descreve as etapas de um cadastro de clientes, o sistema possui dois mecanismos para este procedimento, Cadastro manual e leitura de arquivo para importação.
Pré-Condições	Arquivo com Layout padrão, permissão para acessar o gerenciador de Clientes.
Pós-Condições	
Fluxo Principal	
Ações do Ator (Gerente)	**Ações do Sistema**
1.Acessar Gerenciador de Clientes	
	2. Valida acesso, bloqueia importação de clientes.
3. Manter Clientes	
	4. Exibir cadastro manual de Clientes
Ações do Ator (Administrador)	**Ações do Sistema**
5. Acessar o Gerenciador de clientes	
	6. Valida acesso e libera todas as opções de acesso do gerenciador
7. Importar Arquivo de Clientes	
	8. Validação do layout do arquivo, importação dos clientes
Fluxo de Alternativas	
7. Cancelar importação de dados dos clientes	.

Fluxo de Exceções	
3. Cliente já Cadastrado	Código do erro: 007
8. Layout não possui as especificações de importação	Código do erro: 008

6.1.1.5. caso de uso: gerenciador de tarefas

Gestão de Tarefas. Como dito no decorrer do projeto, problemas podem se tornar tarefas para outros departamentos. O gerenciador de Tarefas será administrado por um analista que fará o encaminhamento das inconsistências reportadas, e análise das mesmas. Pode ser usado também por analistas de testes, para registrar problemas em versões e etc.

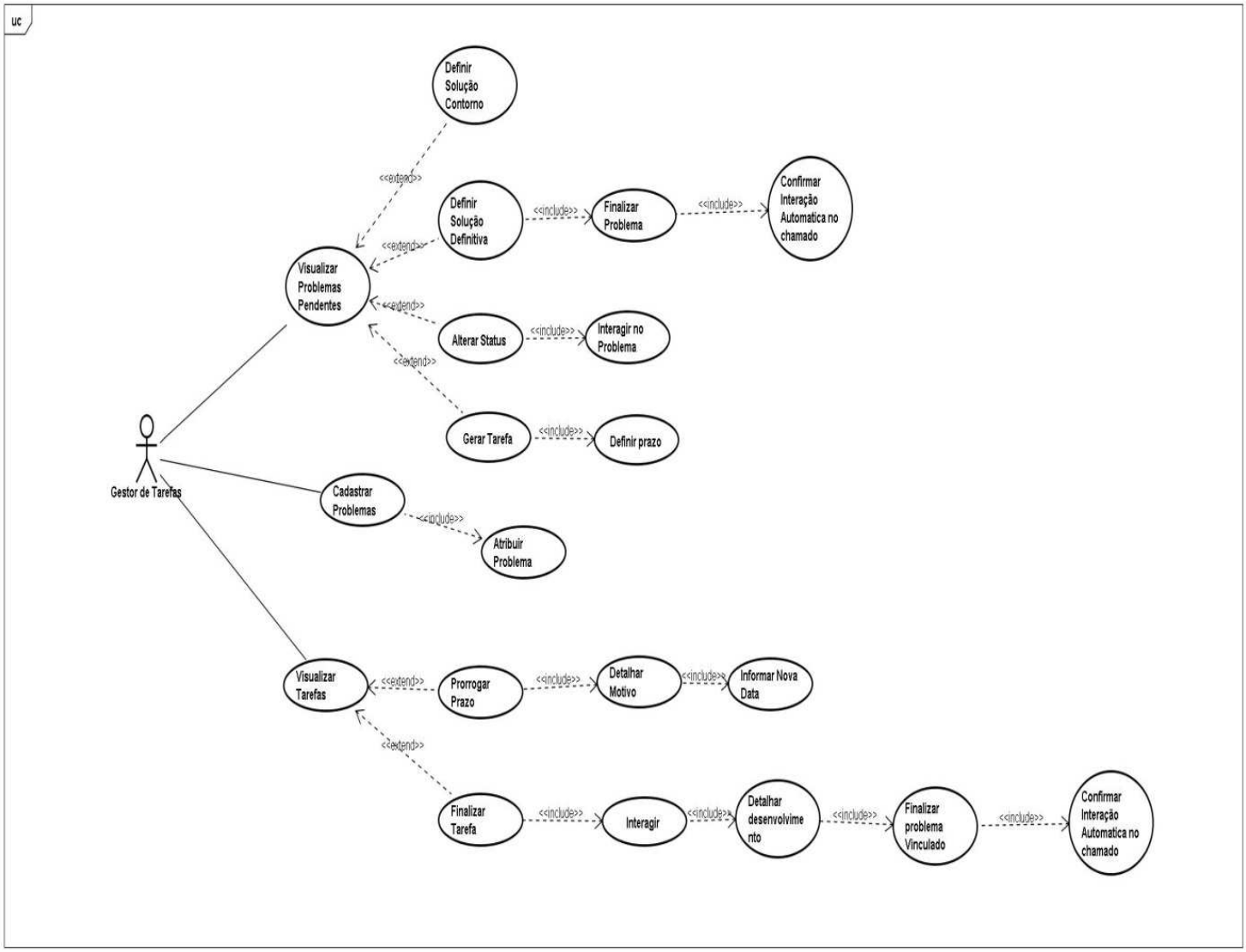


Figura 11 – Gerenciamento de tarefas

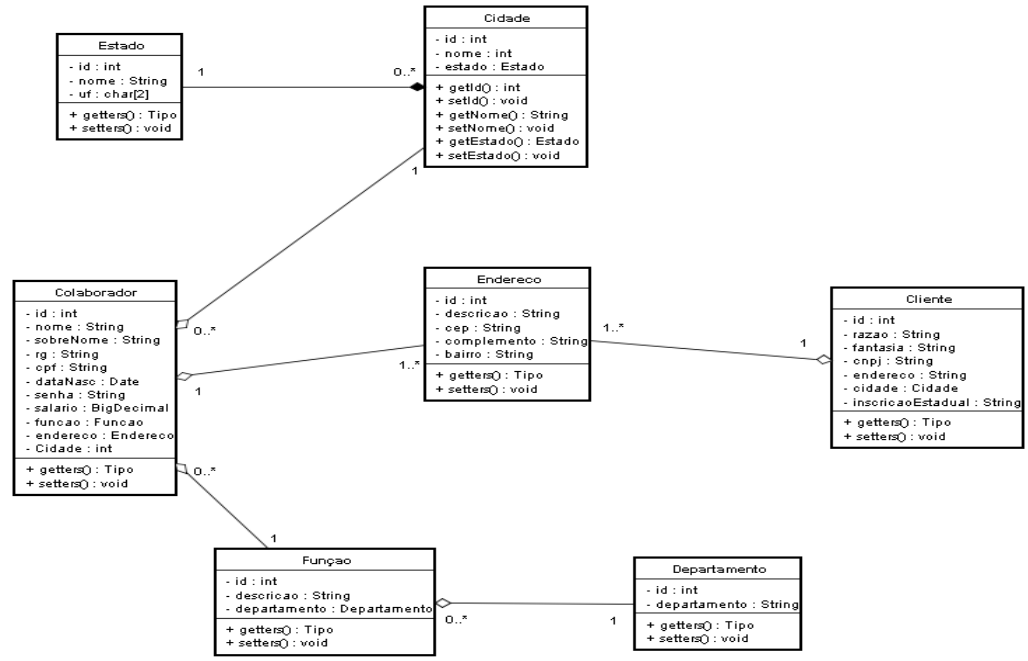
6.1.2 diagrama de classes

De acordo com Guedes(2011, p. 30):

> "O diagrama de classes e provavelmente o mais utilizado e é um dos mais importantes da UML. Serve de apoio para a maioria dos demais diagramas. Como o próprio nome diz, define a estrutura das classes utilizadas pelo sistema, determinando os atributos e métodos que cada classe tem, além de estabelecer como as classes se relacionam entre si."

O Diagrama de classes mostra como as classes irão se comportar em seus relacionamentos. Pode ser usado para a base de um diagrama E.R e também para a relação das tabelas de um banco de dados.

De acordo com a descrição de diagramas de classes, foram elaborados dois diagramas que ilustram todas as classes do sistema, quais as relações entre elas e todas suas características.



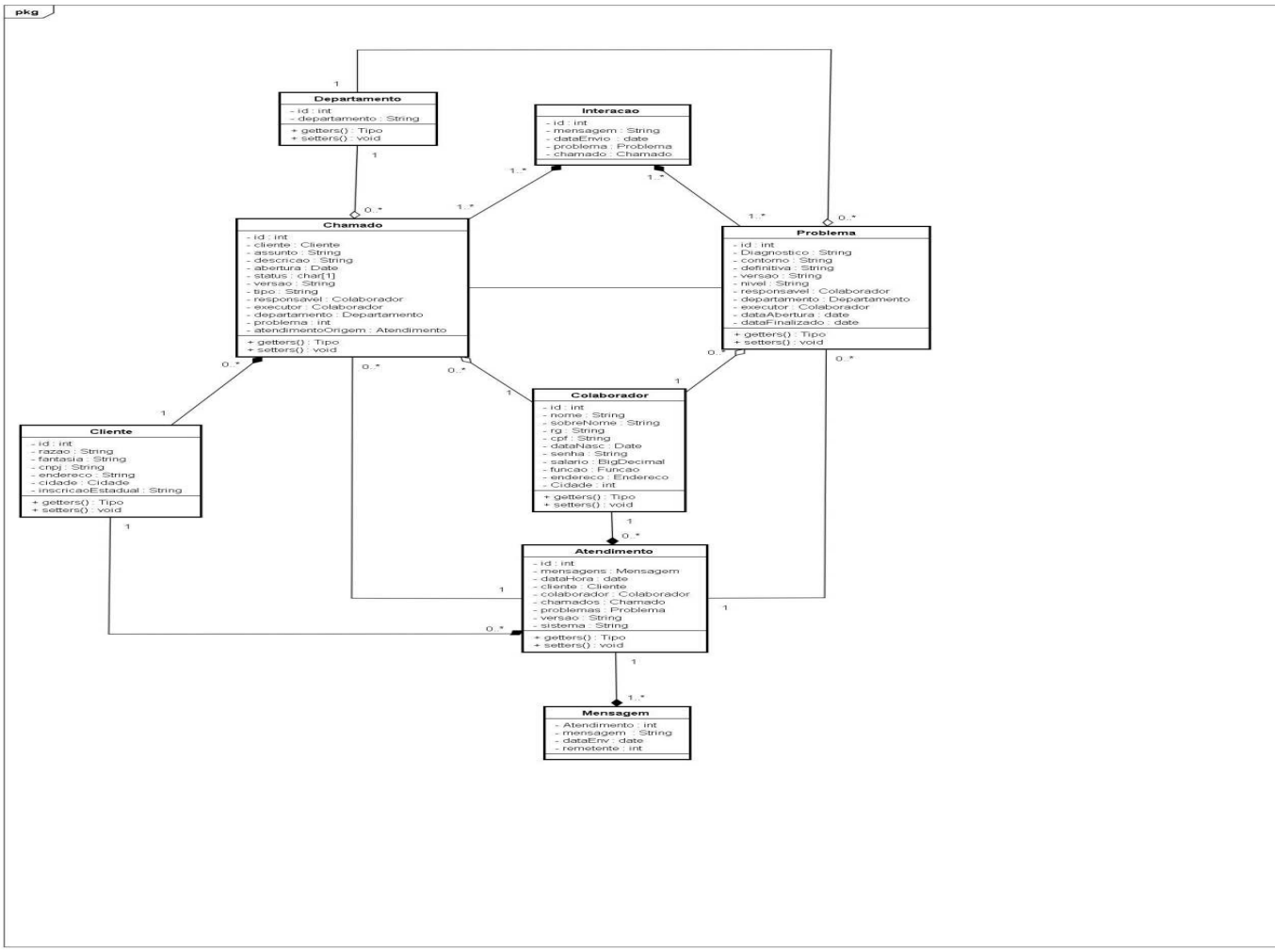


Figura 12 – Diagrama de Classes

6.1.3 diagrama de atividades

Para Guedes (2011, p. 30):

> “O Diagrama de Atividade preocupa-se em descrever os passos a serem percorridos para a conclusão de uma atividade específica, podendo estar ser representada por um método com certo grau de complexidade, um algoritmo, ou mesmo por um processo completo. O diagrama de atividades concentra-se na representação do fluxo de controle de uma atividade”.

O diagrama de atividades, diferente do caso de uso, detalha todas as atividades passos a passo, com todas as intersecções que podem ocorrer entre uma atividade e outra, direcionando o diagrama de acordo com cada atividade realizada. É um diagrama bem completo, detalhista, pode servir como base para muitas outras rotinas, inclusive na programação em si, como rotinas de implementação e etc.

Abaixo veremos algumas rotinas do sistema através do diagrama de atividades.

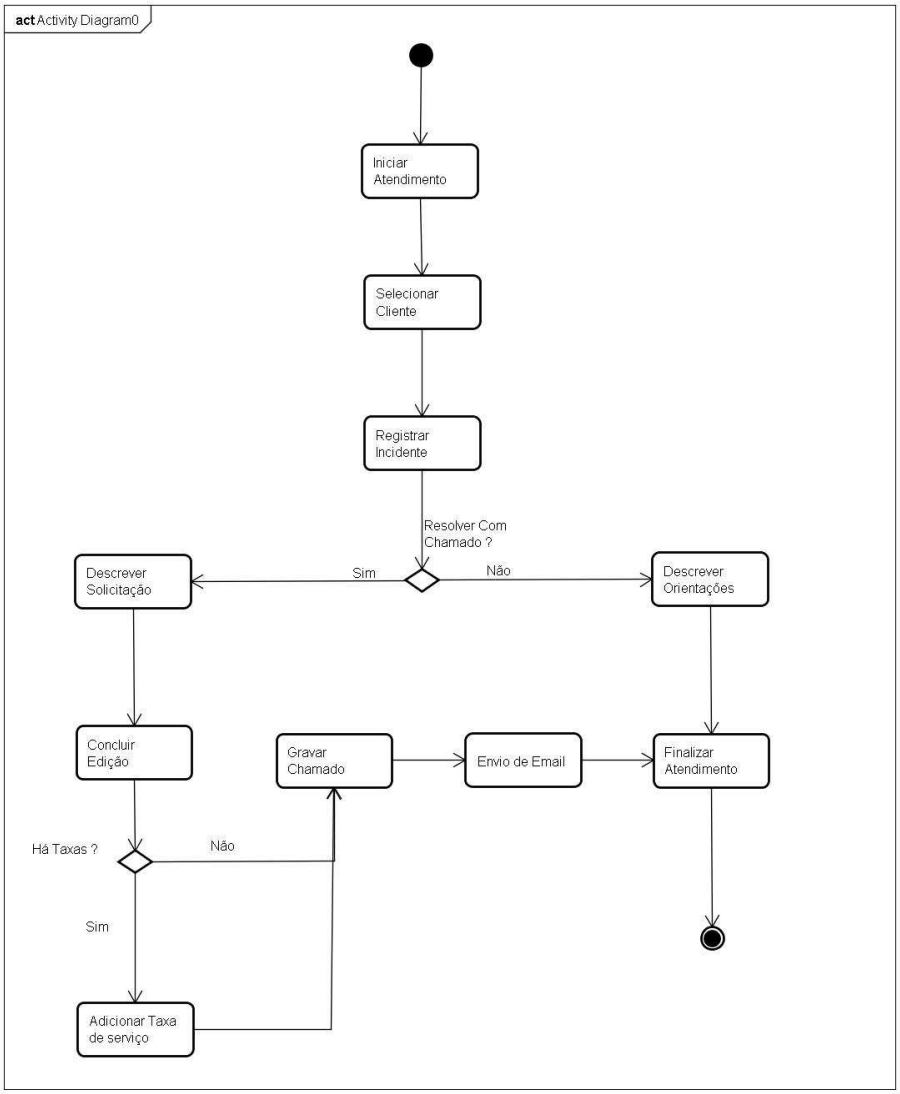


Figura 13 – Diagrama de atividades, fluxo atendimento/solicitação.

Figura 13 – Fluxo de atendimento com abertura de Chamado. Um atendimento pode ser iniciado e finalizado sem ter problemas, ou chamados, pode se tratar apenas de dúvidas do cliente em questão. Porém, um atendimento pode gerar uma ordem de serviço. O chamado é uma ordem de serviço, por se tratar de um procedimento que vai além das atividades prestadas pelo Help Desk.

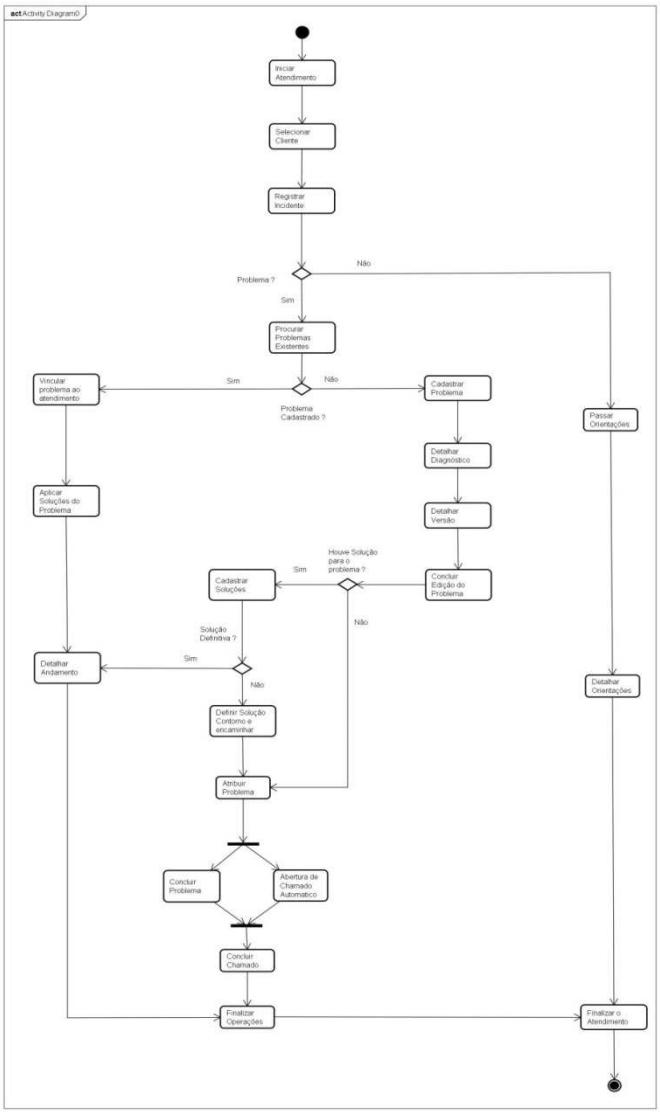


Figura 14 – Fluxo de atividades Atendimento/Problema.

Figura 14 – Esta imagem, mostra todo o fluxo de um atendimento que gerou uma inconsistência (problema). A Figura detalha com todas as atividades como é a rotina de um atendimento com o vinculo de um problema. Mostra as tarefas de cadastro e também como o sistema se comportará quando for aberto um problema.

Podemos analisar também, que ao registrar um problema, o sistema cria um chamado automático para que o cliente tenha todo o acompanhamento da inconsistência e saiba quando e como foi feita a correção da mesma.

6.1.4 – modelagem entidade e relacionamento – mer

O diagrama de entidade e relacionamento é o mais utilizado e um dos mais importantes diagramas na modelagem de um sistema. Neste diagrama são definidos todas as tabelas de banco de dados que serão utilizadas no software, mostra a relação das tabelas e informações das mesmas, afim de, garantir uma visão detalhista da estrutura onde ficarão armazenados todos os dados do sistema. Baseia-se também, assim como a orientação a objetos, no mundo real que consiste em uma coleção de objetivos básicos, chamados de entidades e relacionamentos. Não é necessário ter um sistema para criação do diagrama, pode ser feito a punho, porém, existem muitos softwares que auxiliam na criação.

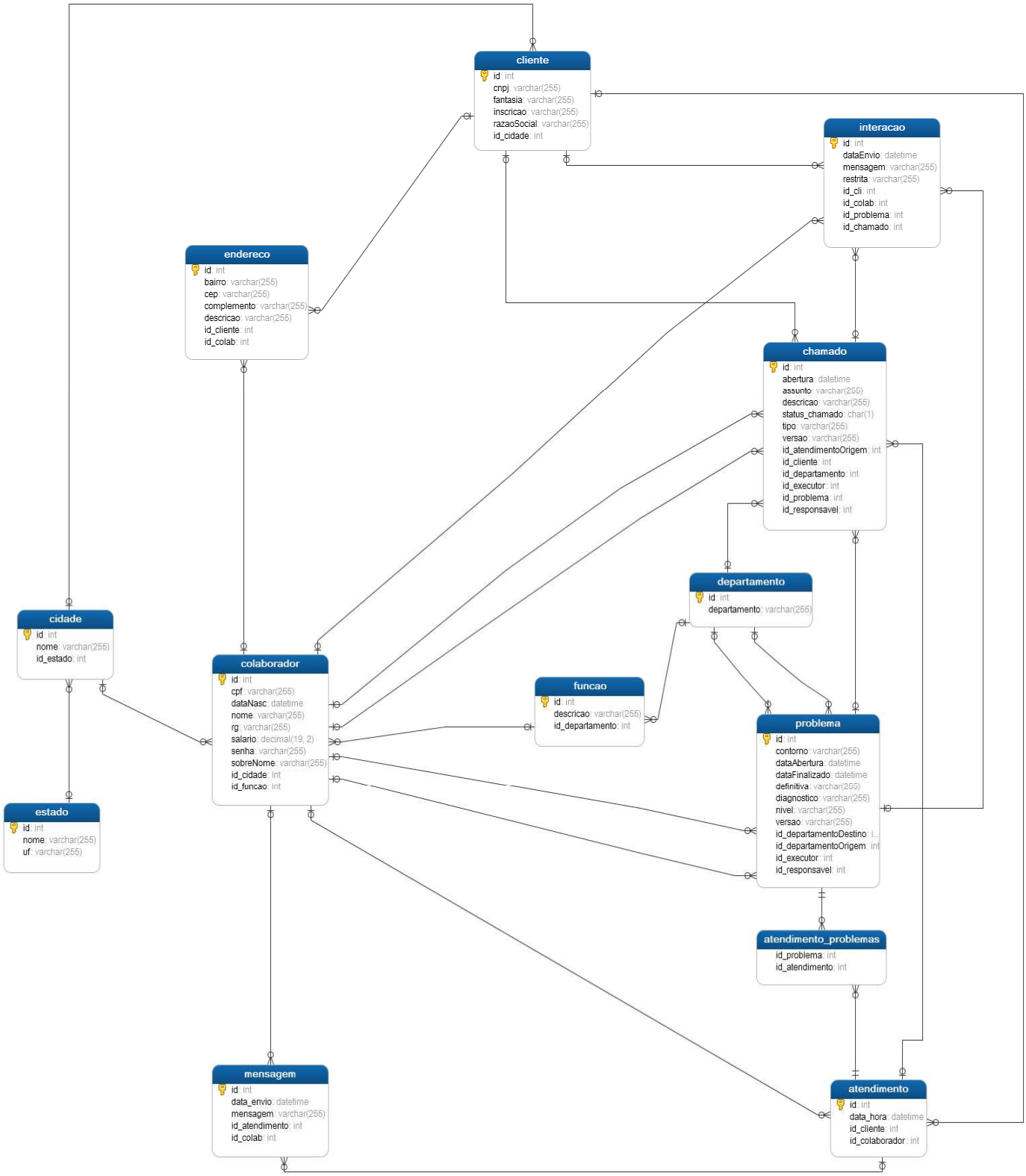


Figura 15 – Diagrama de Entidade e relacionamento.

7. CONCLUSÃO

7.1 PROJETO

O sistema foi projetado para dar um controle geral da área de suporte de empresas que prestam serviços a clientes, e que necessitam de um respaldo tecnológico de eficiência para o auxilio neste procedimento.

A gestão do suporte é de total relevância para a imagem da empresa, pois um suporte eficiente garante total satisfação dos clientes. Porém, para que haja eficiência, é necessário que os analistas tenham o máximo de informações possíveis para se comunicarem com o cliente, prestando assim um suporte de excelência.

Por estes motivos o sistema garantirá o máximo de informações como, problemas conhecidos, gestão de incidentes, acordos de níveis de serviços, gerenciador de solicitações (chamados) para que os clientes tenham espaço para colaborar no crescimento da empresa. O sistema terá um leque de opções para que o analista preste o suporte o mais rápido possível com eficiência.

Além disso, o sistema prestará também, serviços a outros setores, como por exemplo, o setor de desenvolvimento de uma empresa de T.I, gerando a partir dos registros de incidentes e vínculos de problemas a atendimentos, questões a serem analisadas pelo departamento responsável pela manutenção destes levantamentos. O sistema gerará relatórios de rendimento de colaboradores, índices de atendimentos, assuntos mais incidentes, problemas mais vinculados, numero de chamados abertos, para que o gestor de suporte, tenha um total controle de sua equipe.

7.2 PROJETOS FUTUROS

O SDS será um módulo de um sistema maior a ser desenvolvido. O sistema terá integrado todos os recursos de uma empresa, desde rotas de colaboradores externos, até questões administrativas e estratégicas. Ele fará gestão dos três níveis de uma empresa de T.I, que são: produção, tática e estratégia. O SDS é voltado para a produção, pois visa o atendimento direto ao cliente, porém por ele pode ser levantadas muitas questões táticas e estratégicas.

O novo projeto será construído sobre os padrões ERP, que será um assunto estudado futuramente.

REFERÊNCIAS

SODRÉ, M. G.; SOUZA, S. M. **Uma análise comparativa de Metodologias para Governança de Tecnologia da Informação – ITIL e COBIT.** Monografia (Bacharel em Ciência da Computação) Universidade Federal de Santa Catarina. Florianópolis, 2007.

LIMA, L. C. S. Estudo **do modelo de Gestão ITIL e um comparativo com o modelo COBIT**. Monografia (Graduação em Sistemas de Informação) Universidade Estadual de Montes Claros. Montes Claros, 2007.

MORAES, E. A. P.; MARIANO, S. R. H. **Uma Revisão dos Modelos de Gestão Em TI.** VCNEG (Congresso Nacional de Excelência em Gestão), Niterói. 2008

ITGI, Cobit 4.1. Disponível em <www.itgi.org.br> acesso em 10 Setembro 2013.

DEITEL, H. M.; DEITEL, P. J. Java: como programar. Editora Bookman. 6ª ed. São Paulo: 2005.

GOLÇALVES, Edson. **Desenvolvendo aplicações WEB com NetBeans IDE 6,** Rio de Janeiro, Editora Ciência Moderna Ltda, 2008.

AMBLER, Scott W. **Análise e Projeto orientados a Objetos.** Trad. Cláudio Costa, Rio de Janeiro, Infobook, 1997.

GUEDES, Gilleanes T. A. **UML 2: uma abordagem prática,** 2º Ed. São Paulo: Novatec Editora, 2011.

BAUER, Christian e King, Gavin. **Java persistence com Hibernate.** Rio de Janeiro: Editora Ciência moderna Ltda,2007.

MILANI, André. **MySQL Guia do Programador.** São Paulo: Novatec Editora, 2006