FACULDADE DE TECNOLOGIA DE AMERICANA

Curso Superior de Tecnologia em Analise e Desenvolvimento de Sistemas

Jorge Miguel Carraro Gattaz Abdalla

SISTEMA DE GERENCIAMENTO E MONITORAMENTO DE ATIVOS DE TI

Americana, SP

2017

FACULDADE DE TECNOLOGIA DE AMERICANA

Curso Superior de Tecnologia em Analise e Desenvolvimento de Sistemas

Jorge Miguel Carraro Gattaz Abdalla

SISTEMA DE GERENCIAMENTO E MONITORAMENTO DE ATIVOS DE TI

Trabalho de Conclusão de Curso desenvolvido em cumprimento à exigência curricular do Curso Superior de Tecnologia em 2017, sob a orientação Prof. Esp. Antônio Alfredo Lacerda.

Área de concentração: Desenvolvimento de Sistemas.

Americana, SP.

2017

FICHA CATALOGRÁFICA – Biblioteca Fatec Americana - CEETEPS
Dados Internacionais de Catalogação-na-fonte

A116s ABDALLA, Jorge Miguel Carraro Gattaz

Sistema de gerenciamento e monitoramento de ativos de TI. / Jorge Miguel Carraro Gattaz Abdalla. – Americana, 2017.

71f.

Monografia (Curso de Tecnologia em Análise e Desenvolvimento de Sistemas) - - Faculdade de Tecnologia de Americana – Centro Estadual de Educação Tecnológica Paula Souza

Orientador: Prof. Esp. Antonio Alfredo Lacerda

1 Sistemas de informação 2. Desenvolvimento de software I. LACERDA, Antonio Alfredo II. Centro Estadual de Educação Tecnológica Paula Souza – Faculdade de Tecnologia de Americana

CDU: 681.518

681.3.05

Jorge Miguel Carraro Gattaz Abdalla

SISTEMA DE GERENCIAMENTO E MONITORAMENTO DE ATIVOS DE TI

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo em 2017 pelo CEETEPS/Faculdade de Tecnologia – FATEC/ Americana.
Área de concentração: Desenvolvimento de Sistemas.

Americana, 13 de dezembro de 2017.

Banca Examinadora:

Antônio Alfredo Lacerda (Presidente)
Prof. Esp.
Fatec Americana

Francisco Carlos Mancin (Membro)
Prof. Me.
Fatec Americana

Eduardo Antonio Vicentini (Membro)
Prof. Me.
Fatec Americana

AGRADECIMENTOS

Em primeiro lugar, à Deus pela vida, esperança, força, capacidade.

Aos meus pais e irmãos, pelo apoio, amor e carinho.

Em especial ao Professor Esp. Antônio Alfredo Lacerda, pela orientação deste trabalho e pelo apoio em todos os anos de estudo.

Aos professores da Fatec Americana, por todo aprendizado, incentivo e amizade.

A todos amigos e companheiros que trilharam este caminho ao meu lado, apoiando-me e desafiando-me a fazer sempre o melhor.

DEDICATÓRIA

À

Deus,

em quem vivo, existo e me movo.

“*Mas tudo deve ser feito com decência e ordem*”.

1 Coríntios 14:40.

RESUMO

O presente trabalho apresenta as etapas do desenvolvimento de uma ferramenta de gerenciamento e monitoramento de ativos de TI, como uma solução proposta ao problema, que é apresentado através de levantamento bibliográfico realizado principalmente em livros e sites referentes ao assunto, de como realizar esta atividade, simplificando sua execução. Foi criado para isso um protótipo do que pode se tornar o sistema, que atendesse as necessidades e requisitos básicos levantados através do referencial teórico, para que fosse possível a partir deste verificar a viabilidade de utilização e criar, futuramente, um sistema mais completo. Para que fosse possível o desenvolvimento desta aplicação foram utilizadas técnicas de Engenharia de Software, como a modelagem de dados do banco de dados, a criação de diagramas UML, definição de arquitetura de software MVC (*Model, View* e *Controller*), bem como escolhidas as ferramentas para se realizar o desenvolvimento, como linguagem de programação e de criação de interface gráfica, sistema gerenciador de banco de dados, estrutura de servidor de aplicação, que possibilitasse o acesso de várias maquinas a uma única aplicação e banco de dados. Com isso foi possível concluir a criação do protótipo funcional e a verificação de que é viável a utilização deste em um ambiente real, destacando-se também a possibilidade de no futuro implementar novas funcionalidades, seja de monitoramento ou gerenciamento, nesta ferramenta, tornando esta um sistema mais completo e robusto, que possa atender a requisitos mais específicos e complexos que solucionem, de forma mais abrangente, o problema da complexidade do gerenciamento e monitoramento de ativos de TI.

Palavras Chave: gestão de TI; monitoramento; desenvolvimento de software.

ABSTRACT

The present work presents the steps of the development of an IT asset management and monitoring tool, as a proposed solution to the problem, which is presented through a bibliographic survey carried out mainly in books and websites related to the subject, how to carry out this activity, simplifying its execution. For this purpose, a prototype of what can become the system was created, which met the basic needs and requirements raised through the theoretical reference, so that it was possible to verify the viability of use and to create a more complete system in the future. To be able to develop this application, software engineering techniques were used, such as data modeling database, creating UML diagrams, defining MVC (*Model*, *View and Controller*) *software architecture, and choosing the tools to perform the development, such as programming language and graphical interface creation, database manager system, application server structure, which allows the access of several machines to a single application and database. With this, it was possible to conclude the creation of the functional prototype and the verification that it is viable to use it in a real environment, also highlighting the possibility of in the future to implement new functionalities, be it monitoring or management, in this tool, to a more complete and robust system that can attend more specific and complex requirements that more fully solve the complexity of managing and monitoring IT assets.*

Keywords: *IT management; monitoring; software development.*

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1. INTRODUÇÃO

No cotidiano das organizações a dependência de equipamentos tecnológicos, especialmente aqueles de comunicação em rede, entre eles impressoras, computadores e notebooks, *access point*, servidores para diversos fins, e muitos outros, cresce cada vez mais. Junto a isso também surgem necessidades de gerenciar e monitorar todos esses equipamentos, garantindo que as atividades chave da organização sejam desenvolvidas sem interferências, e que qualquer problema seja detectado e resolvido o mais rápido possível.

Frente a essa necessidade, foi proposto neste trabalho o desenvolvimento de um sistema para o gerenciamento e monitoramento de *hosts*, equipamentos e acessórios, chamados de ativos de T.I. Um software que trabalhe na rede interna de uma organização, e possa ser acessado de várias máquinas, através de um navegador de internet, que auxilie a comunicação dentro da organização, agilize a detecção de problemas e, consequentemente, a tomada de decisões para resolver esses problemas.

O sistema proposto tem como objetivo fazer o gerenciamento dos equipamentos tecnológicos, gerenciando o estoque e a distribuição desses, auxiliando o rastreamento e a alocação de ativos, trazendo rapidez na tomada de decisões sobre todos os recursos e economia, através da possibilidade de detecção de reutilização destes recursos.

Também propôs-se incluir um mecanismo no sistema para o monitoramento dos equipamentos que se comuniquem na rede, utilizando-se de protocolos de rede para esse monitoramento, e registrando os resultados para análises e geração de relatórios, gráficos e histórico para auxiliar na detecção de erros e problemas em equipamentos, reduzindo o tempo entre o processo de detecção e manutenção de serviços e ativos. O sistema foi idealizado ainda para permitir acesso de usuários da organização, para que os *stakeholders* de um ativo tenham acesso a informações importantes sobre este. Tudo isso auxiliando o departamento de TI a possuir métricas para medir sua eficiência.

Para o desenvolvimento deste projeto, procurou-se referencial teórico em livros, sites e fontes de pesquisas para levantamento bibliográfico sobre o gerenciamento da

infraestrutura de TI nas organizações, assim como monitoramento dessa infraestrutura. Assim detectou-se que o desafio desse gerenciamento é um tema pertinente para a área de TI, e que o sistema proposto é uma ferramenta que pode contribuir para o tema, e para o desafio da gerência.

Este trabalho foi estruturado em três grandes partes, sendo a primeira, dos três primeiros capítulos, o levantamento teórico de informações sobre gerenciamento e monitoramento de ativos de TI, apresentando todos os conceitos teóricos utilizados para o levantamento de requisitos a serem atendidos. A segunda parte, que compreende o capítulo cinco, apresenta a Engenharia de Software, com a modelagem de requisitos, modelagem de dados, diagramas UML e arquitetura de software, e a terceira parte, compreendendo o capítulo seis, apresenta o desenvolvimento da ferramenta, junto com as definições técnicas do sistema, como linguagens e ferramentas utilizadas, além de exibir o resultado obtido, que é a criação de um protótipo do sistema.

2. INFRAESTRUTURA DE TI

A tecnologia, ao longo dos anos, alterou profundamente a forma como empresas ou qualquer outro tipo de organização funcionam, a forma como agem e também como pensam em seus objetivos, recursos e investimentos. Neste cenário, a área de tecnologia da informação cresceu exponencialmente dentro dessas organizações por se alinhar com a estratégia de negócio das mesmas, passando a ser vista não como um gasto extra, mas sim como um investimento que pode, se for bem feito, trazer redução de gastos ao invés de aumentá-los, além de maior produtividade e agilidade nos processos realizados na organização.

Para isso a TI conta com diversos recursos (*resources*), equipamentos (hardware), sistemas (software), serviços (*services*) rede (*network)* os quais se tornaram parte imprescindível para o funcionamento de qualquer organização. A integração de tudo isso é a chamada de Infraestrutura de TI. Turban e Volonino (2013, p.9) vão dizer que: *"A integração da tecnologia da informação (TI) forma uma estrutura de base, e essa infraestrutura desencadeia uma nova onda de desenvolvimento e avanços em TI"*. Essa estrutura de base, a infraestrutura de TI, é o começo de todos os avanços da organização em direção a uma nova realidade tecnológica e é ela que sustenta todos os serviços prestados, dos quais dependem o correto fluxo de trabalho da organização. A infraestrutura de TI inclui gerenciar, manter, operar e documentar esses recursos, conhecidos como Componentes da TI.

Esses componentes responsáveis pelo avanço tecnológico e melhor performance das atividades podem ser classificados como:

Hardware: todos os equipamentos físicos que suportam a TI dentro da organização, que podem ser classificados como: servidores, que entregam algum serviço; gerenciáveis, que possuem interface de acesso através de programas de computador, e não gerenciáveis; ativos de comunicação em rede que são base para o funcionamento da rede e comunicação da organização; peças que são incrementadas em outros ativos ou complementares, entre outras definições. (Ex.: Computadores, servidores, impressoras, teclados, mouses, etc.).

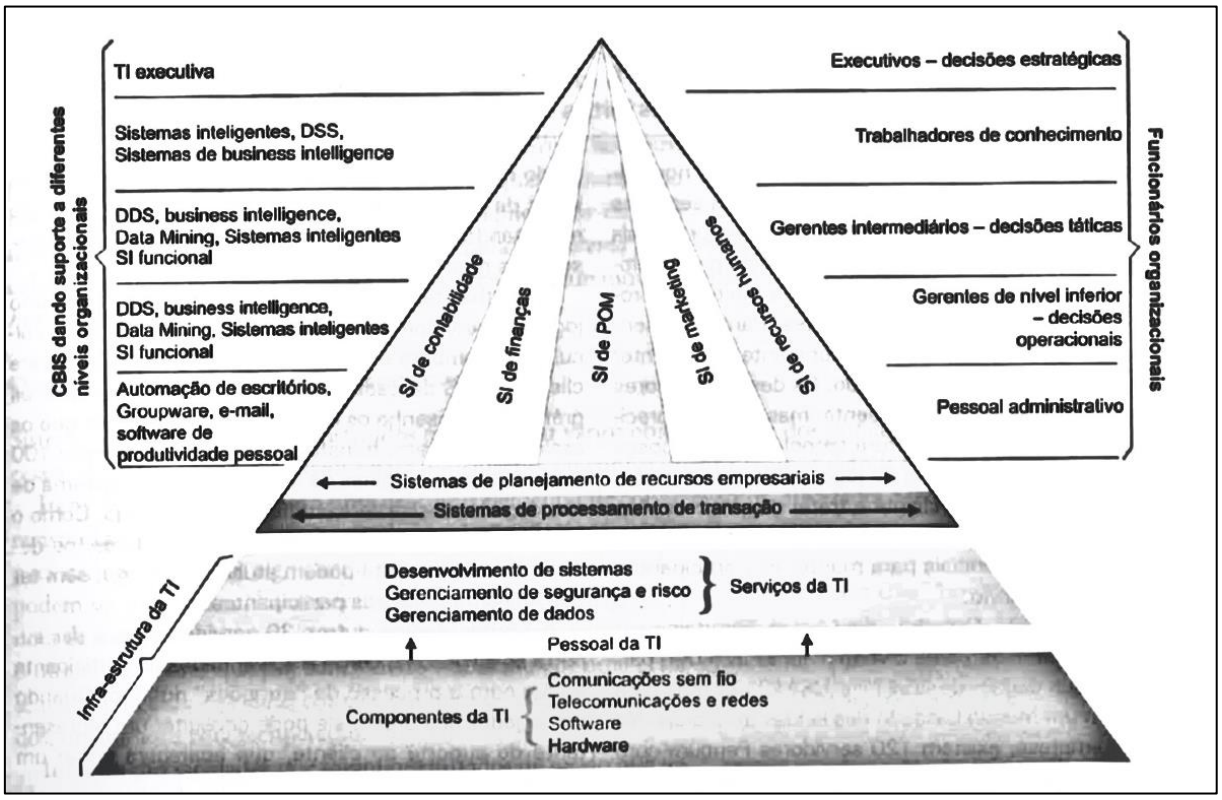
Software: todos os programas de computadores, sejam de sistema ou aplicações, suas licenças de uso, ativos de propriedade intelectual utilizados na

organização, linguagens de programação. Eles permitem que ao hardware acesso aos dados e processamento deles. (Ex.: sistemas operacionais, programas de planilhas, edições de texto, entre outros).

Banco de Dados: dados organizados e depositados em um repositório, ou vários repositórios, que se relacionam. É no banco que o software buscará dados para o hardware processar, e devolverá informações processadas para armazenamento e futuramente recuperação.

Rede: toda a estrutura de hardware, software e banco de dados, em conjunto e conectados por fio ou não, onde é possível acessa-los e compartilhas recursos. Não se pode falar de infraestrutura de TI sem a rede, pois através dela os equipamentos e sistemas podem comunicar-se e cooperar na criação, armazenamento e compartilhamento de informação.

Figura 1 – Pirâmide da Tecnologia da Informação



Fonte: Turban, Rainer Jr. E Potter (2005, p. 41).

Pode-se considerar também que a infraestrutura abrange mais componentes, como ativos prediais que suportam os ativos de hardware e mais componentes que podem ser considerados essenciais ou não para a TI. Na figura 1 podemos visualizar a Infraestrutura de TI dentro do contexto da TI em geral, notando que são os componentes que formam base da pirâmide da TI.

Entretanto a infraestrutura não se limita a esses componentes, ela também está profundamente ligada aos serviços prestados e suportados definidos como conjunto de atividades, recursos e possibilidades que a integração de hardware e software oferecem. Serviço é o que a TI deve entregar, enquanto prestadora de serviço dentro da organização. Esses serviços incluem, como diz Turban, Rainer Jr. E Potter (2005, p. 40): “o gerenciamento de dados desenvolvimento de sistemas e aspectos de segurança”.

Para este trabalho, o foco recai principalmente sobre os componentes da TI, nos objetos chamados de Ativos de TI. Nos próximos capítulo será explorado a definição e a importância de gerenciar e monitorar esses Ativos de TI.

3. GERENCIAMENTO DE ATIVOS DE TI

Quando se fala de ativos dentro de uma organização deve-se ter em mente que estes são tudo aquilo que tem ou traz algum valor para a organização, e dentro desse quadro geral, os ativos de TI são todos os ativos da organização que influenciam, tem valor ou são responsabilidade do departamento de TI. Essa definição inclui todos os componentes de TI da infraestrutura já citados e incluem as pessoas que interagem com os outros componentes e tudo mais que, dentro de cada organização, for de responsabilidade desse departamento.

A infraestrutura de TI é que sustenta e forma a base para todos os serviços entregues, sendo a responsabilidade e a missão de mantê-la, ampliá-la quando necessário, e garantir que todos os serviços continuem sempre disponíveis, papel da gerencia de TI das organizações. O gerenciamento de ativos faz parte dessa responsabilidade, sendo ele um ponto de partida para gerenciar toda a infraestrutura. Ele é feito com inventário de ativos, distribuição de recursos, atendimento ao usuário, monitoramento, e outras atividades, e é de grande importância gerenciar esses recursos, pois a dependência da tecnologia e de seus serviços fazem com que seja crucial para o negócio a disponibilidade desses.

A importância de um ativo traz consigo uma responsabilidade sobre ele, sem gerenciar os ativos a área de TI de uma empresa não tem controle de quem utiliza o que, com quem está cada recurso, e se houvessem falhas, quem seria afetado por elas. Não gerenciar também significa não usar de forma inteligente, não saber quando é necessário investir e descartar, não absorver ao máximo as vantagens que a TI oferece a empresa. Por isso existe a necessidade do gerenciamento e também da utilização de ferramentas para essa atividade, trazendo eficiência, controle e segurança para a TI.

É importante ressaltar que não se pode limitar apenas a levantar um inventário dos equipamentos e saber que eles existem, é necessário ter muito mais informações sobre cada ativo, quem tem responsabilidade sobre ele, quais as interações que este tem com pessoas, softwares, licenças, hardware e tudo que envolve a infraestrutura da organização. Com essas informações claras é possível ter uma noção muito maior do todo, e saber onde cada peça se encaixa dentro deste quebra cabeça que é a TI,

mas para isso é necessário possuir as ferramentas certas, que auxiliem e descompliquem, na medida do possível, essa tarefa. Nesse contexto, tem-se Magalhães e Pinheiro (2010, p. 23) concordando que:

> "O gerenciamento da infraestrutura de TI é feito primordialmente com o auxílio de ferramentas de gerenciamento de sistemas providas pelos diferentes fornecedores desta área. Os processos de Gerenciamento de Serviços de TI são suportados por ferramentas especificas para tanto, as quais necessitam interagir com as aquelas de gerenciamentos de sistemas".

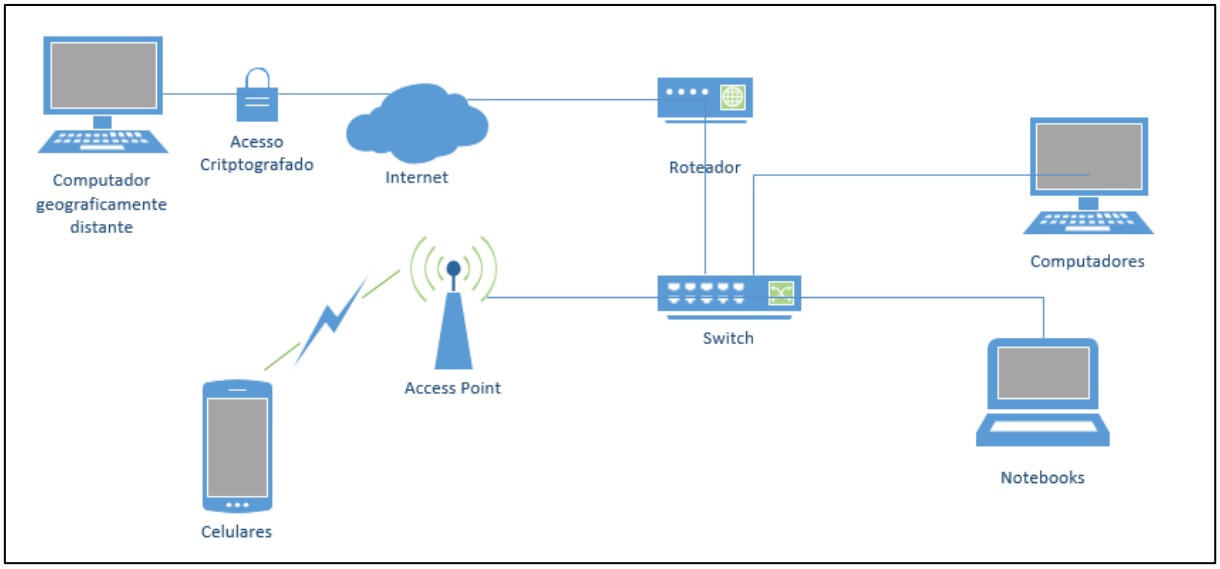
Com essa demanda, torna-se uma necessidade possuir ferramentas que auxiliem na realização dessa atividade, e existem diversas disponíveis e amplamente conhecidas pelos profissionais da área, sejam elas softwares ou boas práticas como os *frameworks* ITIL e Cobit, entre outros *frameworks*, ferramentas pagas ou gratuitas (de código aberto), e é preciso saber quais ferramentas utilizar e de que forma integrá-las para garantir o melhor gerenciamento de acordo com a realidade da organização. Também é necessário saber quais ativos gerenciar, pois, embora em um quadro geral todos os componentes citados sejam importantes, precisamos classifica-los e entender a necessidade deles para cada organização.

Este projeto se propôs a desenvolver uma ferramenta que faça o inventário desses ativos e mais que isso, ajude a gerenciar a utilização, distribuição, tempo de vida e que entregue resultados para tomadas de decisão baseada em um gerenciamento eficiente. Nos próximos capítulos serão abordadas outras faces do gerenciamento no monitoramento de rede, as definições do sistema e suas funcionalidades.

4. MONITORAMENTO DE ATIVOS DE TI

Empresas e instituições foram moldadas ao longo dos anos, conforme a tecnologia de rede de computadores foi inserida nesses ambientes transformando a maneira de se trabalhar e de se organizar. A rede interna, ou LAN (*Local Area* Network) nessas organizações, também chamada de intranet, que segundo Comer (2016, p. 296), “é de propriedade de uma organização [...] e projetada para ser usada somente por funcionários dessa companhia”, em contraponto à internet, rede global, é uma rede local, que existe fisicamente por conexão à cabo ou sem fio de equipamentos numa mesma localidade geográfica, que ainda possibilita a conexão lógica e criptografada passando pela internet, a chamada VPN (*Virtual Private Network*), como exemplificado na figura 2.

Figura 2 - Exemplo de uma rede interna (intranet)



Fonte: Autoria Própria.

Na figura 2 podemos ver um exemplo de uma rede interna que se conecta a internet e conecta computadores geograficamente distantes através da VPN, uma rede privada na internet.

A rede local forma uma parte fundamental da infraestrutura de TI, conectando todos os serviços, equipamentos e recursos, fazendo dela uma peça importante para a continuidade de negócio da organização. Nesse cenário é necessário um contínuo

monitoramento da rede, para detecção de gargalos, falhas, e possíveis erros ou paradas totais, em casos mais extremos. Essa reponsabilidade não se restringe apenas a rede interna, mas a externa (internet) que graças a necessidade de conexão com sites, serviços e compartilhamentos de terceiros torna-se tão essencial para empresa quanto a interna, ainda que não tão controlável, tendo em vista que o papel de prover e dar manutenção ao serviço é de empresas de telecomunicação contratadas. É de grande importância monitorar as duas redes, pois uma falha pode gerar transtorno, perda de dinheiro para a empresa, isolamento que compromete todos os acessos da empresa, onde pode-se exemplificar claramente com a rede bancaria, onde as transações são todas online, portanto uma falha de conexão pode comprometer o pagamento de funcionários, assim como uma falha na rede interna pode comprometer o caminho de acesso para a rede externa.

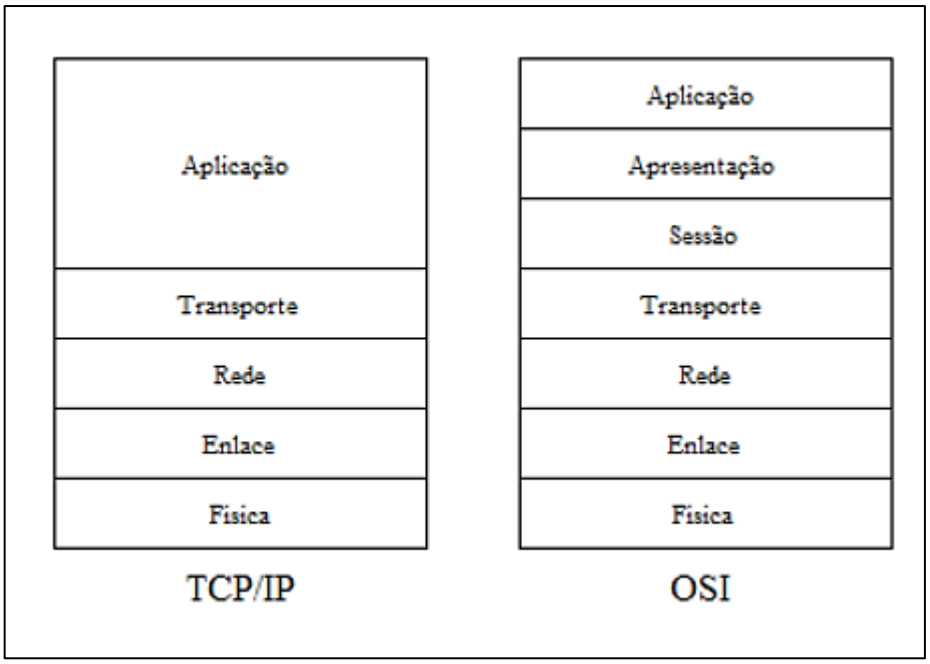
Dentro da rede local acontece a comunicação constante entre os equipamentos, que é vital par a continuidade do negócio da organização, pois, se um ativo que tem capacidade de estar conectado não se comunica com o restante da rede, há um problema que deve ser resolvido. O meio de monitorar esses ativos é utilizando a comunicação na própria LAN, assim quando um ativo não se comunica, essa falha, mesmo que seja apenas uma demora de reposta ou pequena interferência, se for monitorada pode evitar problemas maiores e adiantar a tomada de decisão para manutenção ou melhorias. Deve-se ressaltar que nem todos os ativos são passíveis desse monitoramento, apenas aqueles que tem capacidade de se comunicar através dos protocolos de rede, desses ativos podemos citar computadores, notebooks, servidores, impressoras, switches, e outros tantos.

Como nas organizações é através da rede que se acessa os serviços, sistemas, equipamentos, e até mesmo se faz o *login* no computador, ou seja, pela rede se acessa praticamente todos os recursos de TI, então a rede torna-se o caminho ideal para o monitoramento. Por isso a ferramenta de proposta deste trabalho utilizará a rede local como meio de monitoramento dos serviços de TI, e fará isso através de protocolos de rede, que serão abordados no próximo capitulo.

4.1. MONITORAMENTO DE PROCOLOS DE REDE

Segundo Esteves (2013, p. 16) “Um protocolo é definido como a descrição formal de como são realizadas as comunicações entre equipamentos de rede, [..] e assim conseguir fazer transferência de dados entre estes dispositivos”. Existem diversos protocolos de rede, cada um para um serviço e atuando em uma camada diferente de rede, seja no modelo TCP/IP ou ISO/OSI como mostra a figura 3.

Figura 3 – Comparação entre os modelos TCP/IP e OSI.



Fonte: Esteves (2013, p. 23).

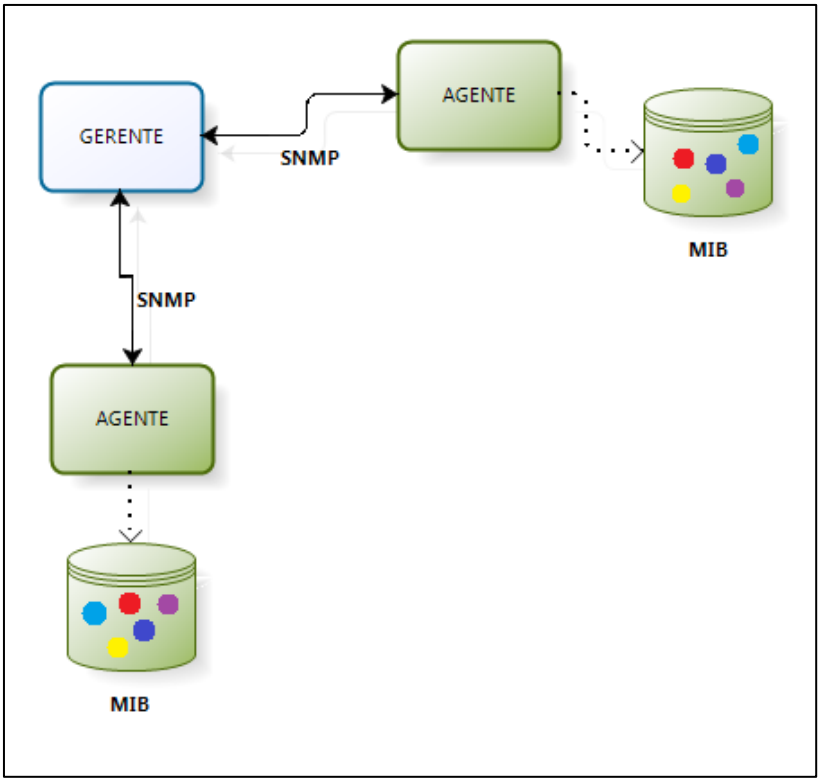
Caso algum protocolo falhe e a comunicação não seja estabelecida, é um sinal que algum serviço pode não estar funcional, e saber disso com antecedência é de extrema importância.

Diversos dos protocolos de rede podem ser monitorados, mas para a ferramenta foram escolhidos três protocolos específicos, apresentados abaixo.

4.1.1. SNMP

O SNMP (*Simple Network Management Protocol*, ou Protocolo Simples de Monitoramento de Rede) trabalha na camada de aplicação do modelo ISO/OSI ou TCP/IP e é o protocolo mais utilizado para gerenciamentos de ativos e serviços de rede. O SNMP trabalha com alguns componentes que podem ser vistos na figura 4.

Figura 4 – Modelo de gerenciamento do SNMP.



Fonte: Site TelcoManager[1]

O agente SNMP, que está no ativo a ser monitorado e que guarda informações sobre o estado do hardware e outros valores que são importantes e também manipula informações que podem ser alteradas dentro desse ativo, quando solicitado. Temos o gerente que pode obter e definir informações dos agentes, através de uma coleção de objetos chamados de MIB (*Management Information Base*), onde cada objeto é identificado por uma OID (*Object Identifier*) e pode realizar operações GET onde a gerente apenas faz requisições ao agente que devolve informações, SET onde o gerente define valores no ativo através do agente e também TRAP onde o agente vai notificar ao gerente eventos que indiquem problemas.

Dessa forma, como o agente e o gerente assumem variavelmente o papel de cliente e servidor dependendo da operação, não se pode falar aqui de uma estrutura cliente-servidor, por isso os termos agentes e gerentes. A partir da coleta de informações que o SNMP possibilita é possível tomar decisões com precisão, saber aonde atuar para resolver problemas e tomar medidas efetivas, já que são muitos os equipamentos de hardware que entendem esse protocolo de rede.

[1] Disponível em: <https://www.telcomanager.com/pt-br/blog/o-que-e-snmp> Acesso em out. 2017.

4.1.2. ICMP

O protocolo ICMP, *Internet Control Message Protocol*, é parte do protocolo IP (*Internet Protocol*) e tem como função ser um relator de problemas, pois segundo Esteves (2013, p. 32), “indica ocorrência de problemas no transporte de algum datagrama, ou ainda é utilizado em operações de controle”, fazendo desse um protocolo útil para testar de forma simples se um ativo está conectada na rede, se comunicando, através do IP (*Internet Protocol*), já que todos os ativos conectados à rede precisam se comunicar através do IP.

Sua utilização mais comum é o famoso comando “ping”, padrão em todos os sistemas operacionais que envia pacotes ICMP para um IP especificado retornando perdas de pacotes, falhas no envio, e erros de roteamento, embora o protocolo não traga nenhuma solução ao problema, ele é um ótimo ponto de partida para detecção e repostas rápidas a problemas com o host, principalmente por ser um protocolo universal para qualquer equipamento. Existem, porém, outros comandos, como o traceroute que faz um “mapa” de roteadores por onde o pacote ICMP passa, mas para a ferramenta proposta apenas o ping será utilizado.

O pacote ICMP devolve como resposta ao ativo que o enviou mensagens de erro, como as apresentadas a seguir:

- *Network Unreachable* (rede não alcançável);
- *Host Unreachable* (host não alcançável);
- *Port Unreachable* (port não alcançável);
- *Destination Host Unknown* (Host destino desconhecido);
- *Destination Network Unknown* (rede destino desconhecida).

Essas e outras diversas mensagens, são úteis para descobrir qual a fonte do erro de comunicação, e ao monitorar constantemente esses erros é possível saber se é um erro comum, o que indicaria algum ativo defeituoso, ou se é um erro incomum, indicando que pode ser algo momentâneo, como uma colisão de pacotes na rede ou um equipamento que está desligado por queda de energia e falha no *nobreak*.

4.1.3. HTTP/HTTPS

O protocolo HTTP, Hypertext Transfer Protocol, ou Protocolo de Transferência de Hipertexto segundo Forouzan (2009, p. 861) “é um protocolo usado principalmente

para acessar dados da web”, padrão de transferência de arquivos na internet, pois ainda segundo Forouzan (2009, p. 861) “permite a transferência de arquivos e usa serviços do TCP na conhecida porta 80”, trabalhando na camada de aplicação dos modelos TCP/IP e OSI.

Ele funciona na comunicação de um servidor web que armazena páginas e conteúdo que estão acessíveis através de uma URL que são requisitados por um cliente, e transferidos para ele. O protocolo HTTPS, *Hypertext Transfer Protocol Secure*, ou Protocolo de Transferência de Hipertexto Seguro, por sua vez utiliza o HTTP e soma a ele uma camada de segurança com protocolos SSL (*Secure Sockets Layer*) ou TSL (*Transport Layer Security*), criptografando a conexão e validando, através de certificados digitais emitidos por entidades reconhecidas, o servidor web.

Muitos serviços que as organizações consomem são disponibilizados na internet, e mesmo aqueles que são locais muitas vezes estão em um servidor web interno, fazendo desse protocolo um importante indicador de disponibilidade de serviços e ativos que os oferecem. Através do monitoramento dessas páginas, se estão acessíveis através da URL, é possível monitorar se um serviço Apache está rodando em um servidor, por exemplo, ou se uma página está com erros que impossibilitem a sua interpretação pelos navegadores, se existe gargalos na rede que impossibilitem a conexão com internet ou com o servidor local, entre outros erros que as falhas de comunicação por esses protocolos podem indicar.

Para a ferramenta deste projeto, mais do que verificar se algum site está respondendo, é verificar qual a resposta de erro retornará caso ele não responda, e partindo dessa mensagem pode-se encontrar o erro mais facilmente e tomar medidas corretivas mais rapidamente.

Todos esses protocolos apresentados foram escolhidos por serem altamente difundidos e utilizados, e por não necessitarem de nenhuma instalação extra de scripts ou software nos equipamentos monitorados. Através desses é possível gerar relatórios, criar alertas e trazer agilidade na detecção e resolução de erros. Mais protocolos e meios de monitoramento podem ser adicionados a ferramenta, partindo das pesquisas que serão realizadas no decorrer do desenvolvimento.

Definidas as bases teóricas que demonstram a necessidade de uma ferramenta que gerencie e monitore os ativos de TI, no próximo capitulo será descrita a Engenharia de Software do sistema, com o levantamento de requisitos e a diagramação para o desenvolvimento do sistema.

5. ENGENHARIA DE SOFTWARE

Neste capitulo apresenta-se o processo de Engenharia de Software do sistema proposto por este projeto. Para se entender o que é a Engenharia de Software Sommerville (2010, p. 5) define que “Engenharia de Software é uma disciplina de engenharia relacionada a todos os aspectos de produção de software, desde os estágios iniciais até sua manutenção, depois que estar em operação”, isso, portanto, inclui técnicas, métodos e processos de especificação, modelagem de dados, arquitetura, desenvolvimento, documentação e manutenção, visando sempre facilitar a produção e garantir a qualidade do software.

Dentre os métodos de modelagem de dados e documentação foram escolhidos para o sistema que será desenvolvido o levantamento de requisitos, diagramas de casos de uso, classes e atividades, e a modelagem de banco de dados utilizando o diagrama de entidade e relacionamento e o padrão de arquitetura de software MVC. O levantamento de requisitos foi escolhido como ponto de partida, pois proporciona uma visão geral das funcionalidades que o sistema deverá ter, sem se preocupar em especificar como essas funcionalidades serão implementadas.

Logo após o levantamento dos requisitos são apresentados os diagramas da linguagem UML, que Guedes (2009, p. 19) define como:

> “[...] Linguagem de Modelagem Unificada – é uma linguagem visual utilizada para modelar softwares baseados no paradigma de orientação a objetos. [...] linguagem-padrão de modelagem adotada internacionalmente pela indústria de engenharia de software [...]”.

A utilização do paradigma de orientação a objetos pede que se crie, pelo menos, os diagramas de casos de uso, classes e atividades, como base para a etapa de desenvolvimento. Além disso, eles documentam o sistema para facilitar a interpretação por parte de outros programadores, também facilitando a manutenção do sistema. Destaca-se que dentre os diagramas apresentados, o diagrama de classes foi feito em uma etapa posterior ao desenvolvimento.

Apresenta-se também a modelagem de dados utilizando o diagrama ER, muito utilizado para modelar o banco de dados, este considerado um pilar que sustenta a aplicação, o que exige um grande cuidado nesta modelagem, para que esta harmonize com o desenvolvimento da aplicação. Por último é apresentado as definições do MVC, padrão de arquitetura de software adotado neste projeto,

importante para organização para a especificação, desenvolvimento e manutenção do sistema.

No tocante a metodologia de Engenharia de Software, nenhum modelo foi seguido, pela dificuldade de aplicá-los na execução feita sem uma equipe responsável pela divisão das várias atividades, fazendo com que o processo não pudesse seguir metodologias de desenvolvimento tradicionais ou ágeis.

Esta etapa de Engenharia de Software tornou possível um desenvolvimento muito mais padronizado e descomplicado, facilitando possíveis manutenções futuras e criação de novas funcionalidades. A seguir as partes que compuseram esta etapa.

5.1. LEVANTAMENTO DE REQUISITOS

Ao se iniciar o projeto de sistema novo uma das primeiras etapas, ou talvez a primeira, para concebe-lo é fazer o levantamento e análise de requisitos. Esta etapa busca compreender o que será feito, analisando aquilo que se deseja realizar, e nesses termos Guedes (2009, p. 22) informa que:

> "As etapas de levantamento e análise de requisitos trabalham com o domínio do problema e tentam determinar 'o que' o software deve fazer e se é realmente possível desenvolver o software solicitado. Na etapa de levantamento de requisitos, o engenheiro de software busca compreender as necessidades do usuário e o que ele deseja que o sistema a ser desenvolvido realize. "

É uma atividade cíclica, a qual sempre se volta buscando comparação do que foi definido e o que foi feito, adicionando novos requisitos ou retirando requisitos que já não são mais considerados parte do escopo do projeto.

Para uma completa compreensão do sistema a ser desenvolvido levantou-se os requisitos baseados nas necessidades descritas nos capítulos de referencial teórico de gerenciamento e monitoramento de ativos de TI. Abaixo foram separados os requisitos funcionais, aqueles que representam ações do sistema, e requisitos não funcionais, aqueles que apresentam características do sistema.

Requisitos funcionais

- Manter cadastros de ativos de TI (ativos de rede, componentes, equipamentos, licenças);
- Manter cadastros de categorias de ativos de TI;
- Registrar informações importantes dos ativos de rede (S.O, modelo, fabricante, categorias);
- Manter histórico de manutenção de ativos de rede;
- Manter cadastro de interfaces de rede de ativos de rede;
- Manter cadastros e acessos de usuários;
- Relacionar ativos e usuários responsáveis;
- Relacionar ativos e componentes;
- Relacionar equipamentos e usuários responsáveis;

- Gerar relatórios das informações de gerenciamento;
- Monitorar ativos de rede por protocolos de rede (ICMP, HTTP/HTTPS, SNMP);
- Definir periodicidade de cada um dos monitoramentos;
- Escolher qual ativo de rede monitorar ou não;
- Manter cadastros de OID do protocolo SNMP;
- Relacionar OID com ativos de rede monitoráveis;
- Notificar *stakeholders* (que são usuários do sistema) de falhas através do e-mail;
- Gerar relatórios das informações de monitoramento.

Requisitos não-funcionais

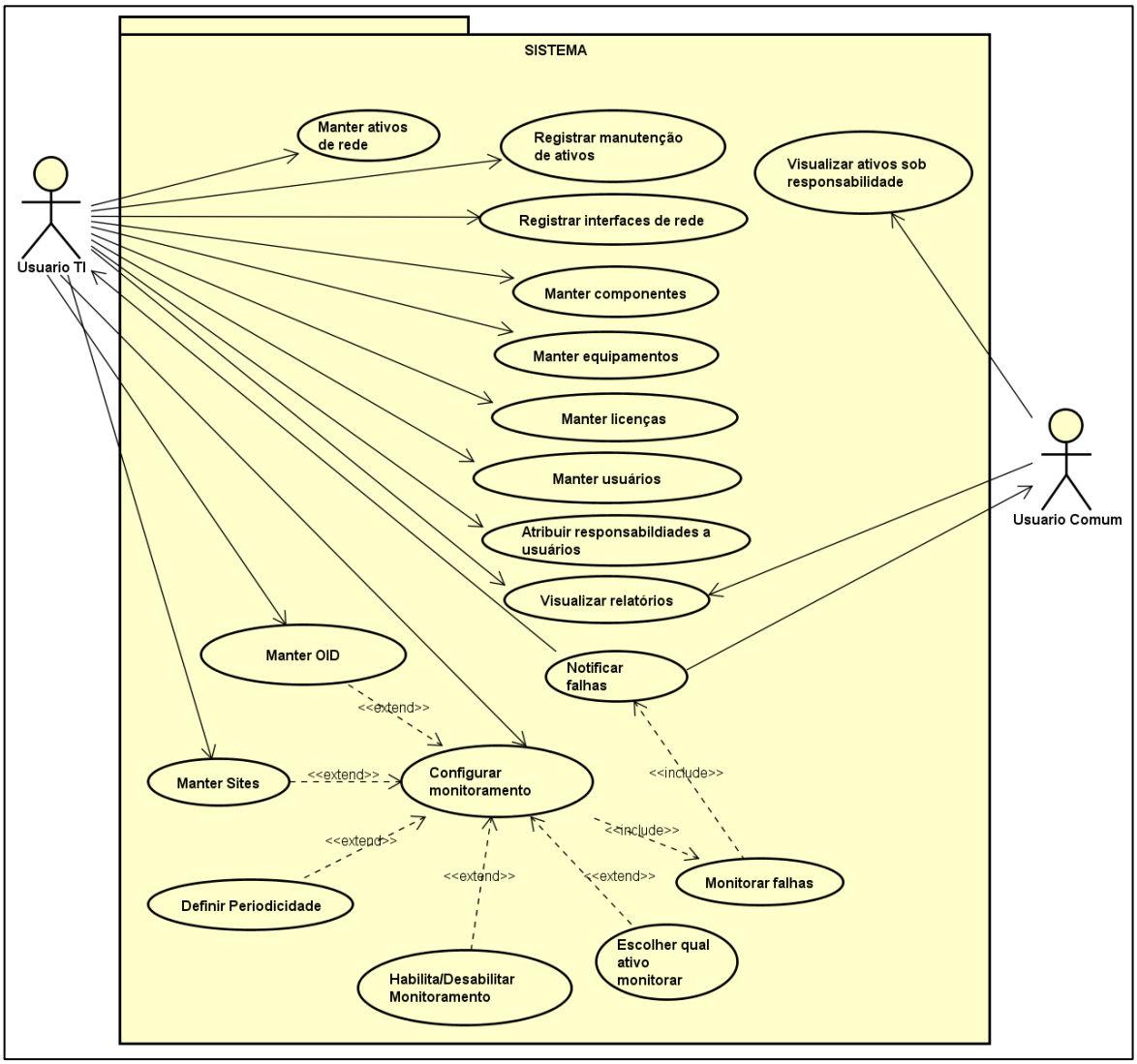
- Back-end compatível tanto com servidor Linux quanto Windows;
- Front-end responsivo e acessível tanto por computador quanto por celular;
- Facilidade na utilização do sistema;
- Execução rápida em qualquer dispositivo;
- Facilidade na visualização de informações.

5.2. DIAGRAMA DE CASOS DE USO

O diagrama de casos de uso “mostra conceitualmente o conjunto de funções que o sistema deve executar para atender aos requisitos do cliente, servindo como um contrato entre o cliente e o desenvolvedor” (LIMA, 2010, p. 57). Embora não exista um cliente especifico para este sistema, ele deve atender os requisitos que foram levantados de acordo com a necessidade observada.

Na figura 5 temos o diagrama de casos de uso desenvolvido a partir dos requisitos funcionais levantados anteriormente que apresenta uma visão da interação dos usuários com o sistema. O diagrama foi desenvolvido com a ferramenta de criação de diagramas UML Astah Community.

Figura 5 – Diagrama de Casos de Uso



Fonte: Autoria própria.

É possível notar que existem dois usuários para esse sistema, sendo eles o “Usuário Comum”, que seria qualquer *stakeholder* que precisa visualizar alguma informação no sistema, e o “Usuário TI”, que é o administrador do sistema e quem realiza as maiores ações dentro do sistema. A seguir a descrição de cada caso de uso:

- **Manter ativos de rede:** cadastrar, editar e excluir ativos de rede. Este caso de uso também abrange cadastrar, editar e excluir modelos, sistemas operacionais, fabricantes e categorias, que são informações relacionadas e pertinentes para os ativos de rede;
- **Registrar manutenção de ativos:** cadastrar e editar manutenções realizadas em um ativo de rede;
- **Registrar interface de rede:** para cada ativo cadastrar, editar ou excluir uma ou mais interfaces de rede;
- **Manter componentes:** cadastrar, editar e excluir componentes, que são internos aos ativos de rede. Inclui cadastrar, editar e excluir categorias para classificação;
- **Manter equipamentos:** cadastrar, editar e excluir equipamentos externos aos ativos, relacionando-os ao usuário. Inclui cadastrar, editar e excluir categorias para classificação;
- **Manter licenças:** cadastrar, editar e excluir licenças relacionadas a ativos. Também inclui cadastrar categorias, editar e excluir para classificação;
- **Manter usuários:** cadastrar, editar e excluir usuários do sistema. Inclui cadastrar, editar e excluir grupos, assim como relaciona-lo a usuários;
- **Atribuir responsabilidades a usuários:** relacionar ou remover a relação de um ativo ou um equipamento a um usuário do sistema;
- **Visualizar ativos sob responsabilidade:** acessar e visualizar, ao menos, todos os ativos e equipamentos sob a responsabilidade de um usuário;
- **Visualizar relatórios:** visualizar todos os relatórios gerados pelo sistema, tanto de monitoramento quanto de gerenciamento, dependendo da permissão do usuário;
- **Configurar Monitoramento:** este caso de uso inclui outros cinco tipos de monitoramento: ICMP (*ping*), HTTP/HTTPS e SNMP. Este caso de uso se

estende em quatro casos de usos opcionais, apresentados abaixo, e um obrigatório;

- **Manter OID:** cadastrar, editar, excluir e relacionar OID para ativos de rede, que será utilizado no monitoramento SNMP. Opcional ao caso de uso “Configurar Monitoramento”, porém pode ser realizada sem este caso de uso;
- **Manter Sites:** cadastrar, editar e excluir sites monitorados por HTTP/HTTPS. Opcional ao caso de uso “Configurar Monitoramento”, porém pode ser realizada sem este caso de uso;
- **Escolher qual ativo monitorar:** definir, através da interface de rede, qual IP será testado pelo monitoramento ICMP (*ping*). Opcional ao caso de uso “Configurar Monitoramento”;
- **Definir periodicidade:** Em todos os três monitoramentos, definir de quanto em quanto tempo ele será executado. Opcional ao caso de uso “Configurar Monitoramento”;
- **Habilitar/Desabilitar monitoramento:** Parar ou iniciar por tempo indeterminado e individualmente qualquer um dos três monitoramentos. Opcional ao caso de uso “Configurar Monitoramento”;
- **Monitorar falhas:** Obrigatoriamente o sistema irá monitorar as falhas através dos três monitoramentos após o caso de uso “Configurar Monitoramento”, gerando uma outra ação obrigatória;
- **Notificar falhas:** Quando o sistema detectar falhas, ele notificará os *stakeholders* do ativo com falha. Este é o único caso de uso que parte de uma ação do sistema para o usuário.

5.3. DIAGRAMA DE CLASSES

O diagrama de classes é um diagrama específico para o paradigma de orientação a objeto, que tem como base as classes de objetos. É importante tanto para uma fase anterior ao desenvolvimento, conceituando as classes que serão implementadas, com seus atributos e métodos, quanto para documentação dessas classes, para posteriormente utilizar esse diagrama para compreender as classes do sistema.

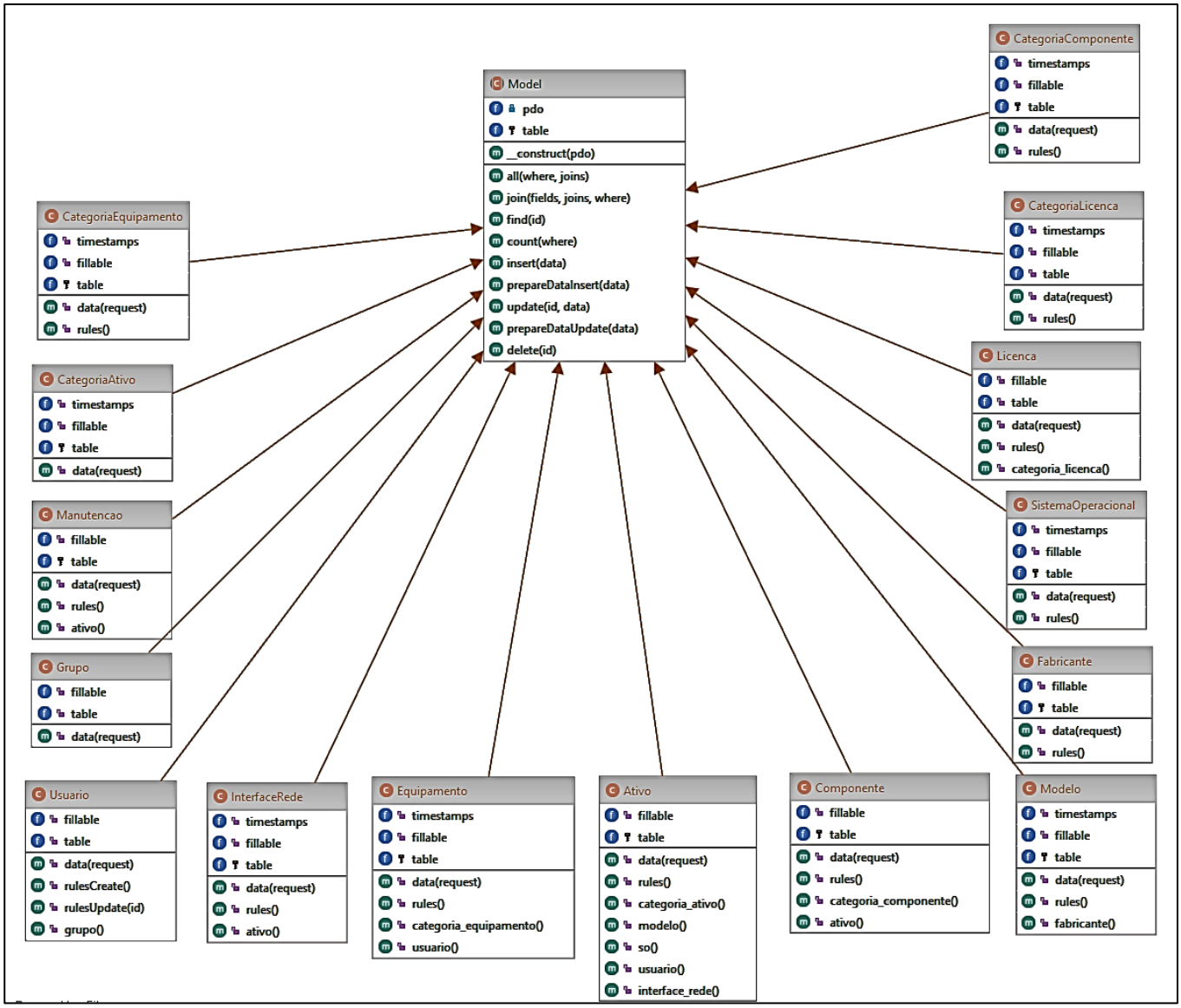
Guedes (2009, p. 106) vem dizendo sobre o diagrama de classes que este é “[...] uma visão estática de como as classes estão organizadas, preocupando-se em como definir a estrutura lógica das mesmas. [...] serve ainda como base para a construção da maioria dos outros diagramas da linguagem UML”.

Embora considerado importante antes do desenvolvimento, neste projeto foi realizado primeiro a criação dos códigos e todas as classes necessárias durante a execução, depois gerou-se os diagramas com a IDE PhpStorm, aplicando, portanto, engenharia reversa nesses códigos transformando-os nos diagramas apresentados a seguir. Este fato se deu escolha de uma linguagem de programação simples, o PHP, com pouco tipagem dos dados e dos retornos de classes, fazendo este diagrama com os relacionamentos, os atributos e métodos ter um caráter muito mais documental do que necessário para o desenvolvimento.

Os diagramas estão divididos em quatro: classes *Model* do Gerenciamento, classes *Model* do Monitoramento, classes *Controller* do Gerenciamento e classes *Controller* do Monitoramento.

Ressalta-se que as definições da Arquitetura de Software e padrão MVC estão descritas ainda neste capitulo, já as definições dos dois módulos do sistema, linguagem de programação, assim como outras demais definições técnicas, são apresentadas no capítulo 6 deste projeto.

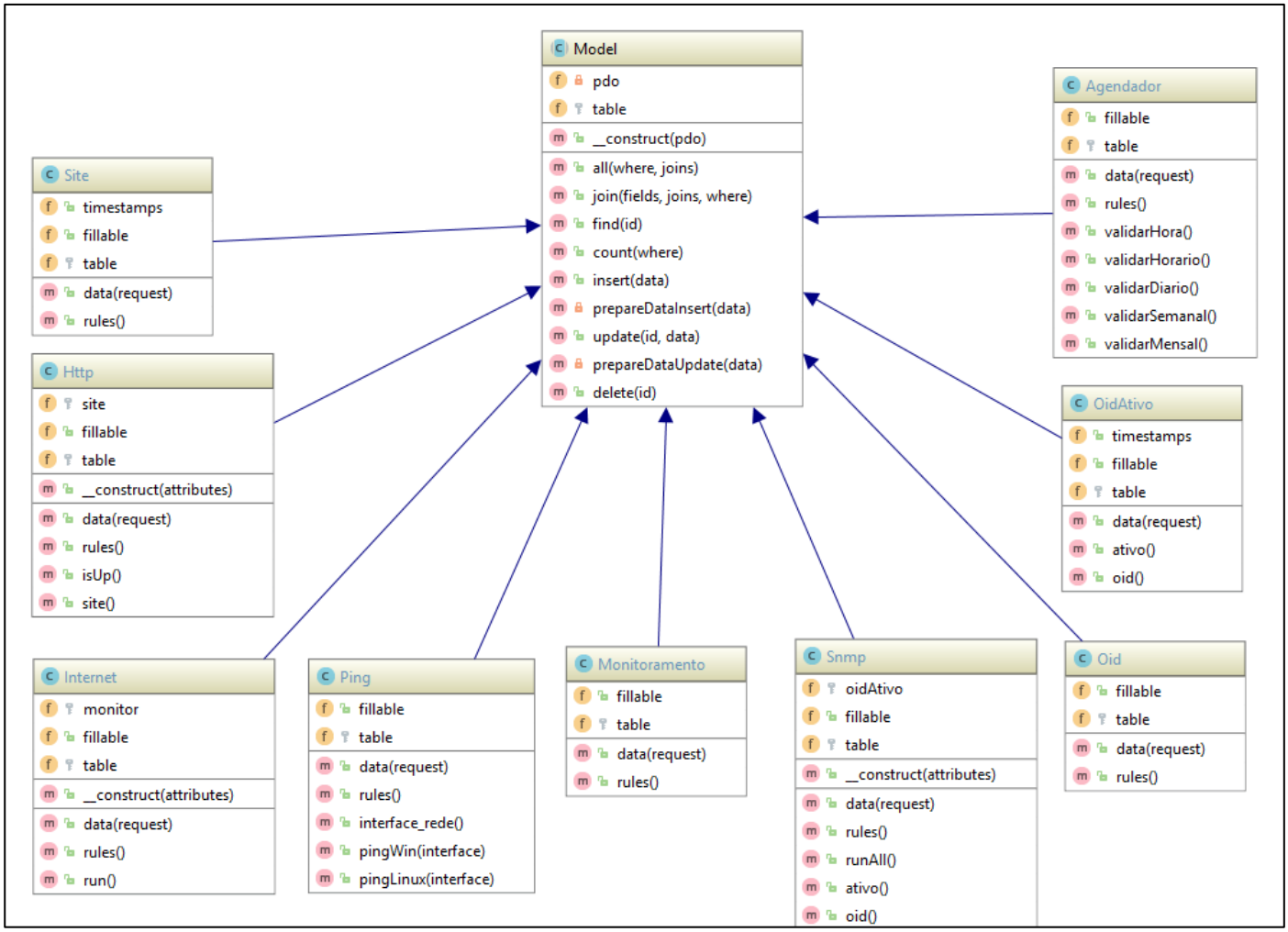
Figura 6 – Classes do Módulo de Gerenciamento da *Model*



Fonte: Autoria própria.

Na figura 6 pode-se ver o diagrama de classes da camada *model* do módulo de Gerenciamento, onde todas as classes herdam atributos e métodos da classe abstrata "*Model*", que trabalha a manipulação direta do banco de dados de forma mais genérica, possibilitando que todas as outras classes possam utilizar-se dos métodos herdados, como "*insert*", "*update*", "*delete*", "*find*" e outros. Também pode-se ver que todas as classes têm os métodos "*data*", que receberá os valores vindos do *front-end*, e os tratará, junto com o método "*rules*", que aplicará regras de validação de dados.

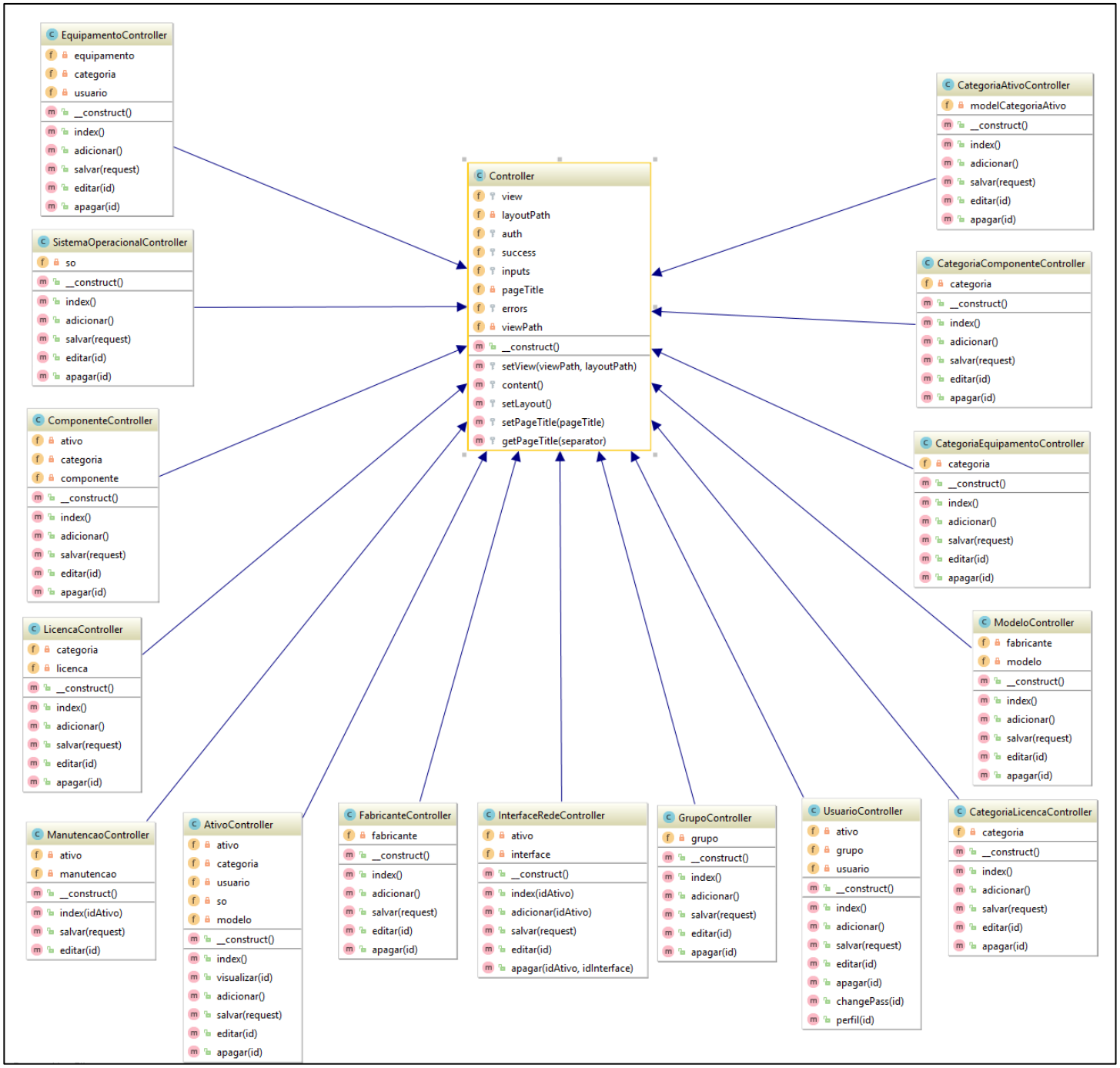
Figura 7 - Classes do Módulo de Monitoramento da *Model*



Fonte: Autoria própria.

O diagrama apresentado na figura 7 mostra todas as classes da camada *model* do módulo Monitoramento, seus atributos e métodos. É importante ressaltar que esta classe abstrata "*Model"* é a mesma da figura 6, e a divisão em dois diagramas foi feita apenas para tornar a visualização dos diagramas mais fácil. As classes herdam, portanto, os métodos de manipulação de banco de dados, e ainda nas classes "Snmp", "Ping", "Internet" e "Http" encontram-se os métodos que executam o monitoramento. Destaca-se também a classe "Agendador", responsável por validar a periodicidade e registrar no banco de dados e a classe "OidAtivo" que relaciona uma informação da classe "Oid" com a classe "Ativo", do módulo de gerenciamento.

Figura 8 - Classes do Módulo de Gerenciamento da *Controller*

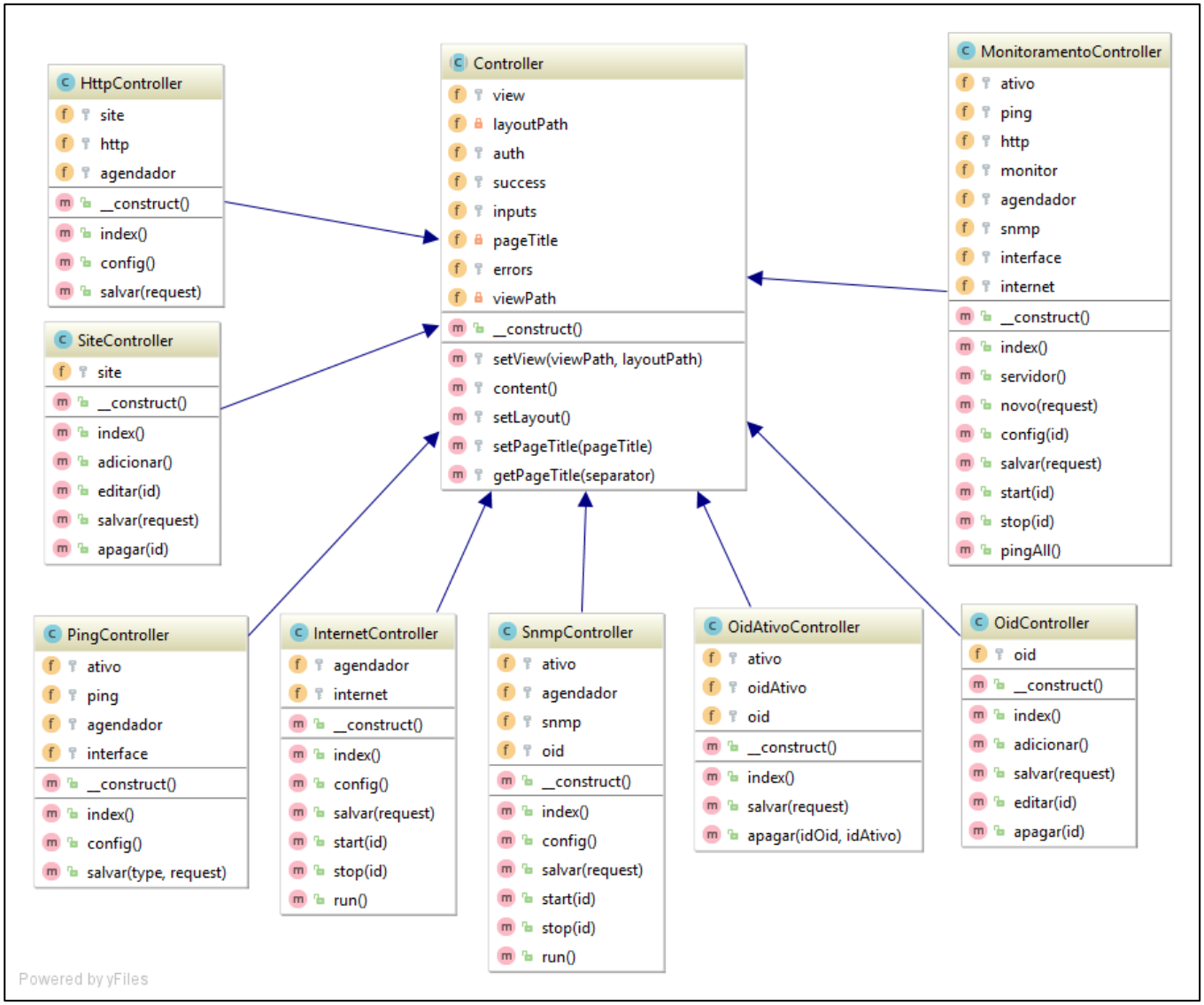


Fonte: Autoria própria.

A figura 8 mostra o diagrama com todas as classes da camada *controller* do módulo de Gerenciamento da aplicação. Esta camada é responsável por fazer a ligação entre a *view* e a *model*. Nota-se que todas herdam métodos e atributos da classe "*Controller"*, que contém todos os métodos comuns a todas as classes herdeiras, como manipulação das telas do sistema, com o método "*setLayout"* e

manipulação de autenticação com o atributo “*auth”*. Alguns atributos, como “*view”*, “*success”* e “*erros”* são utilizados na exibição de informações no *front-end*, já os atributos específicos de cada classe são objetos das classes da camada *model* para realizar as manipulações de dados.

Figura 9 - Classes do Módulo de Gerenciamento da Controller

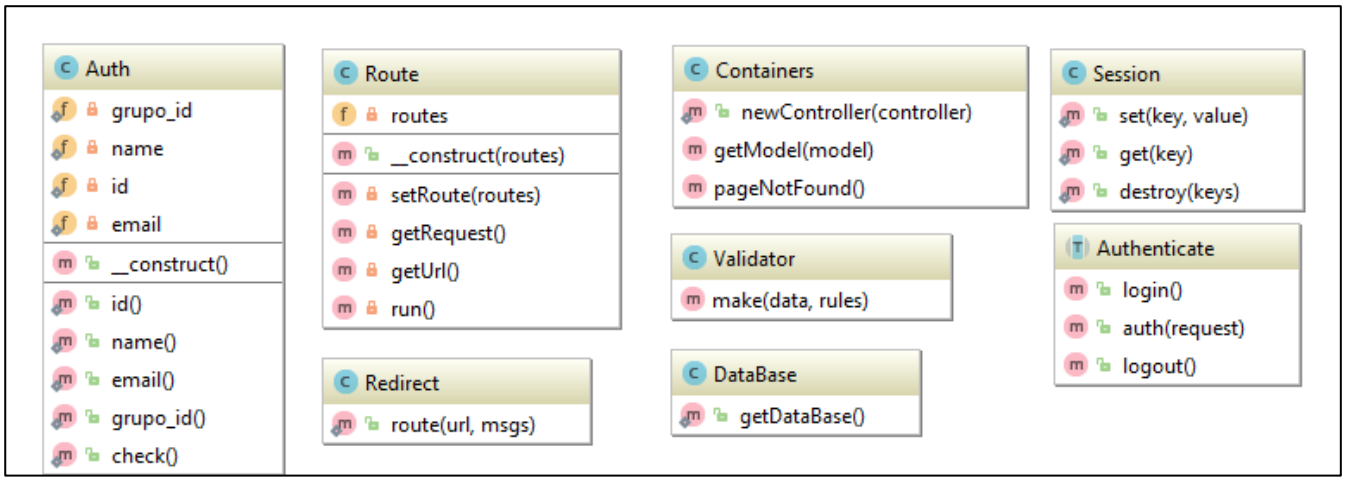


Fonte: Autoria própria.

Na figura 9 tem-se o diagrama com todas as classes da camada *controller* do módulo de Monitoramento da aplicação. Assim como os diagramas da camada *model* apresentados, os diagramas foram divididos entre Gerenciamento e Monitoramento para facilitar a visualização, porém nota-se que a mesma classe “*Controller”* do diagrama da figura 8 aparece de novo neste diagrama da figura 9. Ressalta-se

também a herança dos mesmos atributos e métodos, portanto, com, inclusive, muitos atributos referentes a objetos das classes da camada *model* tanto do módulo de Gerenciamento quanto do Monitoramento. As classes da camada *controller* e *model*, de ambos os módulos, também recebem suporte e requisições de outras classes, chamadas de classes "*Core"*, apresentadas abaixo.

Figura 10 - Classes *Core*



Fonte: Autoria própria.

As classes da figura 10 são classes de suporte e requisições às camadas *controller* e *model*. São apresentadas abaixo suas funções.

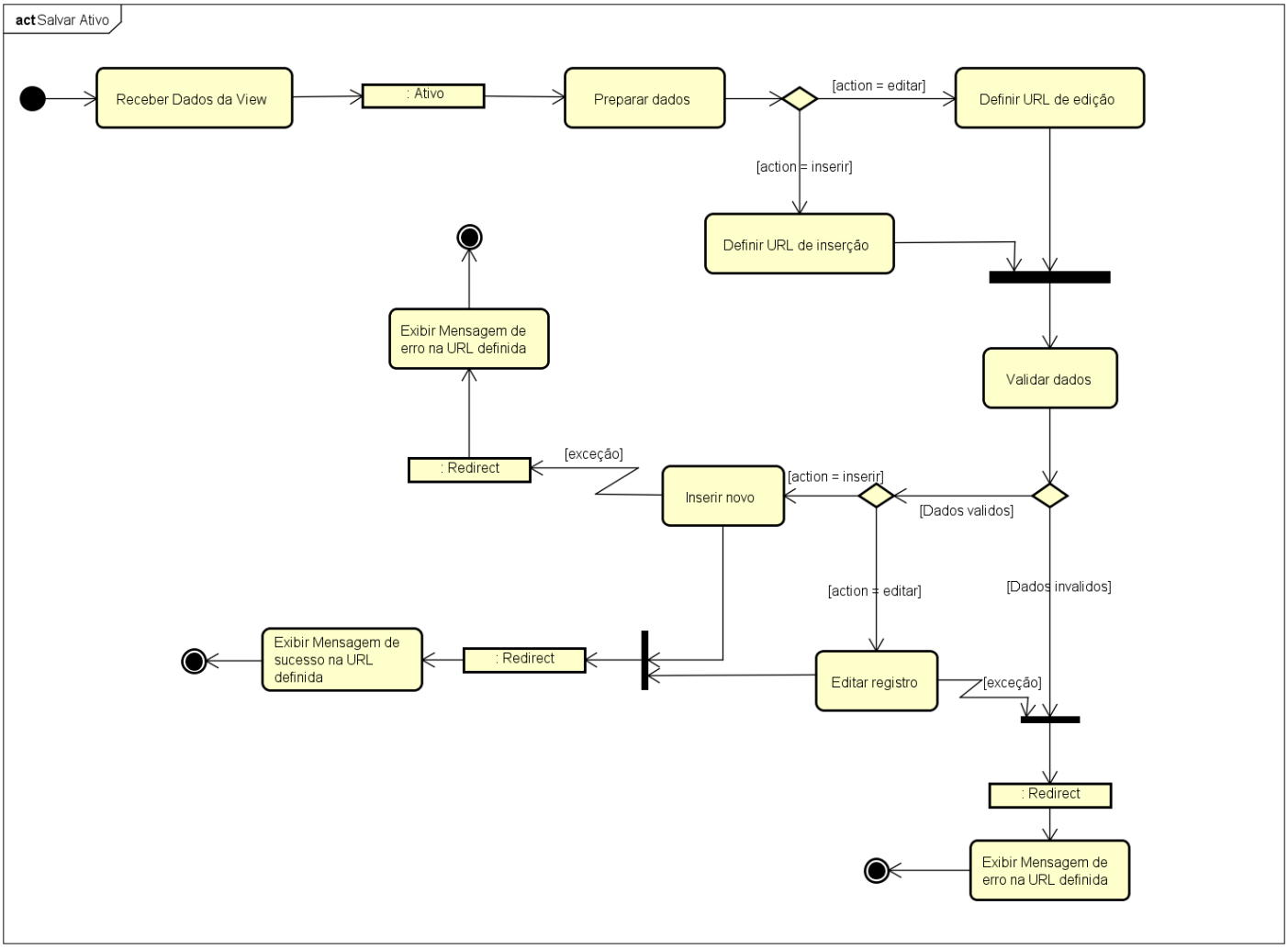
- ***Auth:*** salva os dados do usuário ativo em uma sessão;
- ***Route:*** essa é a classe é especifica para um sistema web, pois faz o tratamento de toda URL requisitada e verifica sua existência e redireciona para a classe *controller* e método adequado;
- ***Redirect:*** redirecionamento e envio de mensagens de uma tela para outra;
- ***Containers:*** abstrai a manipulação de classes *model*, *controller* e requisições incorretas com os métodos "*getModel"*, "*newController"* e *pageNotFound,* respectivamente*;*
- ***Validator:*** valida os dados vindo de formulários (*view*), que são manipulados através do método "*rules"* das classes da *model*;
- ***DataBase:*** manipula a conexão com o banco de dados;
- ***Session:*** abstrai a manipulação da variável *$_SESSION*, nativa da linguagem PHP;

- ***Authenticate:*** manipula as requisições de autenticação fazendo a validação do *login*.

5.4. DIAGRAMA DE ATIVIDADES

O diagrama de atividades é “utilizado, como o próprio nome já diz, para modelar atividades, que podem ser um método ou um algoritmo, ou mesmo um processo completo”. (GUEDES, 2009, p. 285). Utilizou-se deste diagrama para modelagem dos métodos de inserção ou edição e também o de remoção da camada *controller* do sistema. Também se utilizou deste diagrama para a modelagem da atividade de disparo do monitor. A seguir são apresentados os diagramas.

Figura 11 - Diagrama de Atividade Salvar Ativo

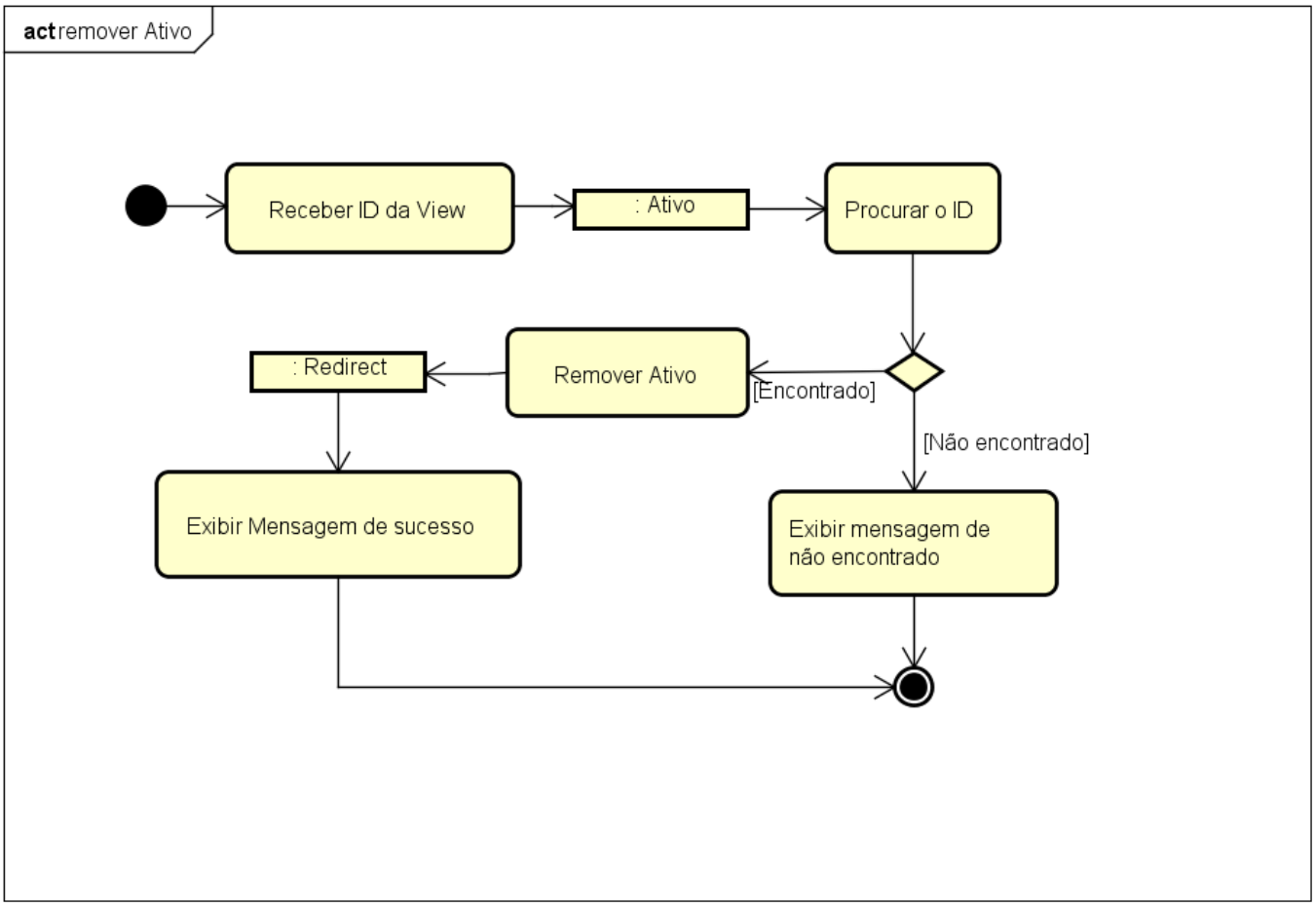


Fonte: Autoria própria.

A figura 11 apresenta o diagrama da atividade do método “salvar” da classe *controller* Ativo. Pode-se notar que os dados são recebidos pela vindos da *view*, tratados pela instancia de um objeto *model* “Ativo”, que prepara, valida e insere um

novo ou editar de acordo com a ação desejada, e depois retorna o valor para a *view* correspondente. Essa atividade se replicará para todos os outros métodos “salvar” de todas as classes da camada *controller*.

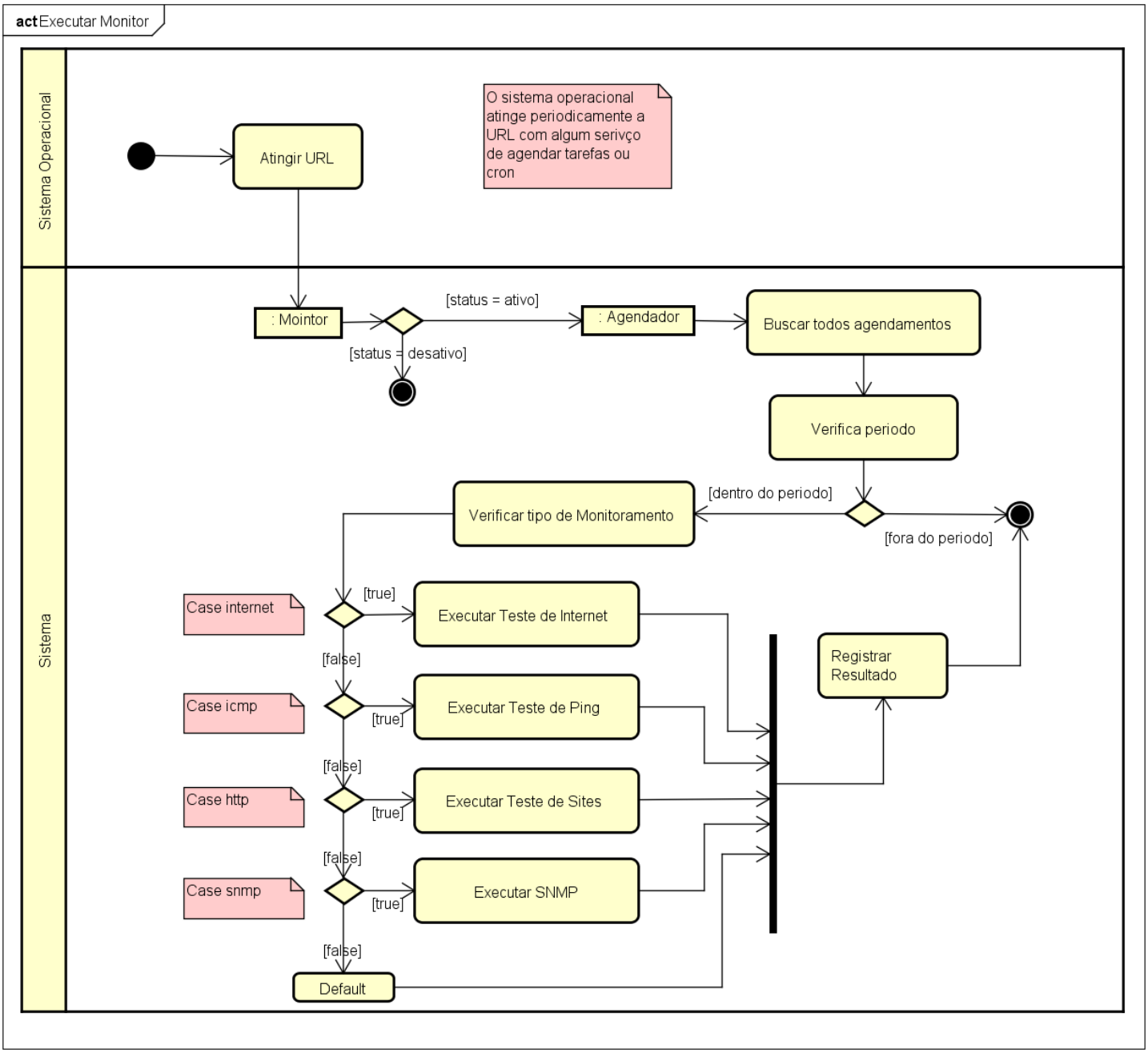
Figura 12 - Diagrama de Atividade Remover Ativo



Fonte: Autoria própria.

Assim como a atividade de salvar e editar, a de remoção, conforme a imagem 12, também é a mesma para todas as classes do *controller*, recebendo o ID através da *view*, instanciando a classe da *model* e enviando o resultado para a *view* novamente.

Figura 13 - Diagrama de Atividades Executar Monitor



Fonte: Autoria própria**.**

Na figura 13 é possível visualizar o diagrama de atividades da execução do monitor, notando que esta atividade começa no sistema operacional, no Agendador de Tarefas do Windows ou na Crontab do Linux, que irá acessar uma URL disponível que iniciará o processo de monitoramento no sistema. Esse acesso precisa ser continuo, para o sistema validar sempre quando estiver no período correto da

execução, e quando ocorrer, após buscar todos os agendamentos, verificar qual o tipo e executar o método que está dentro da *model* da classe de monitoramento.

É importante ressaltar que este diagrama não especifica como cada monitoramento acontece, apenas como cada um é iniciado pela classe “Monitor”, da camada *controller*.

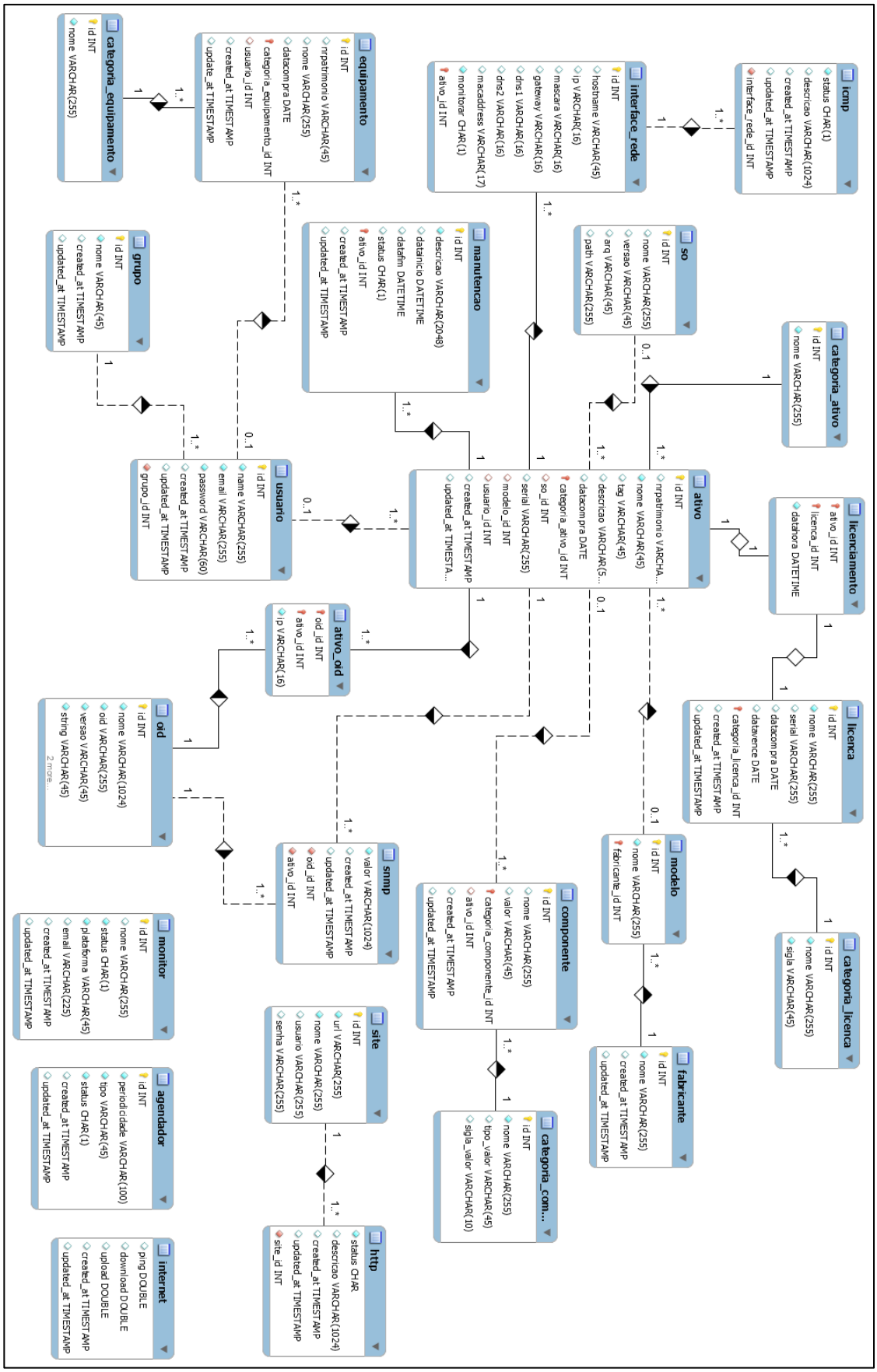
5.5. MODELAGEM DE DADOS

Parte fundamental de todo processo de engenharia de software a modelagem de dados, como o próprio nome sugere, é um processo de criação de modelos que definem entidades, relacionamentos, atributos e ao final como esses dados serão armazenados em um banco de dados físico pelo SGBD (Sistema Gerenciador de Banco de Dados). Pode-se dividir em dois níveis de abstração: modelo conceitual, segundo Heuser (2004, p. 16) “é uma descrição do banco de dados de forma independente de implementação em um SGBD” e modelo lógico que, ainda segundo Heuser (2004, p.17) “é uma descrição de um banco de dados no nível de abstração visto pelo usuário do SGBD”.

A ferramenta mais utilizada para a criação desses modelos é o DER (Diagrama de Entidade e Relacionamento), onde as entidades são representadas por retângulos, e dentro desses descritos seus atributos, e as relações por losangos conectados as entidades por linhas e com indicações de cardinalidade.

Para a modelagem de dados do sistema proposto neste projeto criou-se apenas o modelo lógico, pois utilizou-se a ferramenta de modelagem de banco de dados MySQL Workbench. A ferramenta utilizada criou o banco de dados físico para o sistema a partir do DER, apresentado na próxima página. Mais informações sobre ferramentas utilizadas para o desenvolvimento do sistema são descritas no capitulo 6.

Figura 14 - Diagrama de Entidade e Relacionamento (DER)



Fonte: Autoria própria.

Na figura 14 é possível ver todas as entidades (ou tabelas, quando fisicamente criadas) existentes no banco de dados do sistema e a relação entre elas. Nota-se a centralidade da entidade *ativo*, que representa um ativo de rede dentro do sistema, portanto ela se relaciona diretamente com as entidades *interface de rede*, *sistema operacional*, *manutenção*, *componente*, *modelo*, *usuário* e tem uma relação de muitos para muitos com a entidade *licença*, pois uma licença pode licenciar diversos ativos e um ativos pode possuir diversas licenças.

É importante notar que através dessas relações é possível rastrear um ativo por *grupo* de usuário e *fabricante*, que não se relacionam diretamente com o ativo, porém estão relacionados com entidades diretamente relacionadas ao ativo.

Há ainda algumas entidades que não se relacionam com nenhuma outra, são elas *monitor*, que guarda informações de configuração do servidor da aplicação, *agendador*, que guarda agendamentos de disparos de monitoramento, e *internet*, que guarda os valores de testes da velocidade da internet. Nenhuma dessas mantem ou depende de relacionamento direto com qualquer outra entidade, portanto não tem nenhum atribuído.

Também é importante notar que muitas entidades contêm atributos *created_at* e *updated_at*, uma técnica para manter atualizado a informação de data e hora de criação e de última edição realizada em uma tupla[2] do banco. Nos atributos, utilizou-se a convenção de chamar todas as chaves primarias (identificadoras) de *id* e as chaves estrangeiras com o prefixo *id* mais o nome da tabela de onde vem, como por exemplo *id_ativo* na tabela *interface_rede*.

[2] Tupla: linha de dados de uma consulta ao banco de dados

5.6. ARQUITETURA DE SOFTWARE MVC

Arquitetura de software (no inglês *design pattern*) é uma parte importante no desenvolvimento de qualquer aplicação, pois segundo Engholm Junior (2013, p. 207) “é a estrutura ou estruturas do sistema, que incluem elementos do software, as propriedades externamente visíveis desses elementos e as relações entre eles”. Não é um conceito novo, existem diversos modelos de arquiteturas ligados a *frameworks*, linguagens e técnicas de desenvolvimento, todos levando em conta aspectos específicos na construção de sistemas, como boas práticas, ferramentas de ambiente de desenvolvimento, reaproveitamento de códigos, heranças, classes (ligado a Orientação a Objeto), entre outros aspectos.

Entre eles destaca-se o padrão MVC de *model, view e controller*, ou modelo, visão e controle, que Pitt (2012, p. 1) diz ser “um modelo de arquitetura de software construído ao redor da interconexão de três componentes principais [...] com grande foco em programação orientada a objeto [...]” tradução: autoria própria. Esse modelo divide o desenvolvimento em três camadas separadas, que serão ligadas através da camada controle, o que possibilita a fácil modificação de uma camada ser interferir nas outras, sendo possível substituir toda a parte gráfica da aplicação sem a necessidade de alterações significativas (ou as vezes de qualquer nível) nas camadas do modelo e controle, por exemplo.

A utilização dessa arquitetura trará grandes benefícios ao sistema a ser desenvolvido, como organização de código, reaproveitamento de funções, organização estrutural de camadas, entre outros benefícios. A seguir são detalhadas as camadas do MVC.

5.6.1. *MODEL*

A camada de modelo pode ser definida como “um objeto que representa as informações do domínio de negócios da aplicação” (Dall’Oglio, 2011, p. 477). Pode ser entendido como uma classe, da programação orientada a objetos, que representa algo no mundo físico, como uma pessoa, usuário, produto, ativo, equipamento, peça, etc., e que traz dentro de si todas as regras de manipulação de dados, sendo responsabilidade dessa camada conectar-se com banco de dados diretamente para recuperar ou registrar informações.

Essa camada tem todos os códigos que são necessários para fazer alterações requisitadas pela camada *view* na aplicação e na base de dados. Porém a *model* não decide quando realizar essas ações, ficando essa tarefa de aciona-la depois da requisição da *view* a cargo da camada *controller*.

5.6.2. *VIEW*

Na visão Dall'Olgio (2011, p. 477) explica que "teremos a definição da interface com o usuário, como os campos serão organizados e distribuídos na tela". É a camada de apresentação, que trará os resultados para o usuário a partir das opções que ela também disponibilizará. Essa camada não tem contato com a base de dados, ela se limita apenas a enviar requisições para o controle e receber dele resultados, porém não sabe o que mostra e quando mostrar, quem dirá o que ela deverá exibir e quando será o controle. Para este projeto a *view* será constituída de arquivos HTML, com CSS e *JavaScript*, chamados de *front-end*, que serão descritos com detalhes no capitulo 6 de desenvolvimento.

5.6.3. *CONTROLLER*

Na camada de controle Dall'Olgio (2011, p. 478) ainda diz que "teremos a manipulação dos inputs do usuário, sua interpretação e a execução das tarefas correspondentes". Essa é a camada que fará o "meio de campo" entre a visão e o modelo, ela diz quando deve ser chamada uma visão e quando um modelo deve ser alterado, embora ela não faça alterações.

O controle receberá todas as requisições da visão e iniciará todos os modelos necessários para uma determinada ação, por exemplo, se o usuário quer imprimir na tela uma tabela com dados de funcionários, o controle acionará o modelo funcionários que trará todos os dados dos funcionários e esses dados serão entregues para a visão pelo controle, e a visão exibirá para o usuário aquilo que ele solicitou.

5.6.4. *ROUTER*

O MVC clássico se fecha em torno das três camadas já explicadas, porém com a escolha do PHP como linguagem de desenvolvimento surge uma nova camada, chamada de *router*, ou rota. Essa camada deve sua existência ao fato do PHP ser

uma linguagem para web, onde todos os acessos ao sistema são feitos através de requisições HTTP/HTTPS, ou caminhos URL, e quem define que uma entrada URL vai acessar uma classe *controller* especifica é a classe de rotas.

É possível utilizar a estrutura de arquivos para atingir as classes *controllers*, porém isso expõe os arquivos do servidor, trazendo insegurança, deixa as rotas não amigáveis aos usuários, que teriam que colocar extensão de arquivos, passar argumentos a eles, entre outras más práticas, e acima de tudo, esse método atrapalha a utilização da programação orientada a objeto, obrigado a utilização estrutural do PHP, perdendo todos os benefícios desse método de programação.

Portanto a utilização dessa camada trará maior facilidade na manipulação de todas as requisições ao sistema, e permitirá maior segurança dos arquivos e de acessos a funcionalidades do sistema que não deveriam ser acessadas por usuários não cadastrados.

No próximo capítulo será apresentada a etapa de desenvolvimento do sistema, apresentando conceitos, ferramentas e métodos utilizados.

6. DESENVOLVIMENTO DO SISTEMA

Neste capitulo são abordadas a etapa de desenvolvimento junto as definições técnicas do sistema proposto por este trabalho, abrangendo as escolhas e apresentação das ferramentas, banco de dados, plataforma, linguagens, bibliotecas, *frameworks*, e o motivo da escolha de cada elemento.

O objetivo foi desenvolver um protótipo funcional do sistema, que atenda a maioria dos requisitos levantados no capítulo anterior, sem se aprofundar em todos os detalhes que o desenvolvimento completo de um software apresenta, como níveis de acesso, registro de *logs* de erros ou acessos, entre outros aspectos.

Durante o desenvolvimento do protótipo aplicou-se vários conceitos de desenvolvimento de sistemas, como o MVC apresentado no capítulo anterior e conceitos específicos da linguagem PHP, além de conceitos de infraestrutura para suportar o sistema e o banco de dados, pois o sistema foi idealizado para a utilização dentro de uma organização por um departamento de TI e alguns *stakeholders*, o que traz a necessidade de acessos de vários pontos de um rede a um único repositório de aplicação e dados, conhecido como estrutura cliente-servidor, apresentada neste capítulo.

Muitas pesquisas e estudos foram realizadas para o aprendizado mais aprofundado da linguagem PHP, para obter subsídios suficientes para a execução deste protótipo funcional, entre elas destaca-se a utilização de vídeo aulas no site *YouTube*, em especial do canal TJG Web, que proporcionou grande aprendizagem sobre a criação de *frameworks* da linguagem. Também buscou-se aprender e utilizar outras ferramentas de suporte para simular o servidor em uma máquina de desenvolvimento, ferramentas de versionamento e de modelagem de dados. Um dos grandes desafios foi criar nessa aplicação a independência da plataforma, ou sistema operacional, do servidor da aplicação. Testou-se em um ambiente *Windows*, mas projetou-se para ser funcional em outro ambiente (Linux) com o mínimo possível de alterações.

As bases de lógica de programação, banco de dados, engenharia de software e até mesmo redes de computadores, que foram matérias ministradas em durante o curso tornaram possível a compreensão e desenvolvimento deste protótipo, que pela limitação de tempo, se propôs a atender apenas aos principais requisitos.

6.1. MÓDULOS

O sistema é proposto com dois módulos básicos, que se integram para o correto funcionamento um do outro e para o funcionamento do sistema como um todo. São os módulos de gerenciamento e de monitoramento, sendo que o segundo depende do primeiro para funcionar corretamente, embora essa dependência não seja válida para a relação inversa, podendo-se utilizar o módulo de gerenciamento sem a necessidade de o módulo de monitoramento estar plenamente configurado.

O gerenciamento é um módulo de manutenção (inclusão, edição, exclusão) dos componentes da infraestrutura de TI da organização, onde formam divididos em ativos (de rede), componente, equipamentos, licenças e usuários, para organizar de forma clara os componentes, e por essas categorias conterem ativos de TI que tem características semelhantes. Com esse módulo, a departamento de TI pode ter um inventário organizado e completo de todos os equipamento e ativos importantes para a TI, junto com suas características detalhadas e a forma como eles se relacionam com os outros componentes, simplificando a tarefa de gerenciamento. A seguir a definição para cada categoria criada:

Ativos (de rede): todos os ativos que se conectam à rede, e, portanto, tem uma, ou mais, interface de rede. São eles computadores, notebooks, impressoras, switches, roteadores, entre outros. Todos dessa categoria têm também propriedades de hardware parecidas, como sistema operacional, memoria, processador, unidades de disco, que podem ser atreladas a eles. Além disso, é possível também atribuir modelos e fabricantes de modelos a essa categoria, além de organizar em subcategorias de ativos, para diferençar tipos diferentes. Outra possiblidade é registrar manutenções realizados nesse ativo, uma parte importante do gerenciamento.

Componentes: são definidos como componentes de hardware que podem ser atrelados aos ativos, como memorias, processadores, discos rígido, entre outros. Tem em comum o fato de serem internos a ativos, onde um ativo pode conter diversos componentes, e esses têm valores especiais, como capacidade de processamento, quantidade de memória, etc. Pode-se atribuir uma subcategoria a eles, para melhor organizar, e nela atribuir uma unidade de valor.

Equipamentos: são artigos de hardware externo aos ativos, que estão mais relacionados ao usuário que os utiliza do que ao ativo de rede propriamente dito. São

monitores, teclados, mouses, fones, câmeras, e muitos outros. Seu relacionamento é com usuário, que pode ter em sua responsabilidade muitos equipamentos, e esses podem ser trocados, descartados ou reparados.

Licenças: são permissões para utilizar algum ativo ou sistema. Podem ser organizados por categorias e se relacionam com ativos e usuários, onde cada um pode utilizar várias licenças.

Usuários: ativo humano é uma parte importante do gerenciamento, pois implica na utilização da tecnologia, por isso é possível, no sistema, cadastrar, editar e excluir usuários, e atribuir um acesso para eles no sistema, separando-os por grupos para melhor organizar e localizar os ativos sob sua responsabilidade.

O monitoramento, como o nome indica, é um módulo para monitorar todos os ativos de rede, definidos no módulo de gerenciamento, e, portanto, ele depende que esses sejam cadastrados e que tenha interfaces de rede configuradas para se fazer o monitoramento. Definiu-se no cápitulo 4 os protocolos de rede utilizados no monitoramento desses ativos, porém nem todos os ativos podem ser monitorados pelos três protocolos (ICMP, HTTP/HTTPS e SNMP), por exemplo, um ativo que não tem suporte SNMP não será monitorado pelo mesmo, ou um ativo que não tem uma interface web não será monitorado pelo HTTP/HTTPS. Abaixo os tipos de monitoramentos e como funcionam:

Monitoramento ICMP (*ping*): Monitora as interfaces de rede cadastradas, através do IP utilizando o comando *ping*. As configurações deste tipo de monitoramento permitem definir qual interface será monitorada e qual não, e a periodicidade dos disparos.

Monitoramento HTTP/HTTPS: Para este monitoramento é necessário cadastrar sites, definido uma URL completa para cada, e estes serão verificados, se estão acessíveis ou não. É possível configurar a periodicidade dos disparos.

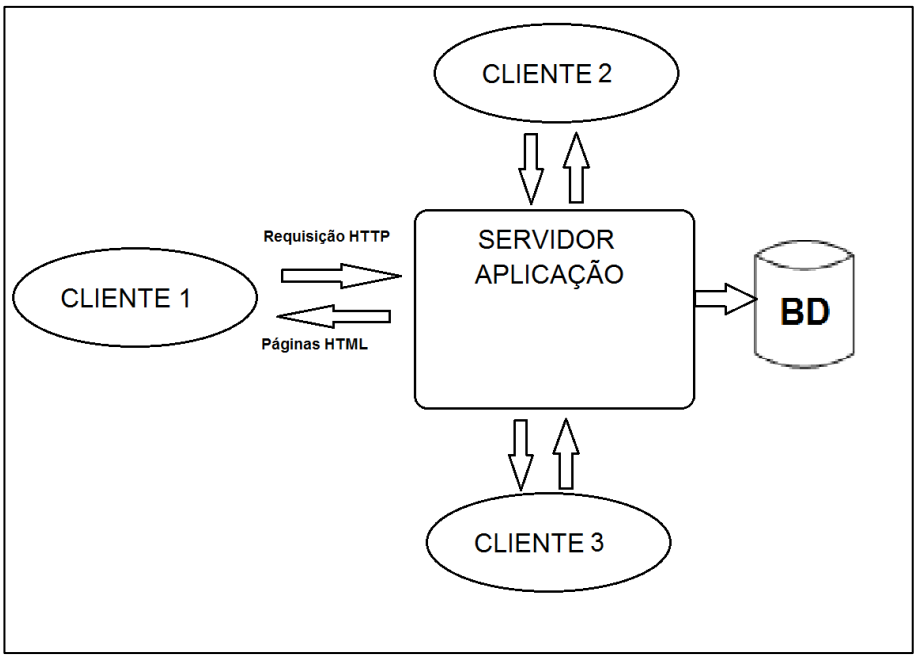
Monitoramento SNMP: É necessário cadastrar identificadores OID e relaciona-los com os ativos, para que o sistema requisite as informações que cada identificador possibilita. É possível configurar a periodicidade dos disparos.

Monitoramento da Velocidade Internet: Este monitoramento não é feito por nenhum protocolo de rede especifico, porém, o módulo conta com um recurso de monitoramento de *download*, *upload* e *ping* da conexão com a internet. É possível configurar a periodicidade dos disparos.

6.2. BACK-END E FRONT-END

Para atender aos requisitos de ter informação sempre à disposição, e ao mesmo tempo, organizada e centralizada em um único repositório, para evitar redundância de dados e informações imprecisas, e para possibilitar que vários equipamentos na rede possam acessar a mesma informação, utilizou-se uma estrutura cliente-servidor, onde vários clientes podem fazer requisições a um único servidor, que concentra um serviço, no caso uma aplicação e uma base de dados como mostra a figura 15. Dessa forma, todo o armazenamento, processamento e disponibilização da informação é feita pelo servidor da aplicação.

Figura 15 – Estrutura Cliente Servidor.



Fonte: Autoria própria.

O servidor da aplicação é “um programa executado em uma máquina classificada como servidora capaz de interpretar requisições recebidas de maquinas clientes” (COSTA; TODESCHINI, 2006, p. 101), já os clientes podem ser quaisquer máquinas que conseguem acessar o servidor de aplicação.

Há muitos benefícios nessa estrutura de servidor de aplicação. Ganha-se rapidez de processamento nas requisições pelo fato de todo processamento ser feito por uma máquina servidor, que tem uma capacidade maior para isso, enquanto os clientes não precisam processar basicamente nada mais que uma exibição da informação, nem necessitam de espaço em disco para armazenar um banco de

dados. Essa vantagem relaciona-se diretamente ao custo, pois não é necessário que os clientes tenham nada mais que um navegador web para acessar a aplicação e seus dados.

Outra vantagem é a facilidade de acesso, pois não existe necessidade de instalar a aplicação nos clientes, o próprio servidor também provê toda a aplicação. Isso também traz compatibilidade com vários dispositivos e sistemas operacionais, que por padrão já tem um navegador.

Algumas variáveis podem fazer essas vantagens não acontecerem, como uma rede de pouca capacidade de trafego de dados, um servidor com pouca capacidade para atender várias requisições, perda de conexão com o servidor; porém todas essas variáveis são mais fáceis de serem manipuladas quando estão relacionadas a um único servidor ao invés de várias aplicações funcionando em vários computadores, e assim outra vantagem dessa estrutura é a facilidade maior de detectar e resolver eventuais problemas na aplicação centralizada.

Portanto, para este projeto decidiu-se utilizar a estrutura de *back-end* e *front-end*, onde está o servidor de aplicação e a base de dados e a visualização da informação para o cliente, respectivamente. Na estrutura de um servidor web o servidor, que é o *back-end*, armazena todos os arquivos necessários para o cliente, onde está o *front-end*, além de toda a programação e base de dados, cabendo ao cliente apenas receber, e a partir de um navegador acessar as informações.

Nos próximos subcapítulos serão apresentados os componentes do *back-end* deste projeto.

6.3. LINGUAGEM DE PROGRAMAÇÃO PHP

Tendo definindo que o sistema funcionará com a estrutura cliente-servidor, para o *back-end* a linguagem de programação escolhida foi PHP, muito utilizada para sistemas baseados em servidores web. O “PHP é uma linguagem de programação desenvolvida para gerar páginas da web interativamente no computador que as serve, que é chamado de servidor web. ” (DAVIS; PHILLIPS, 2008, p. 2). A linguagem é um projeto *open source*, ou seja, de código fonte aberto, onde qualquer pessoa pode adicionar funcionalidades novas a ela. Atualmente está em sua versão 7.1.4.

Como uma linguagem que foi pensando na estrutura de servidor web ela apresenta várias facilidades para o desenvolvedor em um ambiente de rede, com funções próprias, inclusive funções que serão utilizadas para o módulo de monitoramento, que manipulam os protocolos SNMP e HTTP de forma a fornecer um meio de testar esses protocolos de forma nativa à linguagem, e ainda uma função de execução de linha de comando no sistema operacional, possibilitando a utilização do comando *ping* do protocolo ICMP, em qualquer sistema operacional do servidor de aplicação.

É uma linguagem de rápido processamento, não exigindo muito recurso do servidor, e também de rápido desenvolvimento, pois existem muitas fontes e materiais didáticos, além das próprias funções prontas da linguagem, que permitem agilidade nos projetos.

Outra vantagem segundo Davis e Phillipes (2008, p.30) é que “usuários avançados têm a habilidade de mudar o código fonte, e portanto, mudar a maneira com que a linguagem e programa funcionam”. Além de tudo isso o fato de ser projeto de código aberto significa que não há necessidades de licenças ou alguma compra para utilizar as ferramentas.

Todas as vantagens do PHP fazem dessa linguagem a preferida de muitos desenvolvedores que precisam de rapidez no desenvolvimento e não dispõem de muitos recursos financeiros para gastar com licenças de software.

Neste projeto decidiu-se, sem dúvidas, utilizar essa linguagem para todos os processamentos necessários no servidor, inclusive todas as funções de monitoramento. Para o desenvolvimento na linguagem será utilizada a IDE PhpStorm da empresa JetBrains, com licença de estudante, e em alguns casos o editor de texto Sublime Text 3.

6.4. SERVIDOR WEB APACHE

Para o servidor web funcionar com o PHP é necessário o Apache. Segundo Davis e Phillips (2008, p. 4) “Apache é um servidor web que transforma requisições dos navegadores em resultados em páginas da web e sabe como processar o código PHP”. Também é um projeto de código aberto (*open source*) e sem ele não seria possível utilizar o PHP, já que ele liga o cliente que faz requisições e a linguagem no servidor que processa, através de módulos próprios, um deles exclusivo para o PHP, além de armazenar todos os arquivos do *front-end* que será enviado ao cliente.

Como um projeto de código aberto, tem todas as vantagens que isso proporciona, principalmente para usuários avançados que manipulam o código fonte para sua utilização. Conta também com grande material didático e várias comunidades na internet.

Existem outras opções para substituir o Apache, mas ele tem a vantagem de não ter custo, e ter uma grande compatibilidade com sistemas operacionais diversos, o que permite que o servidor possa utilizar tanto um sistema operacional Linux como um Windows.

Para o desenvolvimento do protótipo utilizou-se a ferramenta XAMPP versão 3.2.2, que provê o serviço Apache e o MySQL, em um sistema operacional Windows 10 Professional versão 1709, o que não interfere em qual sistema a estação cliente utilizará, todos conseguirão acessar e fazer requisições ao servidor da mesma forma. Este ambiente de desenvolvimento pode ser considerado um ambiente de teste, pois a ferramenta foi idealizada para funcionar tanto em servidor Windows quanto um Linux, porém apenas foi testada neste ambiente.

6.5. MySQL

Em uma estrutura cliente-servidor toda a base de dados se encontra no servidor, o que permite uma centralização dos dados, onde todos os clientes podem acessar uma única e confiável fonte de dados. Conhecido como servidor de banco de dados, é uma peça importante para qualquer projeto de sistema de informação.

Para cumprir esse papel, a ferramenta mais utilizada junto ao PHP e Apache é o banco de dados MySQL Server, outro projeto *open source*, que atualmente é mantido pela empresa Oracle e que está na versão 5.7.

Existem muitas vantagens em utilizar esse servidor de banco de dados, como por exemplo sua fácil integração com o PHP, segundo Davis e Phillips (2008, p. 2) “[...] foram desenvolvidos um tendo o outro em mente, portanto são fáceis de se utilizar em conjunto. A programação relaciona entre eles é pareada logicamente. ”

Além dessas vantagens, ele tem todas as mesmas vantagens do PHP e Apache, como rapidez, suporte da comunidade, e licença gratuita, o que faz dele ideal para o desenvolvimento de sistema em servidor web.

Visto suas vantagens para o projeto, decidiu-se usá-lo como servidor de banco de dados, e utilizar a interface gráfica de usuário PhpMyAdmin, uma ferramenta gratuita que permite acessar o servidor MySQL de qualquer navegador web e fazer todo o tratamento e manipulação de dados, e que é especifica para usar com MySQL e PHP em desenvolvimento web. Também utilizou-se a ferramenta desktop MySQL Workbench, que possui um módulo de desenho de DER (Diagrama de Entidade e Relacionamento), e partir desse gerou o banco de dados no servidor MySQL.

O MySQL Server finaliza o tripé do *back-end*, que tem como suas bases o servidor web Apache, o servidor de banco de dados MySQL e a linguagem de programação PHP. A partir da próxima seção será apresentado as definições do *front-end* do sistema.

6.6. HTML, CSS e *JavaScript*

O *front-end* é conhecido como o lado do cliente, ou seja, é a parte da programação que é apresentada ao usuário, por exemplo uma tela, mensagem de sucesso ou erro ou qualquer outro elemento que se apresente como uma interface para quem utilizar o sistema que estará no *back-end*, em um servidor. Um *front*-end web é constituído de três linguagens básicas, são elas o HTML, CSS e *JavaScript.*

- **HTML**

HTML é uma linguagem de marcação, seu nome pode ser traduzido como Linguagem de Marcação de Hipertexto, e segundo Duckett (2010, p. 26) “HTML e XHTML são necessárias para explicar a estrutura de quaisquer páginas web. [...] pode considerar XHTML mais como a versão recente do HTML”. Portanto, o HTML é uma base estrutural de uma página web, descrevendo, ou marcando, os elementos que esse documento contém. Sem essa base não há como os navegadores desenharem as telas, que são interpretação das marcações, porém sem complementos as telas seriam apenas linhas e estruturas, é por isso que é utilizado o CSS para dar aparecia ao documento HTML.

- **CSS**

CSS (*Cascading Style Sheets*) é um documento de estilos, atualmente na versão CSS3. Segundo Duckett (2010, p. 256) “CSS trabalha permitindo a você associar regras aos elementos que aparecem no documento. Estas regras governam como o conteúdo desses elementos devem ser exibidos”. É possível definir cores, transparências, tamanhos, direções do texto, alinhamentos e muitas outras definições. Sem esses estilos, as telas web seriam certamente menos atrativas ao usuário e mais difíceis de controlar para o desenvolvedor, pois ele teria de aplicar regras em cada marcação, ao invés de criar regras para todo um tipo de marcação do documento.

- ***JavaScript***

Uma interface de usuário com telas apenas em HTML e CSS seria muito estático, e existem necessidades que essas duas linguagens, que são de marcação e

estilo, não linguagens de programação, não poderão atender. Elas se limitam as telas, não a criação de *scripts* ou pequenos códigos que possibilitem tratamento e processamento de informação. Para isso é utilizado a linguagem *JavaScript*, que fará o papel de uma programação intermediaria, feita no computador cliente, e que auxiliará a linguagem final, no *back-end* a apresentar dados, validar informações, editar o próprio documento HTML, tudo no *front-end*, ou seja, no lado do cliente.

Segundo diz Flanagan (2013, p. 19) “o *JavaScript* é a linguagem de programação da Web. A ampla maioria dos sites modernos usa *JavaScript* e todos os navegadores modernos [...] incluem interpretadores *JavaScript [...]*”. Por esse fato, essa linguagem é compatível com qualquer sistema operacional, trazendo um enorme benefício pela facilidade na qual vários equipamentos acessam a interface e seja possível realizar e executar procedimentos lógicos sem a necessidade de uma linguagem de programação especifica para cada equipamento.

Para o sistema, com a escolha de se utilizar a estrutura de cliente-servidor, não há dúvidas que o tripé do *front-end* HTML, CSS e JavaScript seja a melhor opção para o tratamento da interface. No próximo capitulo serão abordadas bibliotecas *front-end* utilizadas para facilitar o desenvolvimento da interface.

6.7. BIBLIOTECAS, *FRAMEWORKS FRONT-END*

Com a expansão da web e as necessidades de agilidade no processo de criação de interfaces utilizando as linguagens HTML, CSS e JavaScript, foram desenvolvidas várias ferramentas para facilitar o processo de construção de páginas web. Essas ferramentas são como pacotes prontos, para utilizar nos projetos de forma fácil e ágil, e ainda padronizar, sempre utilizando as melhores técnicas, as interfaces, tornando mais agradável ao usuário que a utilizará. São conhecidas como bibliotecas e *frameworks*, onde cada uma serve a um objetivo.

Bibliotecas são códigos prontos que executam uma função. Em termos simples, são para serem referenciar em uma aplicação e utilizar seus recursos, facilitando o trabalho de programação, suprindo a necessidade de se codificar funções que já foram implementadas e amplamente testadas.

Frameworks são um ambiente maior, com várias bibliotecas reunidas, ou seja, várias funções prontas para se utilizar, que fazem com que as aplicações feitas a partir desses sejam, mesmo que apenas em seu núcleo, padronizadas e mais seguras, evitando os erros que novos códigos não testados podem ter.

Utilizar essas ferramentas trará agilidade, segurança, prevenção de erros, padronização, entre outros benefícios para a ferramenta proposta. A tabela 1 traz uma lista de bibliotecas e *frameworks* que serão utilizadas no sistema, todos eles são *open source*, ou seja, de código aberto, e estão disponíveis para utilização nos sites de seus criadores.

Tabela 1 – Bibliotecas e Frameworks.

Nome da Ferramenta	**Descrição**
Bootstrap	Framework para desenvolvimento *front-end* responsivo.
jQuery	Biblioteca que abstrai a linguagem JavaScript facilitando sua utilização
DataTables	Biblioteca jQuery para estilização de tabelas, permitindo paginação, pesquisa e outros recursos.
AmCharts	Biblioteca JavaScript para geração de gráficos (barra, pizza e linhas) e mapas.

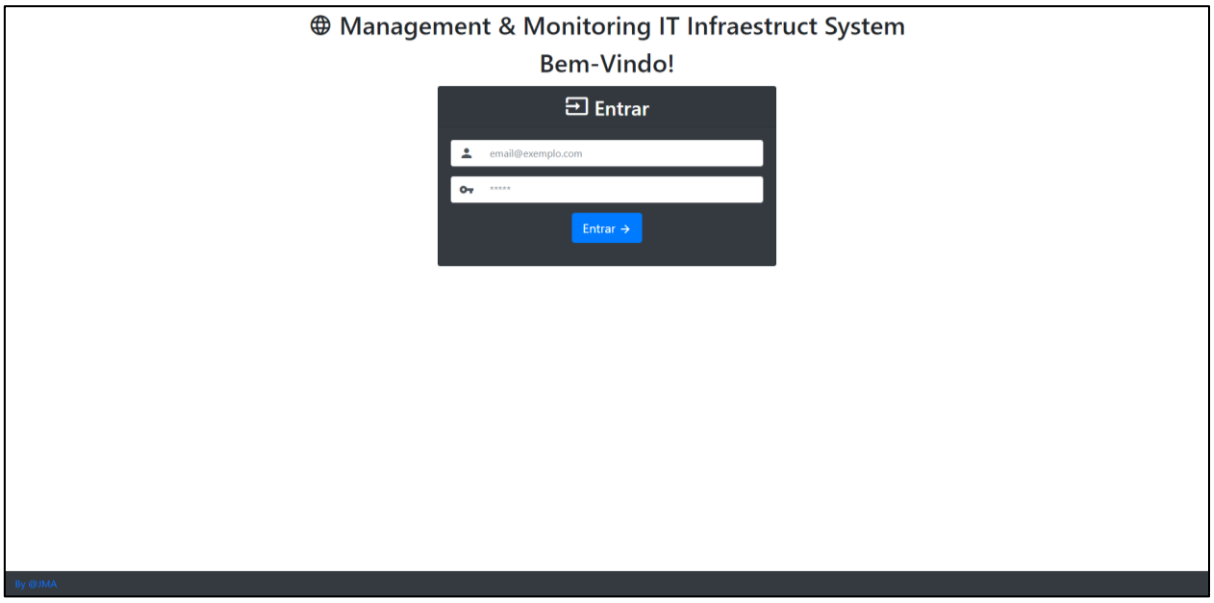
Fonte: Autoria própria.

6.8. TELAS DO SISTEMA

Neste subcapítulo são apresentadas as principais telas e as funcionalidades já implementadas no protótipo do sistema proposto, desenvolvido para atender os requisitos gerais definidos no capitulo 5. Os dados apresentados foram inseridos, alterados e recuperados diretamente do banco de dados da aplicação.

Procurou-se apresentar apenas as telas principais, pois elas representam o molde geral de todas as outras telas, evitando apresentação de todas as telas que contém as mesmas funcionalidades gerais.

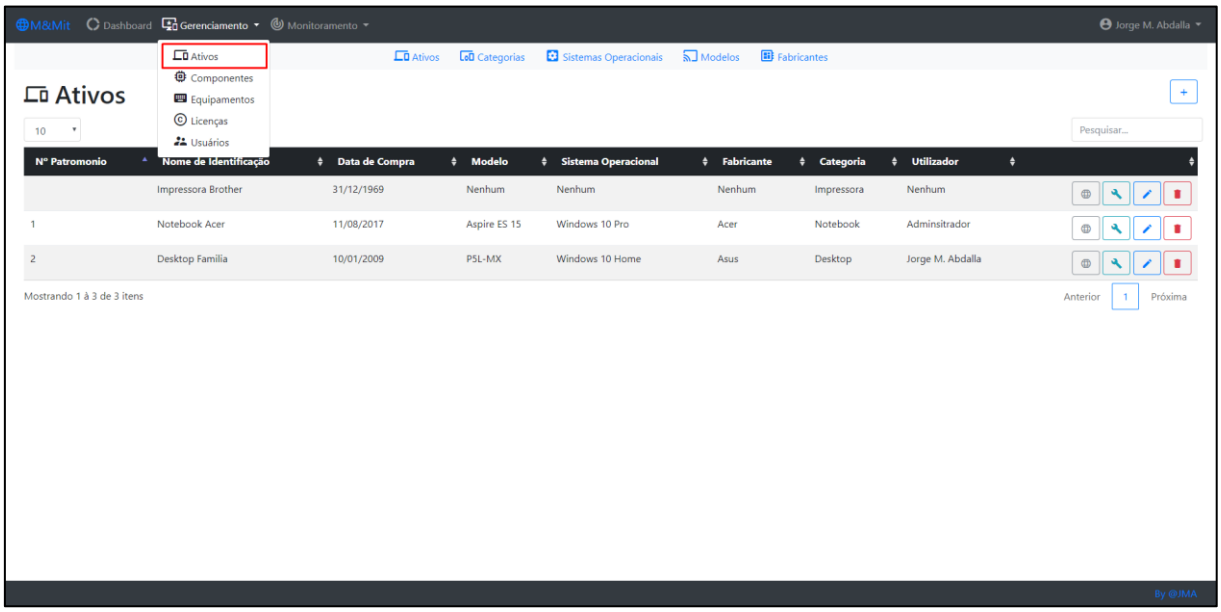
Figura 16 - Tela de *Login*



Fonte: Autoria própria.

A figura 16 apresenta a tela de *login* do sistema. Para entrar no sistema o usuário precisa utilizar uma combinação de e-mail e senha validados cadastrados no banco de dados. Nenhuma URL é acessível caso o usuário não tenha feito o *login*.

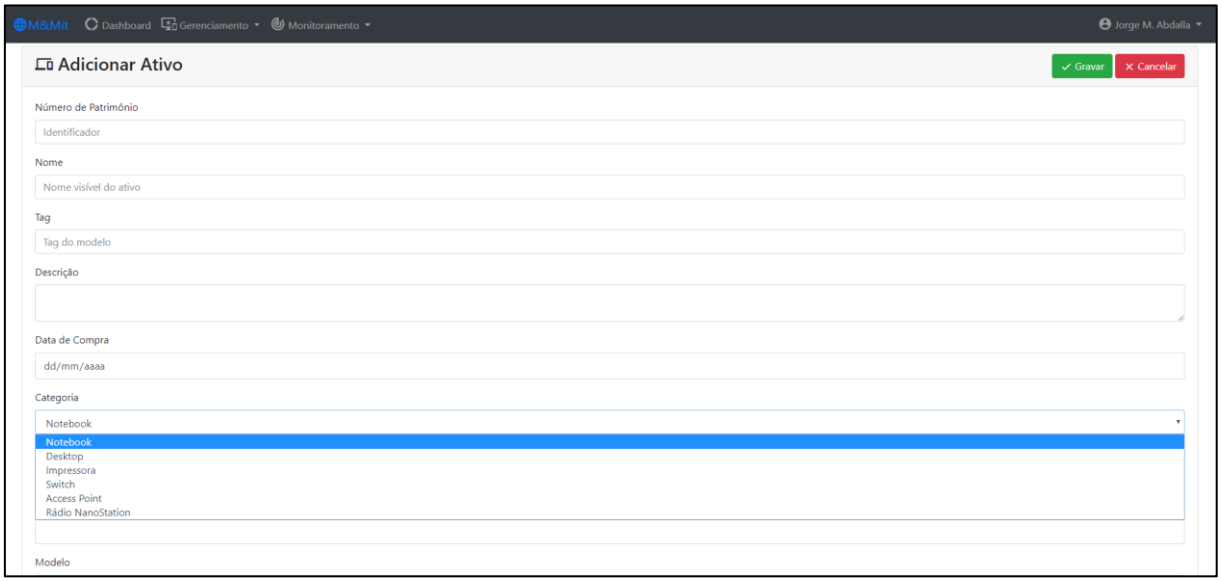
Figura 17 - Tela de Ativos



Fonte: Autoria própria.

A figura 17 apresenta a tela com todos ativos, com destaque vermelho para o item do menu superior para esta tela. Nesta tela é possível adicionar novos ativos, registrar manutenções, interfaces de rede, editar e deletar ativos. Além disso, é possível navegar pelo menu inferior e acessar as telas de cadastros de categorias de ativos, sistemas operacionais, modelos de ativos e fabricantes desses modelos.

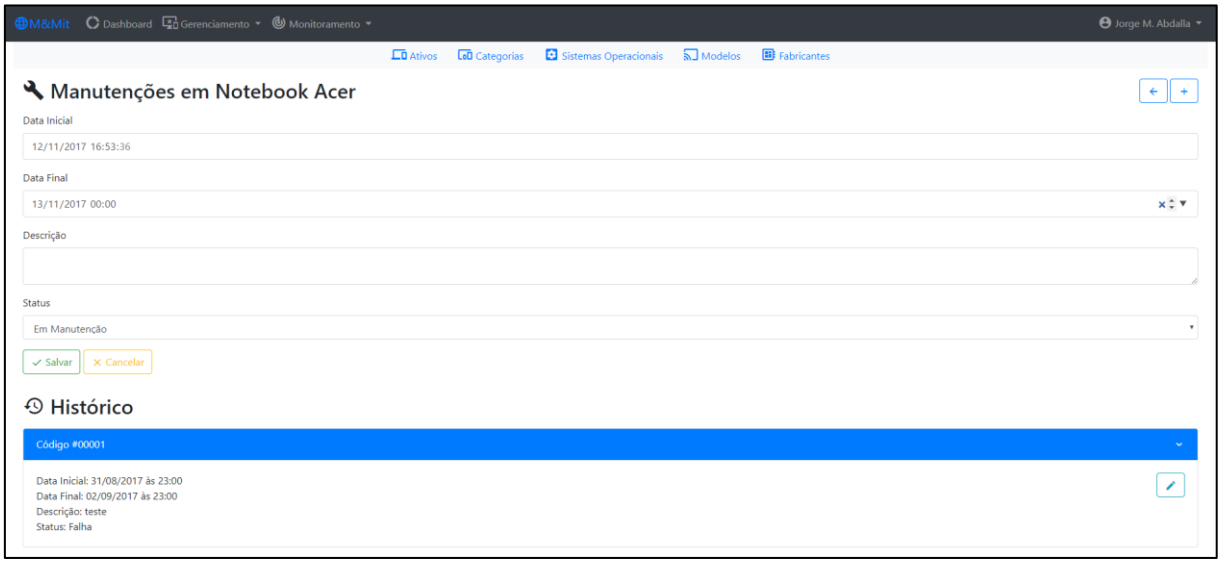
Figura 18 - Tela de Cadastro de Ativo



Fonte: Autoria própria.

A figura 18 apresenta o formulário de cadastro de um novo Ativo. Entre as informações destaca-se o campo “Categoria”, que traz informações cadastradas em “Categorias de Ativos”.

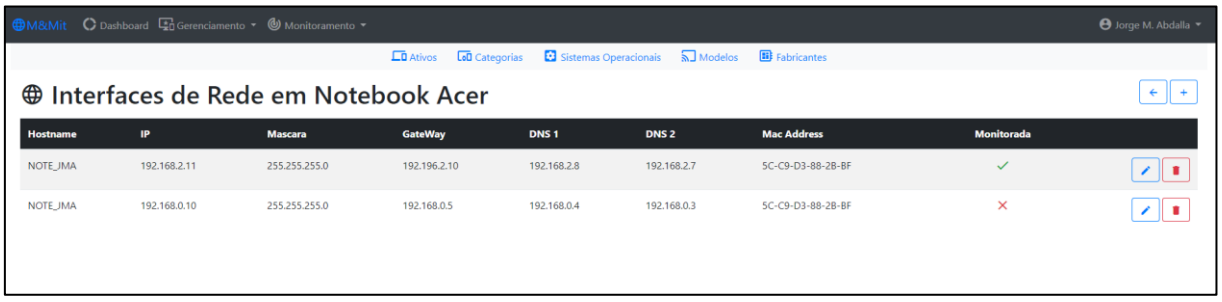
Figura 19 - Tela de Manutenção de Ativos



Fonte: Autoria própria.

Para registrar manutenções é necessário clicar no ícone de manutenção na tabela de ativos, na figura 19 para chegar a tela de manutenções da figura 16, onde é possível ver o histórico de manutenções do ativo ou registrar uma nova.

Figura 20 - Tela de Interfaces de Rede

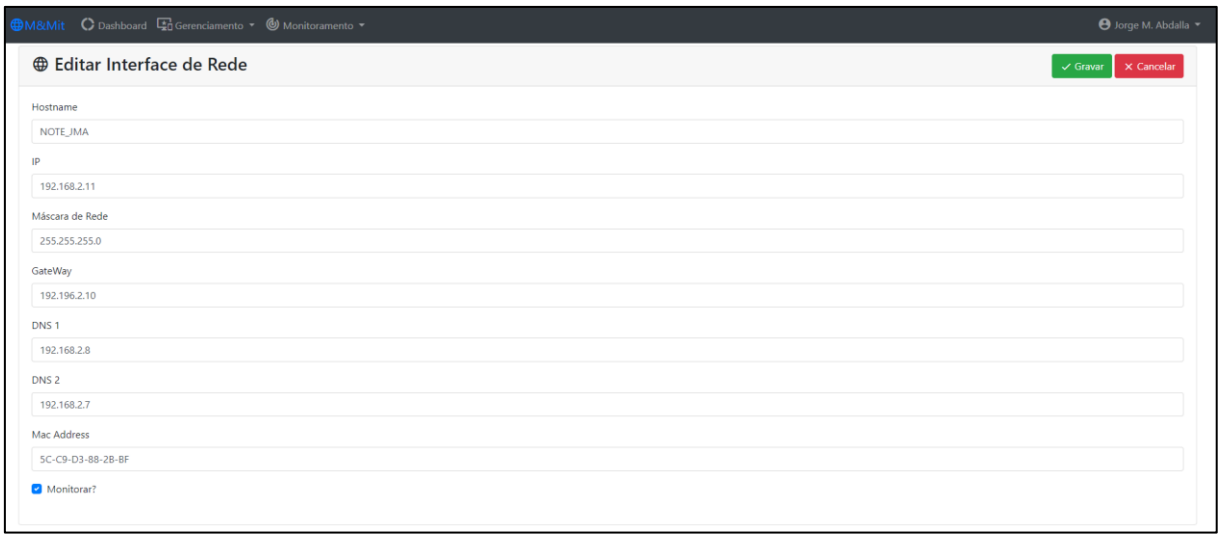


Fonte: Autoria própria.

Da mesma forma, para acessar e registrar interfaces de rede é necessário clicar no ícone de interface de rede no ativo desejado na tabela de ativos da figura 17 e na

tela de interfaces de rede da figura 20 ver as interfaces registradas, registrar uma nova ou editar alguma.

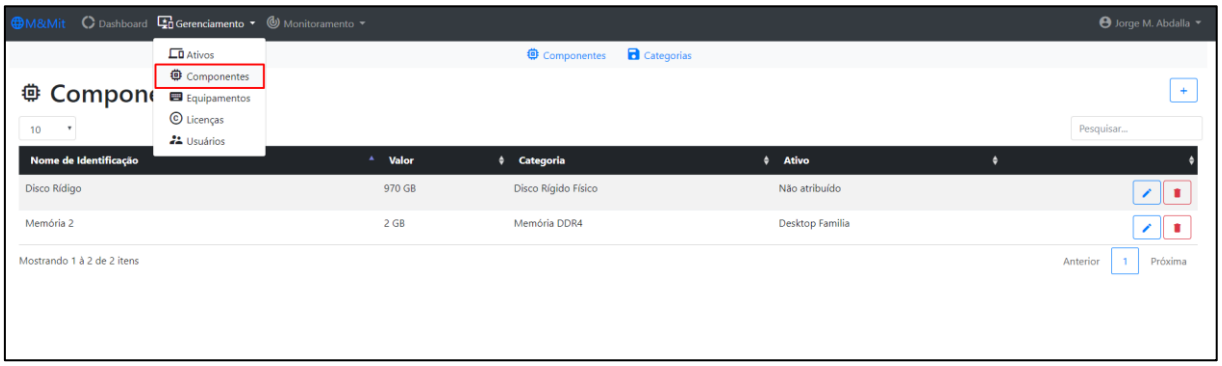
Figura 21 - Tela de Cadastro de Interface de Rede



Fonte: Autoria própria.

Ao cadastrar ou editar uma interface, como na figura 21, é possível definir informações que serão pertinentes ao monitoramento, e definir se essa interface de rede será ou não monitorada pela opção “Monitorar? ”.

Figura 22 - Tela de Componentes

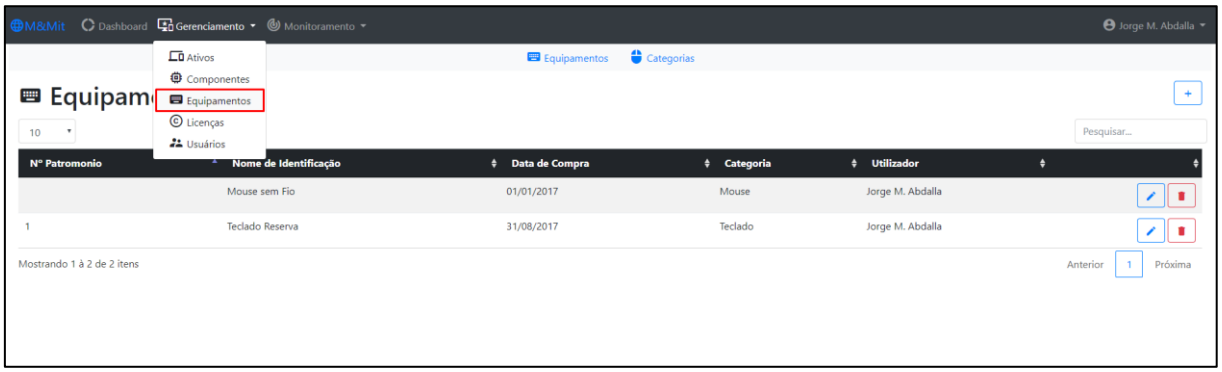


Fonte: Autoria própria.

Na figura 22 tem-se a tela de componentes, onde são cadastrados, editados e excluídos os componentes internos de um ativo de rede. Em destaque o item de menu para acessar essa tela. Os componentes também podem ser classificados em categorias, e nessas pode-se inserir unidades de valor do componente (ex.: Giga-

hertz, Megabyte) e siglas para essas unidades (ex.: GHz, MB). Essas informações serão inseridas nos campos da categoria de componentes.

Figura 23 - Tela de Equipamentos

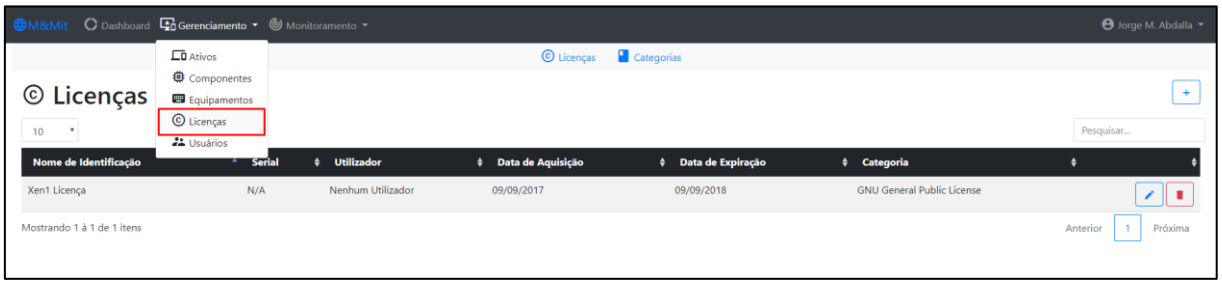


Fonte: Autoria própria.

Além dos componentes internos do ativo, há os equipamentos externos que ajudam na utilização desses, porém estão relacionados ao utilizador (usuário no sistema), ao invés do ativo. Na figura 23 tem-se a tela de cadastro, edição e exclusão de equipamentos, em destaque para o item do menu de acesso a essa tela.

Os equipamentos também podem ser categorizados, para facilitar a busca por informações através de classificação, acessando a tela das categorias no menu inferior.

Figura 24 - Tela de Licenças

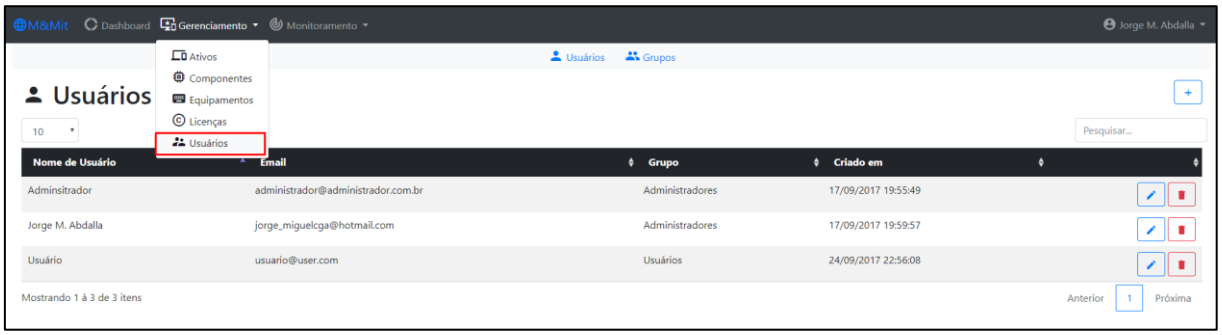


Fonte: Autoria própria**.**

A figura 24 mostra a tela de licenças, onde são cadastradas, editadas ou excluídas licenças de software, equipamentos, usuários, ativos, entre outras. Essas licenças podem ser atreladas a ativos cadastrados no sistema ou podem apenas ser cadastradas para consulta e arquivamento. Também é possível cadastrar, editar e

excluir categorias de licenças, na tela de categorias. Essa classificação facilita a busca e recuperação de informações classificadas por categorias.

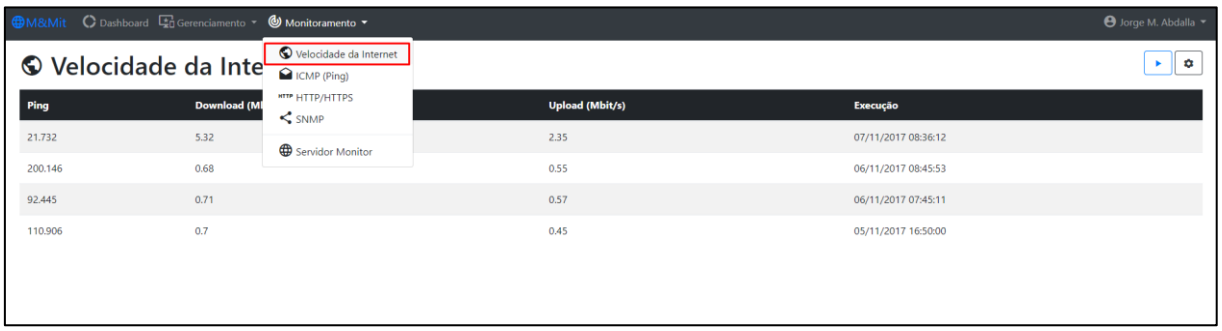
Figura 25 - Tela de Usuários



Fonte: Autoria própria.

A figura 25 mostra a tela de cadastro, edição e exclusão de usuários, onde são atribuídos acessos ao sistema através de e-mail e senha cadastrados. Esses usuários também são relacionados a ativos e equipamentos, nas respectivas telas desses itens. Os usuários podem ser classificados em grupos para melhor organizar tipos de usuários ou classificar por departamentos em uma organização. Ainda não há implementação de níveis de acesso no protótipo, mas essa informação também será útil para a criação desses níveis.

Figura 26 - Tela de Monitoramento da Velocidade da Internet



Fonte: Autoria própria.

A primeira tela do módulo de monitoramento é a da “Velocidade da Internet”, acessada pelo item de menu em destaque na figura 26. Nesta tela são apresentados o histórico de testes, com informações de *ping*, *download* e *upload* e data e hora da

execução do teste. Nesta tela é possível realizar um disparo fora do agendado, como o botão azul na parte direita superior da tela.

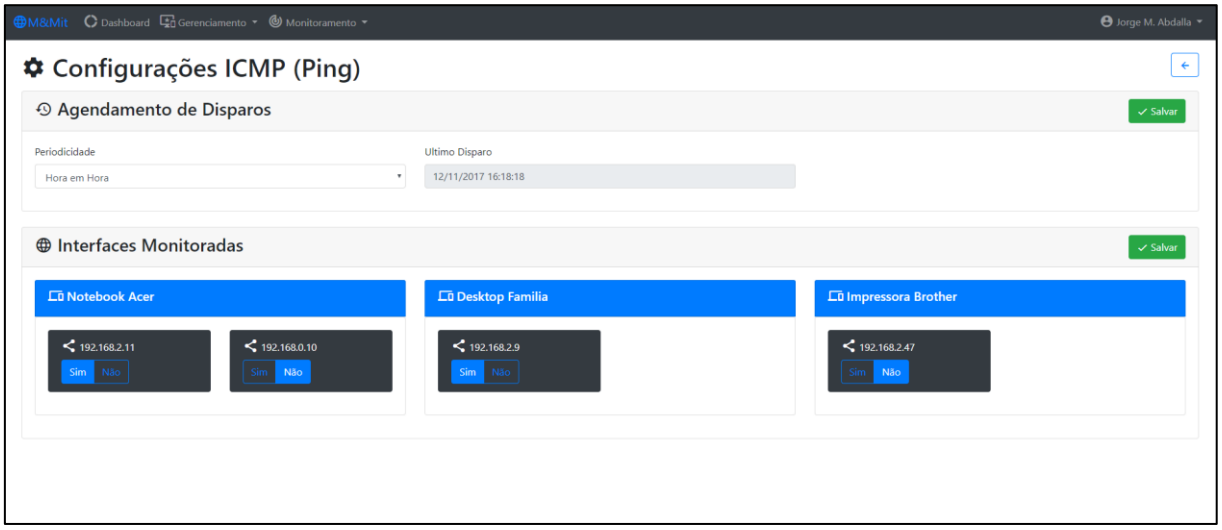
Clicando em configurações é possível configurar o agendamento de disparos do teste de internet e habilitar ou desabilitar esse teste.

Figura 27 - Tela de Monitoramento ICMP

Fonte: Autoria própria.

Na tela apresentada na figura 27 tem-se as últimas execuções do monitoramento pelo comando *ping* de cada interface de rede monitorada. Em destaque para o item do menu de acesso a essa tela e o “X” vermelho de inacessível. Caso a interface esteja acessível, aparecerá um símbolo de *check* verde de acessível.

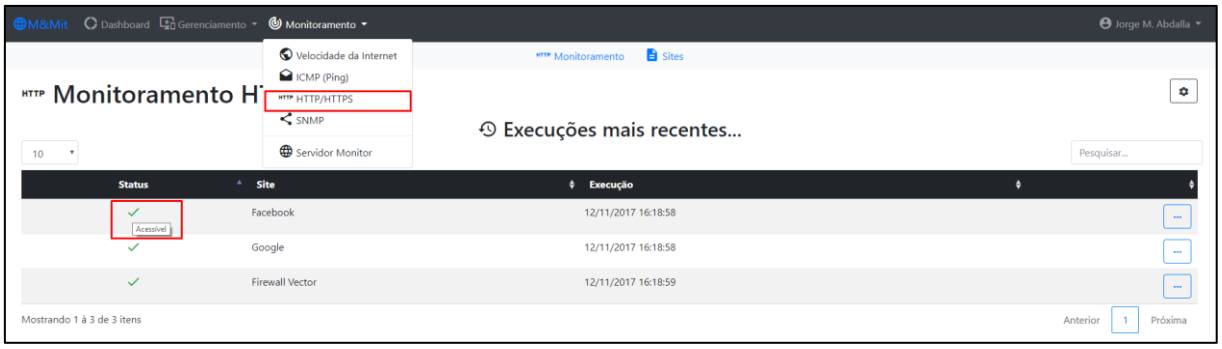
Figura 28 - Tela de Configuração do Monitoramento ICMP



Fonte: Autoria própria.

A figura 28 apresenta a tela de configuração do monitoramento ICMP, onde além de configurar a periodicidade dos disparos também é possível configurar quais interfaces de rede de cada ativo são monitoradas. Mesmo que um ativo tenha mais de uma interface, pode-se configurar para monitorar apenas qual se deseja, não sendo necessário monitorar todas.

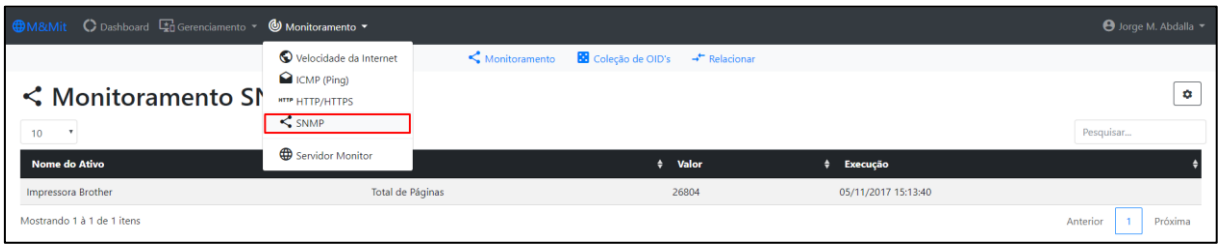
Figura 29 - Tela de Monitoramento HTTP/HTTPS



Fonte: Autoria própria.

A figura 29 apresenta a tela com as execuções mais recentes do monitoramento de sites HTTP/HTTPS, com destaque para o item de menu de acesso a tela e o símbolo de *check* verde, indicando sucesso no teste. Também é possível configurar a periodicidade. Para este monitoramento é necessário cadastrar sites, na tela de sites clicando no menu inferior.

Figura 30 - Tela de Monitoramento SNMP

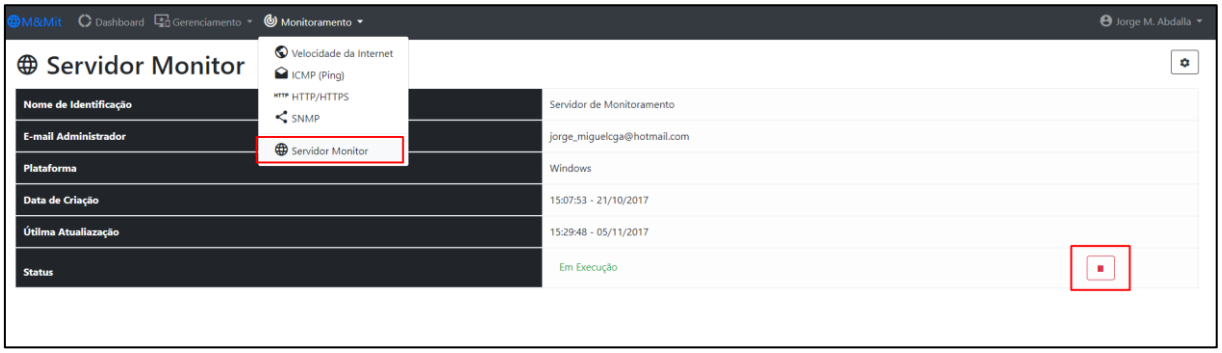


Fonte: Autoria própria.

Na tela da figura 30 pode-se ver os últimos dados monitorados pelo protocolo SNMP, com descrição do valor, o valor e hora da execução. Clicando em configurações também é possível configurar a periodicidade deste monitoramento.

Para configurar este monitoramento é necessário cadastrar identificadores OID, e relacionar esses identificadores com os ativos, acessando as telas para essas ações no menu inferior.

Figura 31 - Tela do Servidor Monitor



Fonte: Autoria própria.

Na figura 31 pode-se ver que é possível também configurar informações gerais do servidor de monitoramento, como e-mail administrador (que receberá todos os e-mails), plataforma de instalação deste servidor, além de poder parar todas as execuções alterando o status do servidor com o botão em destaque na parte direita inferior.

Conforme apresentado neste capítulo, o protótipo atendeu aos requisitos básico do módulo de gerenciamento possibilitando cadastrar, editar e excluir ativos de TI necessários para se realizar essa gerência de forma eficiente. Também atendeu os requisitos do módulo de monitoramento realizando periodicamente, através dos protocolos de rede especificados, teste e busca de valores, e registrando estas execuções no banco de dados. Pode-se considerar, desta forma, que o protótipo é plenamente funcional em um ambiente de rede real, e evidencia as contribuições que agregam à gestão dos recursos de TI.

7. CONSIDERAÇÕES FINAIS

Após a conclusão do desenvolvimento do protótipo observou-se a possibilidade real de tornar o gerenciamento e monitoramento dos ativos de TI uma tarefa simplificada e organizada, que traria muitos benefícios para quem utilizasse o protótipo, que já conta com características que o tornam implantável e funcional.

Observou-se também a necessidade de capacitação técnica, que tornaram esse projeto viável, devem andar juntas, pois sistemas de informação são realmente úteis quando atendem necessidades reais, descomplicando processos complexos e facilitando o modo de trabalhar das organizações, utilizando todos os conhecimentos adquiridos, ou adquirindo novos, e interligando-os de maneira a funcionarem juntos para um propósito definido.

É importante que se destaque que o sistema ainda é um protótipo do que pode vir a ser, com recursos muito limitados, mas com capacidade muito grande de expansão. Novas funcionalidades podem ser inseridas, como, por exemplo, notificações enviadas para aplicativo de celular, sistema de alerta de e-mails, níveis de acessos ao sistema, função essa que se considera o próximo passo para tornar o sistema plenamente implantável, possibilitando o acesso personalizável de diversos tipos de usuários.

Novos tipos de monitoramento podem ser desenvolvidos, de acordo com outros protocolos de rede, assim como testes de portas de serviços de rede, comandos e consultas SQL, ou qualquer tipo mais especifico de monitoramento. No que diz respeito ao gerenciamento de ativos, conforme a área de TI vai expandindo, novas realidades de ativos podem surgir, e novas necessidades podem ser implementadas na ferramenta para possibilitar um gerenciamento eficiente de toda espécie de componentes da área.

Esse trabalho, portanto, conclui com um protótipo que pode se tornar um sistema de grande auxílio aos profissionais da área de TI e suas organizações, além de estimular o debate ao tema, onde novas propostas poderão ser agregadas ao sistema, de forma a contribuir com a valorização da ferramenta bem como enriquecer o gerenciamento de recursos de TI.

REFERÊNCIAS

COMER, Douglas E. **Redes de Computadores e Internet**. Trad. José Valdeni de Lima; Valter Roesler. 6.ed. Porto Alegre: Bookman Editora Ltda., 2016. p. 296. Disponível em: <https://books.google.com.br/books?id=1nwdDAAAQBAJ&printsec=frontcover&dq=redes+de+computadores&hl=pt-BR&sa=X&ved=0ahUKEwjituovvTTAhWKf5AKHVObC8M4ChDoAQgsMAI#v=onepage&q=intranet&f=false>. Acesso em: 16 maio 2017.

COSTA, Ramon Gomes; TODESCHINI, Leonardo. **<WEB>:** Como programar usando ferramentas livres. 1.ed. Rio de Janeiro: Alta Books, 2006. p. 101.

DAVIS, Michele E.; PHILLIPS, Jon A. **Aprendendo PHP e MySQL**. Trad. Rita Sussekind. 2 ed. Rio de Janeiro: Alta Books, 2008. p. 2-5.

DUCKETT, Jon. **Introdução à Programação Web com HTML, XHTML e CSS**, Trad. Acauan Fernandes. 2.ed. Rio de Janeiro, Editora Ciência Moderna Ltda, 2010. p. 26, 256.

ENGHOLM JUNIOR, Hélio. **Engenharia de Software na Prática**. 1 ed. São Paulo: Novatec Editora Ltda., 2013. p. 207. Disponível em: < https://books.google.com.br/books?id=Gq6LBgAAQBAJ&pg=RA1-PA5&dq=arquitetura+de+software&hl=pt-BR&sa=X&ved=0ahUKEwi7qObCksXUAhWGf5AKHTbkC1I4ChDoAQgrMAE#v=onepage&q&f=false>. Acesso em 17 jun. 2017.

ESTEVES, Antonio Matheus Benaion. **Sistema de monitoramento de redes baseado nos protocolos SNMP e Spanning Tree**. 2013. 220 f. Dissertação (Mestrado Profissional em Física) – Centro Brasileiro de Pesquisas Físicas, Rio de Janeiro, 2013. Disponível em: <http://cbpfindex.cbpf.br/publication_pdfs/Dissertacao_AntonioBenaion.2014_01_09_11_55_36.pdf>. Acesso em: 16 jun. 2017.

FLANAGAN, David. **JavaScript:** o guia definitivo. Trad. João Eduardo Nóbrega Tortello. 6.ed. São Paulo: Bookman Editora Ltda, 2013. p. 19.

FOROUZAN, Behrouz A. **Comunicação de Dados e Redes de Computadores**. Trad. Ariovaldo Griesi. 4.ed. Porto Alegre: Bookman, 2010. p. 861.

GUEDES, Gilleanes T. A. **UML:** uma abordagem prática. São Paulo: Novatec Editora, 2009. p. 19, 22, 106, 285-230.

HEUSER, C. A. **Projeto de Banco de Dados**. 5 ed. Porto Alegre: Bookman, 2004. p. 16, 17. Acesso em: 12 out. 2017. Disponível em: <http://www.fernandozaidan.com.br/pit-grad/Diversos/Livros_Disciplinas/Projeto_de_Banco_de_Dados_-_Carlos_Alberto_Heuser.pdf>.

LIMA, Adilson da Silva. **UML 2.0:** Do Requisito à Solução. 4.ed. São Paulo: Érica Ltda., 2010. p. 57.

MAGALHÃES, Ivan Luizio; PINHEIRO, Walfrido Brito. **Gerenciamento de Serviços de TI na Prática:** Uma abordagem com base na ITIL®. 1.ed. São Paulo: Novatec Editora Ltda., 2010.

PITT, Chris. **Pro PHP MVC**. 1.ed. New York: Apress, 2012. p. 1. Disponível em: <https://books.google.com.br/books?id=lMs0bObg8bIC&printsec=frontcover&dq=php+mvc&hl=pt-BR&sa=X&ved=0ahUKEwiJxoakj8XUAhUFDZAKHWNHAkAQ6AEIKjAA#v=onepage&q&f=false>. Acesso em: 17 jun. 2017.

SOMMERVILLE, Ian. **Engenharia de Software.** Trad. Selma Shin Shimizu Melnikoff. 8.ed. São Paulo: Pearson Addison, 2010. p. 5.

TJG WEB. **Curso PHP - Micro Framework do zero**. *Youtube*, 2017. Disponivel em: <https://www.youtube.com/playlist?list=PLSYIyzca1f9wGynWlC-SH2lVBkE8S81A0>. Acesso em: 19 set. 2017.

TURBAN, Efraim; VOLONINO, Linda. **Tecnologia da Informação para Gestão:** Em busca do melhor desempenho estratégico e operacional. Trad. Aline Evers. 8.ed. Porto Alegre: Bookman Editora Ltda., 2013. p. 9.