**UNIVERSIDADE PAULISTA**

**ALEXANDRE CAPI RAMPANI**

**CLEITON FERNANDO DOS SANTOS**

**GUILHERME REVOREDO**

**JOÃO HENRIQUE LEVADA**

**LETÍCIA CRISTINA SOARES**

**PROJETO INTEGRADO MULTIDISCIPLINAR (4º SEMESTRE) – CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**ARARAQUARA**

**2018**

**ALEXANDRE CAPI RAMPANI**

**CLEITON FERNANDO**

**GUILHERME REVOREDO**

**JOÃO HENRIQUE LEVADA**

**LETÍCIA CRISTINA SOARES**

**PROJETO INTEGRADO MULTIDISCIPLINAR (4º SEMESTRE) – CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

Trabalho de conclusão do quarto semestre do curso de Análise e desenvolvimento de sistemas apresentado à UNIP.

Orientadora: Prof. Kamila Rios

**ARARAQUARA**

**2018**

*[epígrafe]*

**Resumo**

O objetivo principal deste trabalho é apresentar os detalhes de implementação e planejamento que levaram à construção de um sistema de helpdesk para empresas prestadoras de serviços na área de manutenção de computadores.

Ao longo do documento, portanto, serão apresentadas as camadas MVC (model, view e controller) que compõem o software, bem como todo o seu processo de desenvolvimento – desde o levantamento de requisitos até a implementação - o foi orquestrado por técnicas de gestão de projetos orientadas pelo PMBOK.

Por fim, será desenvolvido um estudo acerca da criação e manutenção de uma startup, modelo de empresa com o qual o sistema em questão se integraria perfeitamente, dada sua efetividade e a flexibilidade daquele modelo empresarial.

**Abstract**

The main objective of this work is to present the implementation and planning details that led to the construction of a helpdesk system for companies that provide services in the area of computer maintenance.

Throughout the document, therefore, the MVC (model, view and controller) layers that make up the software, as well as its entire development process - from requirements gathering to implementation - will be presented. It has been orchestrated by management techniques projects guided by the PMBOK.

Finally, a study will be developed on the creation and maintenance of a startup, a business model with which the system in question would integrate perfectly, given its effectiveness and the flexibility of that business model.

Sumário

[Lista de Figuras 7](#__RefHeading___Toc138_1997538900)

[Lista de Tabelas 8](#__RefHeading___Toc2719_1355530280)

[1.INTRODUÇÃO 9](#__RefHeading___Toc2721_1355530280)

[2.PROJETO DE SISTEMAS ORIENTADO A OBJETO 11](#__RefHeading___Toc83_1187944198)

[2.1 Diagrama de casos de uso 11](#__RefHeading___Toc85_1187944198)

[2.2. Diagrama de classes 12](#__RefHeading___Toc87_1187944198)

[2.3.Diagramas de comunicação 13](#__RefHeading___Toc89_1187944198)

[2.4.Diagramas de sequência 13](#__RefHeading___Toc393_1713782431)

[Os diagramas de sequência foram construídos para dar apoio na visualização da atividade-fim do sistema P4, que é o gerenciamento de chamados. Por meio desse tipo de diagrama UML, foi possível visualizar a maneira como os fluxos de abrir, atualizar e encerrar chamados organizam-se temporalmente no sistema, conforme pode ser constatado a partir do apêndice I. 13](#__RefHeading___Toc395_1713782431)

[2.5.Diagramas de pacotes 13](#__RefHeading___Toc397_1713782431)

[Apêndice A – Diagrama de casos de uso 15](#__RefHeading___Toc372_1713782431)

[Apêndice B – Diagrama de classes 16](#__RefHeading___Toc374_1713782431)

[ Apêndice C - Diagrama de comunicação (Abrir chamado) 17](#__RefHeading___Toc376_1713782431)

[ Apêndice D - Diagrama de comunicação (Atualizar chamado) 18](#__RefHeading___Toc378_1713782431)

[ Apêndice E - Diagrama de comunicação (Atualizar atendente) 19](#__RefHeading___Toc380_1713782431)

[ Apêndice F - Diagrama de comunicação (Criar atendente) 20](#__RefHeading___Toc382_1713782431)

[ Apêndice G - Diagrama de comunicação (Criar cliente) 21](#__RefHeading___Toc384_1713782431)

[ Apêndice H - Diagrama de comunicação (Atualizar cliente) 22](#__RefHeading___Toc386_1713782431)

[ Apêndice I - Diagrama de sequência (Abrir chamado) 23](#__RefHeading___Toc405_1713782431)

[ Apêndice J - Diagrama de sequência (Atualizar chamado) 24](#__RefHeading___Toc407_1713782431)

[ Apêndice K - Diagrama de sequência (Encerrar chamado) 25](#__RefHeading___Toc409_1713782431)

# **Lista de Figuras**

# Lista de Tabelas

# 1.INTRODUÇÃO

1. Com o desenvolvimento e popularização da internet em meados dos anos 2000 em desktops e agora, na década de 2010, em smartphones, a maneira como o software é projetado e implementado vem mudando de maneira bastante substancial. A ideia de que um sistema computacional é representado por tudo aquilo que está na máquina do usuário – bibliotecas e interface - caiu por terra com a adoção de novos modelos arquiteturais no desenvolvimento de produtos de software, que exploram com mais eficácia a infraestrutura proporcionada pelo protocolo HTTP e pela estabilidade cada vez maior das conexões à rede mundial de computadores.
2. Neste sentido, o sistema P4, que será apresentado ao longo deste trabalho, foi alicerçado sobre essas premissas arquiteturais e de desenvolvimento web. Serão pormenorizados, portanto, os detalhes das tecnologias empregadas em sua construção e a maneira como foram implementadas a fim de que o sistema tenha a qualidade esperada pelos clientes, ou seja, com escopo bem definido e fiel aos requisitos levantados no início do projeto, bem como com tempo e custo coerentes com o estipulado.
3. O modelo arquitetural escolhido para sustentar o projeto foi o MVC (model, view, controller), graças à sua capacidade de dividir responsabilidades dentro da comunicação entre as partes do sistema e garantir que haja redução da complexidade de sua manutenção. A concretização desse padrão arquitetural se deu por meio do REST, modelo cujo princípio é extrair todo o potencial do protocolo HTTP.
4. Dentro desse contexto, a linguagem utilizada para desenvolvimento backend da API responsável pela comunicação entre os módulos do sistema foi o Java, devido à sua flexibilidade já reconhecida no desenvolvimento web, quando em associação com frameworks voltados também para desenvolvimento web. No caso, optou-se pelo framework Spring, cujo módulo MVC dá respaldo a projetos de pequeno porte, mas também é respeitado no contexto de desenvolvimento de aplicações Java Enterprise.
5. A fim de garantir um desenvolvimento consciente do software e com retrabalho reduzido, utilizou-se o arcabouço teórico proposto pela engenharia de software no que diz respeito o projeto de sistemas orientado a objetivo. Para isso, a diagramação UML foi adotada com o objetivo de balizar a implementação em todos os seus aspectos.
6. Por fim, e com o objetivo de promover a organicidade e coesão durante o desenvolvimento, foram adotadas técnicas de gestão de projeto para manter o escopo sob controle da equipe. Algumas premissas do guia PMBOK que se mostraram adequadas ao baseline do projeto foram adotadas, bem como ferramentas e filosofias de desenvolvimento ágil, garantindo entregas concretas e periódicas.

# 2.PROJETO DE SISTEMAS ORIENTADO A OBJETO

1. Um software de qualidade pressupõe sistematização no seu desenvolvimento, e é justamente a fim de garantir essa solidez que a engenharia de software e outras técnicas de desenvolvimento são empregadas na construção de sistemas computacionais. Dentro desse ferramental fornecido aos desenvolvedores e projetistas, pode-se destacar a orientação a objetos como um importante filão da engenharia de software.
2. Conceitos como os de herança, classes e objetos foram adotados pelos engenheiros de software para que a concepção do sistema como um todo pudesse acompanhar as técnicas de orientação a objetos, cujo objetivo é aproximar o software do mundo real. Nascem daí as noções de análise de sistemas orientada a objeto e projeto de sistemas orientado a objeto, com a função de garantir aos envolvidos no projeto uma base sólida para sua evolução.
3. Por análise de sistemas orientada a objeto, compreende-se o processo por meio do qual são identificados os pontos importantes do software, com base nos dados colhidos na fase inicial do projeto, como na fase de levantamento de requisitos, por exemplo. Essa informação construída pela etapa de análise é imprescindível para consolidação da próxima etapa, escopo deste capítulo: o projeto de sistemas orientado a objeto.
4. Se a etapa de análise representa o passo em que se chega às informações importantes a nível técnico sobre o sistema, transcendendo o levantamento de requisitos e as informações de alto nível, o projeto de análise compreende o momento a partir do qual são traçadas estratégias para interligar as informações obtidas anteriormente, com a finalidade de se chegar ao resultado esperado pelo cliente. Pode-se resumir os dois processos sucintamente: enquanto um faz o levantamento de problemas, o outro procura valer-se de meios para solucioná-los.

## 2.1 Diagrama de casos de uso

* 1. Dentre os diagramas à disposição da equipe de desenvolvimento, o diagramas de casos de uso é o que representa o sistema sob a perspectiva do usuário, portanto é um diagrama com grande capacidade de abstração (afinal, é necessário que seja didático para transmitir as funcionalidades sob a perspectiva do usuário) e sem intenções de transmitir detalhes técnicos a respeito da implementação ou arquitetura. Para o sistema P4, os seguintes valores estão sendo assumidos pelos elementos do diagrama:
  2. Ator: atendente do helpdesk.
  3. Casos de uso:
  4. Cadastrar atendente: o atendente pode cadastrar outro atendente a fim de corresponder à demanda por novos funcionários na empresa.
  5. Fazer login: o atendente tem uma tela de login para fazer sua autenticação no sistema.
  6. Cadastrar cliente: o atendente pode cadastrar novos clientes da empresa.
  7. Abrir chamado: o atendente pode abrir um chamado para responder à demanda do negócio da empresa.
  8. Visualizar chamado: o atendente pode visualizar na tela um chamado com todos os seus atributos.
  9. Atualizar chamado: o atendente pode acrescentar informações sobre o chamado.
  10. Transferir chamado: o atendente pode transferir o chamado para outro atendente.
  11. Encerrar chamado: o atendente pode fechar um chamado, impossibilitando-o de receber novas atualizações.
  12. Buscar chamado: o atendente pode pesquisar por chamados de acordo com os seguintes critérios:

I)Buscar chamado por protocolo: o atendente pode pesquisar por um chamado utilizando o número de protocolo dele.

II)Buscar chamado por cliente: o atendente pode pesquisar o chamado pelo nome do cliente.

O diagrama de casos de uso está referenciado no Apêndice A – Diagrama de casos de uso, no qual é possível verificar com maior clareza as funcionalidades à disposição do usuário do sistema em questão.

## 2.2. Diagrama de classes

* 1. Os diagramas de classes fornecem uma representação do sistema com um nível de detalhamento técnico mais apurado, envolvendo as classes que são implementadas no sistema. No caso do sistema P4, alguns métodos não são exibidos neste diagrama em função da arquitetura e do padrão de projeto utilizados. Uma vez que o modelo MVC segmenta o sistema em camadas, grande parte da comunicação entre o ator e os dados é intermediada pelos controllers.
  2. Neste sentido, o nível de abstração do diagrama de classes é alto, acompanhando a filosofia do MVC, REST e Spring adotados no projeto: reduzir a complexidade do código e tornar o sistema fluido e didático.
  3. A classe Atendente, por exemplo, não chama nenhum método porque esta responsabilidade fica a cargo dos controllers. O código abaixo ilustra essa divisão de responsabilidades, na medida em que demonstra a criação de um novo atendente pelo controller da classe atendente (AtendenteRestController):

|  |
| --- |
| * 1. @PostMapping   2. public ResponseEntity<URI> criar(@RequestBody NovoAtendenteDto novoAtendente) throws DomainException {   3. String novoId;   4. novoId = atendenteAppService.criar(novoAtendente);   5. URI novoAtendenteUri = super.criarUriPorId(novoId);   6. return ResponseEntity.created(novoAtendenteUri).build;   7. } |

## 2.3.Diagramas de comunicação

Os diagramas de comunicação são utilizados para demonstrar a troca de mensagem entre as classes que compõem o sistema. São parecidos com os diagramas de sequência, entretanto, não priorizam a ordem cronológica da troca de mensagem, mas sim as próprias mensagens que são trocadas e as classes envolvidas nessas interações.

Como a arquitetura adotada para no projeto é o MVC, pressupõe-se a troca de mensagens entre classes pertencentes a camadas com responsabilidades diversas dentro da organização deste sistema. Por conseguinte, para suportar o projeto do sistema, foram criados diagramas de chamado para os principais casos de uso, que são: A) abrir chamado, transferir e atualizar chamado (este último inclui também o encerramento de chamado), B) criar e atualizar cliente e C) criar e atualizar atendente. Esses três grupos de mensagens estão separados por seus controllers, services e repositories específicos, conforme se pode verificar a partir do apêndice C

## 2.4.Diagramas de sequência

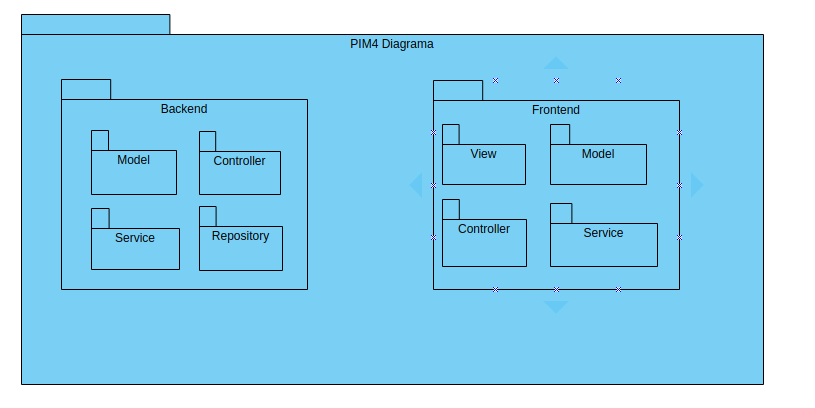
Os diagramas de sequência foram construídos para dar apoio na visualização da atividade-fim do sistema P4, que é o gerenciamento de chamados. Por meio desse tipo de diagrama UML, foi possível visualizar a maneira como os fluxos de abrir, atualizar e encerrar chamados organizam-se temporalmente no sistema, conforme pode ser constatado a partir do apêndice I.

## 2.5.Diagramas de pacotes

Para representar o sistema sob uma perspectiva macro valendo-se da UML, foi utilizado o diagrama de pacotes para esta finalidade. Segundo o site Visual Paradigm, um diagrama de pacotes é conceituado da seguinte maneira:

Package diagram, a kind of structural diagram, shows the arrangement and organization of model elements in middle to large scale project. Package diagram can show both structure and dependencies between sub-systems or modules, showing different views of a system, for example, as multi-layered (aka multi-tiered) application - multi-layered application model.

Novamente, o diagrama de pacotes foi elaborado de maneira a evidenciar a separação de responsabilidades do sistema por meio da arquitetura MVC, conforme é possível visualizar na figura abaixo:



# Apêndice A – Diagrama de casos de uso

# 

# Apêndice B – Diagrama de classes

# 

# Apêndice C - Diagrama de comunicação (Abrir chamado)

# 

# Apêndice D - Diagrama de comunicação (Atualizar chamado)

# 

# Apêndice E - Diagrama de comunicação (Atualizar atendente)

# 

# Apêndice F - Diagrama de comunicação (Criar atendente)

# 

# Apêndice G - Diagrama de comunicação (Criar cliente)

# 

# Apêndice H - Diagrama de comunicação (Atualizar cliente)

# 

# Apêndice I - Diagrama de sequência (Abrir chamado)

# Apêndice J - Diagrama de sequência (Atualizar chamado)

# 

# Apêndice K - Diagrama de sequência (Encerrar chamado)

# 