

UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO

CURSO DE ENGENHARIA DA COMPUTAÇÃO

HERBERT HUDSON GONÇALVES TRINDADE - RA 1817498

JOSAFÁ JOSÉ DOS SANTOS JUNIOR - RA 1804968

MATHEUS DE FREITAS FUJII - RA 1805141

VICTOR GARDIM RODRIGUES - RA 1816313

BRUNO RAFAEL GIMENES - RA 1804160

GUILHERME RODRIGO DA SILVA - RA 1600376

**SISTEMA DE GERENCIAMENTO DE ASSISTÊNCIAS
TÉCNICAS COM JAVA, MYSQL E HIBERNATE**

Link da Apresentação:
<<https://youtu.be/aZtrYvxXYQg>>

Hortolândia - SP
2021

UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO

CURSO DE ENGENHARIA DA COMPUTAÇÃO

HERBERT HUDSON GONÇALVES TRINDADE - RA 1817498

JOSAFÁ JOSÉ DOS SANTOS JUNIOR - RA 1804968

MATHEUS DE FREITAS FUJII - RA 1805141

VICTOR GARDIM RODRIGUES - RA 1816313

BRUNO RAFAEL GIMENES - RA 1804160

GUILHERME RODRIGO DA SILVA - RA 1600376

SISTEMA DE GERENCIAMENTO DE ASSISTÊNCIAS TÉCNICAS COM JAVA, MYSQL E HIBERNATE

Trabalho de Conclusão de Curso
apresentado à Universidade Virtual do
Estado de São Paulo, como parte dos
requisitos necessários à obtenção do título
de Engenheiro, no Curso de Engenharia da
Computação, sob orientação do Prof. Me.
Guilherme Schiavão Padovani

Hortolândia - SP
2021

RESUMO

O avanço e popularização da tecnologia possibilitou uma verdadeira revolução na vida cotidiana das pessoas. Quanto maior o número de aparelhos, contudo, maior a necessidade de reparos e consertos em lugares especializados. Com o aumento de demanda, as assistências técnicas passam a enfrentar o desafio de gerenciar o negócio de forma a otimizar tempo e custos, sem abrir mão da excelência no atendimento e execução dos serviços. Diante deste cenário, foi proposta a construção de um software que ajude neste gerenciamento. Para isso, além de levantamento bibliográfico, foi utilizada a experiência empírica dos integrantes do grupo.

Palavras-chave: Assistência técnica, gerenciamento, Java, MySQL, Hibernate

ABSTRACT

With the advance and popularization of technology, people's lives turned into something completely different. The increase in the number of devices, however, brought the need for specialized repairs. With the rising demand, technical assistances are now facing the challenge of managing the business in order to optimize time and costs, without sacrificing neither service excellence nor execution. Given this scenario, it was proposed to build a software that helps its management. For this, in addition to a bibliographic survey, the empirical experience of the members of the group was used.

Keywords: Technical assistance; Management; Java; MySQL; Hibernate.

Lista de Figuras

Figura 1. Linha do tempo da história da computação.	13
Figura 2. Implementação da arquitetura MVC no SIGEAT.	18
Figura 3. Implementação do código de mapeamento objeto-relacional de Usuários no SIGEAT	19
Figura 4. Chamadas às bibliotecas do Hibernate para persistência de dados	20
Figura 5. Teste unitário do método que retorna todos os clientes registrados	24
Figura 6. Resultado aprovado de um teste unitário do SIGEAT	25
Figura 7. Fluxograma de processo da assistência técnica de Paulo.	26
Figura 8. Diagrama Entidade-Relacionamento do SIGEAT	27
Figura 9a. Representação de um ator em um diagrama de casos de uso.	28
Figura 9b. Caso de uso associado à um ator.	28
Figura 10. Diagrama de casos de uso do SIGEAT.	29
Figura 11. Representação de uma classe em um diagrama de classes.	30
Figura 12. Classes de mapeamento objeto-relacional do SIGEAT.	30
Figura 13. Classes de acesso aos dados.	31
Figura 14. Classes de controle do SIGEAT.	31
Figura 15. Classes da camada de aplicação do SIGEAT.	32
Figura 16. Tela de login do SIGEAT.	33
Figura 17. Tela de cadastro de novos usuários do SIGEAT.	34
Figura 18. Tela de pesquisa de usuários do SIGEAT.	34
Figura 19. Tela de cadastro de clientes do SIGEAT.	35
Figura 20. Tela de pesquisa de clientes do SIGEAT.	35
Figura 21. Tela de OS do SIGEAT.	36
Figura 22 :Relatório de clientes.	36
Figura 23. Documento de ordem de serviço.	37

Sumário

1 INTRODUÇÃO	6
2 OBJETIVOS	8
2.1 OBJETIVO GERAL	8
2.2 OBJETIVOS ESPECÍFICOS	8
3 FUNDAMENTAÇÃO TEÓRICA	8
3.1 LINHA DO TEMPO DA HISTÓRIA DA COMPUTAÇÃO	9
3.1.1 COMPUTAÇÃO PRÉ-ELETRÔNICA	9
3.1.2 COMPUTAÇÃO ELETRÔNICA	10
3.1.3 COMPUTAÇÃO DIGITAL	13
3.2 A LINGUAGEM JAVA	14
3.3 BREVE HISTÓRIA DOS BANCOS DE DADOS RELACIONAIS	16
3.5 HIBERNATE	18
3.6 ENGENHARIA DE SOFTWARE	21
3.6.1 LINGUAGEM DE MODELAGEM UNIFICADA(UML)	21
3.6.2 GIT	22
3.6.3 TRELLO	23
3.6.4 TESTES UNITÁRIOS	23
4 METODOLOGIA	25
4.1 ANÁLISE E LEVANTAMENTO DE REQUISITOS	25
4.2 MODELO DO BANCO DE DADOS	27
4.3 DIAGRAMA DE CASOS DE USO	27
4.4 DIAGRAMA DE CLASSES	29
5 RESULTADOS/DISSCUSSÕES	32
5.1 INTRODUÇÃO AO SIGEAT:	33
5.2 APRESENTAÇÃO DO SIGEAT	33
6 CONSIDERAÇÕES FINAIS	37
7 REFERÊNCIAS	39
GLOSSÁRIO	41
ANEXOS	42

1 INTRODUÇÃO

Os computadores, da forma que os conhecemos hoje, passaram por diversas transformações e foram se aperfeiçoando ao longo do tempo, acompanhando os avanços das áreas da matemática, física, química, engenharia e eletrônica.

Desde a primeira geração de computadores (que funcionavam a válvula), que datam da década de 1950, até o início da terceira geração, cujo início remete à meados da década de 1960, os computadores tinham dimensões e necessidades de funcionamento que tornava proibitivo seu uso de forma domiciliar.

Com o desenvolvimento da tecnologia da informação, a partir de meados da década de 1970, contudo, os computadores começaram a diminuir de tamanho, aumentar sua velocidade e capacidade de processamento de dados e começaram a diminuir de tamanho.

Apenas a partir da década de 1990, quando o custo de produção, aliado a máquinas com menor consumo de energia, possibilitaram a popularização dos computadores pessoais. Essa popularização pôde ser sentida, em especial, com o surgimento de computadores portáteis, em meados de 2000, e, em especial, dos smartphones, em meados da década de 2010.

Para que se tenha ideia deste fenômeno, o Brasil tem 152 (cento e cinquenta e dois) milhões de usuários de internet, o que corresponde a 81% da população do país (com 10 anos ou mais). A estimativa é da pesquisa TIC Domicílios-2020 (Edição COVID-19 - Metodologia Adaptada), promovida pelo Comitê Gestor da Internet do Brasil (CGI.br). Destes usuários, 42% tem acesso a internet via computador (desktop, notebook e tablet), 99% acessam via telefone celular, 10 % via aparelho de videogame e 44% via televisão.

A mesma pesquisa (TIC Domicílios, 2020) demonstra que, em relação aos domicílios, o principal tipo de conexão é a banda larga fixa (68%), com especial aumento das conexões por cabo ou fibra óptica. O estudo também mostrou que houve um aumento na presença de computador (desktop, portátil ou tablet) nos domicílios (de 39% em 2019 para 45% em 2020), revertendo uma tendência de declínio que vinha se desenhando nos últimos anos.

A democratização da tecnologia, não obstante, trouxe consigo uma nova demanda: lugares e profissionais que pudessem levar estes equipamentos em caso

de dano e/ou necessidade de atualização tanto ao hardware quanto ao software. Desta demanda surgiram as assistências técnicas. A princípio, especializadas em desktops e, com a evolução da tecnologia, também se especializaram em notebooks e outros computadores portáteis e, por fim, em tablets e smartphones.

Segundo (Magalhães & Pinheiro, 2007), toda empresa prestadora de serviços está apoiada em três pilares básicos: infraestrutura, processos e pessoas. A infraestrutura fornece as ferramentas que apoiam os processos da empresa, que, por sua vez, são executados por pessoas. Usualmente, estabelecimentos especializados em manutenção de equipamentos de informática não apresentam grandes necessidades infra estruturais, contudo, seu desempenho depende muito mais do estabelecimento de rotinas com processos eficientes realizados por profissionais bons e capacitados.

Em assistências técnicas, as rotinas envolvem mais que planejar e controlar os serviços prestados pela empresa, outros aspectos como a satisfação do cliente, a capacidade de cobertura no atendimento, a capacidade e tratabilidade na comunicação e a qualidade do trabalho executado são de extrema importância.

O gerenciamento deste negócio exige – em cada um dos seus processos – eficiência e assertividade, tanto para garantir que o cliente aprove e recomende o serviço (via de regra, é a indicação de terceiros que origina o maior número de novos clientes) quanto para assegurar bons resultados à empresa.

Como exercício de imaginação, basta pensar no tamanho da confusão que a falta de gestão (ou mesmo a gestão falha) nessa área, como produtos entregues para clientes errados, falta de peças para manutenção, visitas em horários trocados, não execução de serviços no prazo estabelecido ou orçamentos não condizentes, podem ocasionar. Na realidade, esses tipos de intercorrências não são incomuns de ocorrerem.

Dentre as principais necessidades do setor, em especial as assistências de pequeno porte (que são a maioria), está a falta de informatização, por mais contraditório que pareça. Essa não informatização acaba tendo como consequência a desorganização dos processos, seja no gerenciamento das requisições de serviços e orçamentos ou no manejo de equipamentos que entram e saem da assistência, bem como no controle de peças e outros artigos de estoque.

Para melhor organizar esta informatização, criou-se um roteiro a ser seguido baseado em cinco etapas: 1. Recepção; 2. Orçamento; 3. Execução da ordem de serviço; 4. Complementação da OS com relatório de execução e, 5. Emissão de nota fiscal e liberação de saída. Desta forma, pode-se criar um sistema (software) com pontos de checagem e travas que garantam a correta execução do fluxograma de trabalho.

2 OBJETIVOS

A seguir são apresentados os objetivos gerais e específicos deste trabalho.

2.1 OBJETIVO GERAL

Desenvolver um software que auxilie no gerenciamento de assistências técnicas de informática. Neste sistema estarão contidos dados referentes aos clientes (para fins de identificação e comunicação), usuários internos (técnicos - para fins de identificação de responsável) e às ordens de serviço (OS), para fins de identificação de equipamentos, serviços a serem realizados, demandas de materiais e peças e workflow.

2.2 OBJETIVOS ESPECÍFICOS

Ao final deste projeto, espera-se ter construído uma solução que satisfaça os pontos anteriormente listados. Para além da satisfação dos requisitos funcionais, espera-se que o produto final obtido seja capaz de oferecer um design simples e intuitivo, em especial na sua utilização diária, de forma que a curva de aprendizagem, por qualquer novo funcionário, ou até em caso de expansão da utilização do software para outras assistências, seja a menor possível.

3 FUNDAMENTAÇÃO TEÓRICA

Abaixo é apresentada a fundamentação teórica contendo conceitos que devem ser conhecidos para a melhor compreensão deste trabalho.

3.1 LINHA DO TEMPO DA HISTÓRIA DA COMPUTAÇÃO

A computação evoluiu muito desde seus primórdios até os dias de hoje; podemos dividir este processo em três grandes períodos, a saber:

3.1.1 COMPUTAÇÃO PRÉ-ELETRÔNICA

Por volta de 5500 A.C, povos da antiguidade utilizavam o ábaco para computar as trocas, vendas e outras necessidades contábeis e, por isso, é considerado a primeira calculadora do mundo, e o início da computação. Já na Grécia, em aproximadamente 200 A.C foi desenvolvido o crivo de Erastóstenes, um algoritmo simples e prático para encontrar números primos.

Já muito tempo depois, nos idos de 1500 foi projetado por Leonardo da Vinci um calculador mecânico, que utilizaria engrenagens para efetuar cálculos numéricos, mas nunca foi construído.

Em 1632 William Oughtred criou a régua de cálculo, que funcionava como uma calculadora analógica. Pouco tempo depois, em 1640 foi criada por Blaise Pascal a la pascaline, para ajudar seu pai que era coletor de impostos e podia efetuar as 4 operações básicas (adição, subtração, multiplicação e divisão).

Praticamente duzentos anos se passaram até que, em 1822, foi criada a máquina diferencial por Charles Babbage, que era capaz de realizar cálculos polinomiais. Na sequência (1837), o mesmo Babbage, com auxílio fundamental de Augusta Ada Byron King, Condessa de Lovelace, propuseram uma evolução da máquina diferencial, a máquina analítica. Durante o período em que esteve envolvida com o projeto, ela desenvolveu os algoritmos que permitiriam à máquina computar os valores de funções matemáticas, e por isso, foi considerada como a primeira programadora da história.

Em 1890 Hermann Hollerith criou uma máquina que utilizava diversos cartões perfurados, para o preenchimento de dados, esta máquina foi muito utilizada para preencher dados de pesquisa, tais como o censo.

3.1.2 COMPUTAÇÃO ELETRÔNICA

O próximo grande avanço ocorreu em 1906, quando Lee D. Forest inventou a válvula eletrônica, que foi muito utilizada na construção dos primeiros computadores. Trinta anos depois, o matemático inglês Alan Turing apresentou um dispositivo imaginário, que podia ler, escrever e apagar símbolos binários em uma fita de comprimento ilimitado dividida por quadrados de mesmo tamanho, era o conceito fundamental dos computadores como se conhece.

Em 1941 o cientista alemão Konrad Zuse criou o primeiro computador programável para resolver equações mais complexas, o Z3; também foi o primeiro computador a utilizar o sistema binário. Na mesma época, em 1944, foi construído o Harvard Mark I, um computador projetado em 1930 por Howard Aiken e construído em parceria com a IBM, utilizava programas em papel perfurado e também usava relés.

Mas foi em 1946, projetado por John William Mauchly e John Presper Eckert Jr, que entrou em funcionamento o ENIAC (Electronic Numerical Integrator and Computer). Este foi considerado o primeiro computador eletrônico, e contava com aproximadamente 17.468 válvulas e ocupava um espaço de 170 metros quadrados.

Enquanto o ENIAC estava em desenvolvimento, John Von Neumann apresentou uma arquitetura que segue até os dias atuais como uma arquitetura básica de um computador. Esta arquitetura foi utilizada, por exemplo, no EDVAC (Electronic Discrete Variable Automatic Computer) que também utilizava o sistema binário, na sua construção e foi projetado pelos mesmos desenvolvedores do ENIAC.

Criado em 1947 por John Bardeen, Walter Brattain e William Shockley, da Bell Labs, revolucionou a computação, substituindo as válvulas e sendo muito importante até os dias de hoje nos eletrônicos, os três ganharam o prêmio Nobel de física em 1956.

Em 1958 Jack Kilby projetou o primeiro circuito integrado na Texas Instruments, no mesmo período Robert Noyce desenvolveu um projeto similar, os dois são considerados inventores deste componente muito importante até os dias de hoje. Quatro anos depois, Douglas Engelbart projetou o primeiro mouse juntamente com Bill English.

Em 1969 foi criada a Arpanet, desenvolvida pela ARPA, setor de pesquisas do departamento de defesa americano, cujo objetivo era permitir que os computadores pudessem trocar informações entre si, é considerada a precursora da internet moderna.

Criada em 1972 por Dennis Ritchie, que trabalhava no Bell Labs, a linguagem C é utilizada até os dias atuais, sendo sua principal aplicação a criação de sistemas operacionais.

Duas das maiores empresas do ramo computacional surgiram praticamente juntas. Fundada em 1975 por Bill Gates e Paul Allen, desenvolvedora do sistema operacional Windows, a empresa também começou a atuar em outros ramos da tecnologia como games, nuvem, entre outros, sendo uma das maiores empresas de tecnologia atualmente. Já a Apple, fundada em 1976 por Steve Jobs, Steve Wozniak e Ronald Wayne, focada no desenvolvimento de hardware e software, a Apple se tornou uma das empresas mais bem sucedidas no ramo da tecnologia, sendo uma das empresas mais valiosas do mundo atualmente.

O início da popularização dos computadores se deu em 1981, quando foi lançado pela IBM um computador para uso pessoal; Utilizava o processador Intel 8088 de 4,77 MHz de clock e memória RAM de 16 Kb e outra de 64 Kb podendo ser expandida até 256 Kb. Sua arquitetura ainda é utilizada nos computadores modernos. Em 1984 o Macintosh foi apresentado pela Apple, inicialmente o Mac OS ficou conhecido como System e foi o sistema operacional utilizado pela Apple, foi o sistema que popularizou o uso da interface gráfica.

A seguir, em 1985 foi a vez da Microsoft apresentar seu Windows 1.0, sendo apenas um software de interface gráfica rodando sobre o MS-DOS. Somente com Windows NT em 1993, o Windows passou a ser considerado como um sistema operacional completo. O Windows é atualmente o sistema operacional mais utilizado em computadores, estando muito à frente de seus concorrentes.

Outro marco de 1985 foi a criação da linguagem C++ por Bjarne Stroustrup, sendo uma evolução da linguagem C permitindo a **POO**(Programação Orientada a Objeto), muito popular até os dias atuais. Outro avanço no campo das linguagens de programação foi o surgimento da Python em 1990. Criada por Guido Van Rossum, é

uma linguagem muito poderosa e que possui sintaxe extremamente simples; com isso vem ganhando cada vez mais popularidade.

Um dos pontos de virada na evolução da computação, da informática e da forma como lidamos com a informação se deu em 1990, quando Tim Berners-Lee criou a World Wide Web, a internet atual, ela se popularizou rapidamente e vem se tornando cada vez mais importante na vida das pessoas.

Em 1991 Linus Torvalds desenvolveu o Linux, sistema operacional livre, gratuito e de código aberto que permite que qualquer desenvolvedor possa criar sua própria versão do sistema, tendo como base o núcleo criado por Torvalds. Existem diversas distribuições de sistemas Linux, entre elas o Ubuntu, Linux Mint, Manjaro, Debian, entre outros.

A linguagem Java foi criada em 1995 por James Gosling e sua equipe, que trabalhavam na Sun Microsystems, sendo uma linguagem que roda através de uma máquina virtual contendo seu compilador, isto permite que um mesmo código possa rodar em vários sistemas operacionais sem ou com poucas alterações, tendo uma grande portabilidade. Na sequência, no mesmo ano, foi criada a Javascript por Brendan Eich para uso no navegador Netscape, seu nome na verdade é EcmaScript, sendo executada diretamente no navegador do usuário sendo extremamente simples de se trabalhar. Com a criação do NodeJS ganhou mais versatilidade, podendo ser utilizada em qualquer tipo de aplicação, incluindo aplicativos mobile.

A Internet das Coisas (IOT - Internet Of Things, na sigla em inglês) foi um termo criado por Kevin Ashton em 1999, para exprimir a troca de informações pela internet entre computadores, além de outros dispositivos. Hoje em dia temos smartphones, televisores, video games, relógios, veículos, e muitos outros dispositivos trocando informações através da internet.

Criada em 2000 pela Microsoft fazendo parte da plataforma Dotnet, a criação da Linguagem C# (ou C+++++) é creditada a Anders Hejlsberg, sendo uma linguagem multiparadigma e muito popular até os dias atuais.

3.1.3 COMPUTAÇÃO DIGITAL

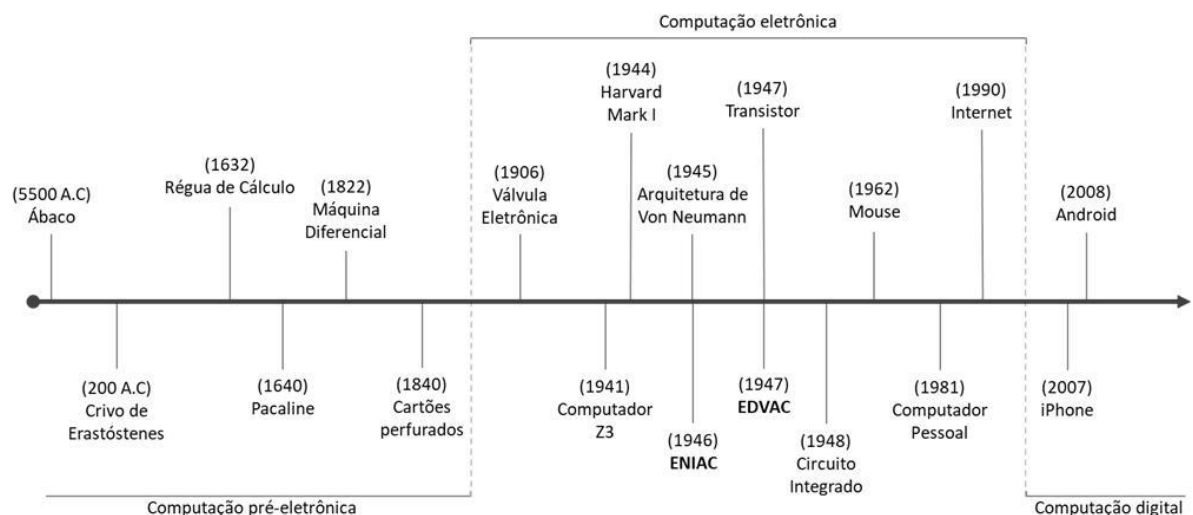
Em 2007 a Apple apresentou o iPhone, um dispositivo trouxe várias funções até então disponíveis apenas em um computador para dentro de um celular, tendo um sistema operacional completo, o IOS.

No ano seguinte foi lançado pela Google o Android, que também trouxe diversas funcionalidades de computadores para celulares, diversos fabricantes de celulares adotaram o Android como sistema operacional, ajudando a popularização dos smartphones e tornando o Android o sistema operacional mais popular entre os smartphones.

Com o advento destes dois sistemas, somados à expansão da IOT, o mundo iniciou uma revolução digital.

Para que se entenda a dimensão dessa história, a figura abaixo apresenta os pontos de destaque.

Figura 1. Linha do tempo da história da computação.



Fonte: próprios autores

Depois desta pequena linha do tempo da história da computação vamos focar no Java, a linguagem de programação utilizada na construção da solução apresentada neste trabalho.

3.2 A LINGUAGEM JAVA

O Java é uma das linguagens de programação mais populares atualmente, sendo aplicada em diversas áreas no desenvolvimento de software, agora vamos conhecer a história desta importante linguagem de programação que faz parte da vida de muitos desenvolvedores e usuários.

O desenvolvimento da linguagem Java começou no ano de 1991, Patrick Naughton, Mike Sheridan, e James Gosling, da Sun Microsystems, empresa fabricante de computadores, semicondutores e software, foram os responsáveis pelo projeto que ficou conhecido como “Green Project”. A princípio o “Green Project” tinha como objetivo integrar computadores e outros equipamentos domésticos, a equipe de Gosling ficou conhecida como “Green Team”.

Em 1992 juntamente com uma equipe de 13 pessoas, eles apresentaram um protótipo chamado “Star Seven(*7)”, que consistia em um controle remoto com uma tela touchscreen(sensível ao toque), também foi apresentado um mascote para o projeto, esse mascote foi chamado de “Duke”, ele era o guia virtual do equipamento, ensinando o usuário a usar suas funcionalidades, e acompanha a linguagem Java até os dias atuais. O “Star Seven” podia controlar diversos dispositivos e aplicações, James Gosling batizou a linguagem de programação criada para o dispositivo de “Oak (carvalho)”, em alusão ao carvalho que sempre podia ser observado através da janela de seu escritório.

O “Star Seven” era um equipamento que estava à frente de seu tempo, porém o mercado não estava preparado para isso, fazendo com que o dispositivo criado por James Gosling e sua equipe fosse esquecido rapidamente.

A internet estava se tornando cada vez mais popular, James Gosling e sua equipe viram uma oportunidade nisso, adaptando a linguagem “Oak” para a internet. Em 1995 uma nova versão do “Oak” foi lançada, desta vez a linguagem foi finalmente batizada de Java, em referência ao termo “Java coffee”, um café considerado bem forte, que a equipe consumia durante o trabalho.

A linguagem Java foi rapidamente adotada por vários desenvolvedores, em pouco tempo se tornou a linguagem mais popular e foi uma grande revolução, a Netscape inclusive batizou sua linguagem de Javascript por questões de marketing devido à popularidade do Java. O Java funciona através da JRE (Java Runtime Environment), um software que contém uma máquina virtual (JVM) instalada no

computador no usuário, o código Java é compilado em um *bytecode* e depois interpretado pela JVM, a JRE possui versões para diversos sistemas operacionais, isto permite que um mesmo código fonte funcione nestes sistemas operacionais sem ou com poucas alterações, permitindo uma grande portabilidade, este é dos grandes diferenciais do Java. O kit de desenvolvimento é conhecido como JDK (Java Development Kit), ele contém o JRE e também algumas ferramentas para criação de programas Java.

Em 2008 a Oracle empresa focada no desenvolvimento de hardware e software e bancos de dados, comprou a Sun Microsystems, ficando com a concessão do Java.

O Java é a linguagem de programação nativa utilizada para criar aplicativos para o sistema operacional Android, também é possível utilizar outras linguagens, entretanto é necessário o uso de **frameworks**. O Android possui a maior parte do mercado de sistemas operacionais para smartphones, pois foi adotado pela maioria dos fabricantes. Atualmente a Google vem tentando substituir o Java por uma linguagem chamada Kotlin, devido a divergências que teve com a Oracle, mas o Java segue presente na plataforma Android.

O Java possui uma das maiores comunidades de desenvolvedores, por muitos anos foi a linguagem mais popular entre os desenvolvedores, mesmo tendo perdido esse posto ainda é uma das linguagens de programação mais populares e vem ganhando novas versões.

Para desenvolver aplicações Java podemos utilizar diversas **IDEs**, neste trabalho utilizaremos o Netbeans, IDE oficial da linguagem, sendo mantida pela Apache Foundation.

O Netbeans fornece um ambiente integrado para desenvolver e compilar programas. O Netbeans é gratuito, com código aberto e multiplataforma. (BERTOLINI *et. al*, 2019).

O Netbeans, possui um editor de código fonte integrado, com vários recursos, para desenvolvimento em aplicações C e java e aplicações web, possui suporte a banco de dados e é independente, podendo ser utilizado em Windows, Linux e MAC OS. (BERTOLINI *et. al*, 2019).

Um ponto muito importante, é ressaltar que a utilização do Netbeans, auxilia na escrita dos programas, ou seja, a sintaxe, porém a lógica e semântica ficará à critério do programador. (BERTOLINI *et. al*, 2019).

3.3 BREVE HISTÓRIA DOS BANCOS DE DADOS RELACIONAIS

Bancos de dados são ferramentas extremamente importantes, eles são responsáveis por armazenar diversas informações que são de muito valor tanto do mundo digital quanto no real, por exemplo, um banco que precisa armazenar diversas informações de seus clientes, órgãos governamentais que armazenam informações de seus cidadãos, uma empresa que guarda todos os seus dados fiscais, operacionais, etc. Bancos de dados permitem armazenar essas informações e muitos outros tipos de informação no meio digital, diminuindo o uso de papel e otimizando espaço dentro das instituições, sendo menos custoso do que organizar pastas em um arquivo para consultas posteriores, também dispensando o uso de planilhas digitais que podem apresentar inconsistência de dados se não forem editadas e atualizadas constantemente.

A maioria das aplicações pioneiras que utilizava banco de dados mantinha registros de grandes organizações, como universidades, hospitais, bancos, etc. Em muitas destas aplicações, existiam registros de estruturas muito semelhantes, com informações similares sendo mantidas para cada registro. Havia muitos tipos de registros e diversos inter-relacionamentos entre eles. Um dos principais problemas destes sistemas de bancos de dados era a mistura entre relacionamentos conceituais, o armazenamento físico e a localização de registros no disco. Era complicado reorganizar o banco de dados quando mudanças eram feitas para atender a novos requisitos de aplicação (ELMASRI e NAVATHE, 2005).

Em 1970 Edgar Frank Codd da IBM apresentou o modelo de dados relacional, que foi muito importante para a evolução dos bancos de dados quando explorado por outros cientistas. No final de década de 1970 foi criado um sistema tendo como base a idéia de Codd chamado “Sistema R” e também foi criada uma linguagem de consulta, que segue como padrão deste tipo de banco de dados até os dias de hoje, a linguagem SQL (Structured Query Language) ou linguagem de consulta estruturada. Diversos sistemas de bancos de dados surgiram e o modelo relacional logo se tornou muito popular.

No modelo relacional (também conhecido como SQL) as entidades podem apresentar diversos relacionamentos entre si, isto é feito através de uma chave chamada chave estrangeira, evitando a repetição de informações em cada registro,

por exemplo, podemos ter duas entidades, clientes e pedido, um cliente pode fazer vários pedidos, cada entidade representa uma tabela no banco de dados, se for necessária uma atualização em algum cliente cadastrado, todos os pedidos que se relacionam com o cliente atualizado, já terão suas informações de cliente atualizadas automaticamente, isto é muito importante para manter a consistência dos dados.

Os bancos de dados relacionais dividem em tabelas que podem ser relacionadas entre si, cada registro de cada tabela também pode ser chamado de *tupla* ou linha, as colunas representam os atributos de cada entidade que serão armazenados, por exemplo, podem-se armazenar informações como nome, sobrenome, sexo, data de nascimento de um cliente, com cada um destes itens representando uma coluna de uma tabela cliente. Cada tabela possui uma coluna com valor único, isto é, nenhum valor pode estar presente em mais de uma linha garantindo que não haja repetição de um mesmo registro, essa coluna chama-se chave primária. Bancos de dados relacionais podem armazenar diversos tipos de dados como valores numéricos, cadeias de caracteres, datas, valores monetários, imagens, entre outros.

Existem diversos sistemas de bancos de dados relacionais como SQL Server, Oracle DB, MariaDB, PostgreSQL, SQLite, entre outros. Um dos bancos de dados relacionais mais populares até os dias atuais é o MySQL. Ele foi criado em 1980 por David Axmark, Allan Larsson e Michael Widenius. O MySQL é um SGDB (Sistema de Gerenciamento de Banco de Dados) extremamente simples de se trabalhar, com suporte a diversas plataformas, sendo utilizadas por diversas grandes empresas. Possui o código aberto e até a presente data se encontra administrado pela Oracle, grande empresa de tecnologia que também detém a plataforma Java, adquirida através da compra da Sun Microsystems em 2009.

3.4 DESIGN PATTERN MVC

O Design Pattern MVC, é um padrão de arquitetura de desenvolvimento de software no qual as funcionalidades são divididas em três camadas, estas camadas são:

1. Model (modelo). Camada responsável pelo acesso às tabelas do banco de dados, o mapeamento objeto-relacional é feito nesta camada bem como as funções **CRUD** (consultas e **transações** no banco de dados);

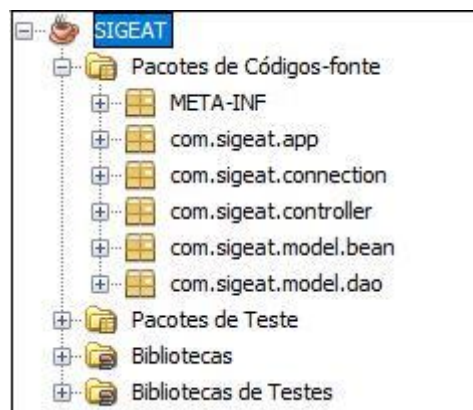
2. View (visão). Camada responsável pela exibição dos dados para o usuário, a interface gráfica, as janelas do software são o principal componente desta camada.

3. Controller (controle). Camada responsável pelas regras de negócio, ou seja, pela validação das restrições de uso das funcionalidades do software, por exemplo, o nome de um cliente sempre deve ser informado ao criar um novo registro.

O padrão MVC é uma arquitetura muito segura, garantindo o bom funcionamento do software, também se ganha tempo de desenvolvimento bem como uma ótima organização do código, facilitando atualizações futuras no software.

Na figura abaixo temos a representação da implementação do Design Pattern MVC no SIGEAT dentro da IDE Netbeans, note que os pacotes representam estas camadas abordadas acima.

Figura 2. Implementação da arquitetura MVC no SIGEAT.



Fonte: Próprios autores

A camada de modelo se subdivide em duas partes, o bean representa o mapeamento objeto-relacional do banco de dados, já **DAO** (Data Access Object) representa apenas os **métodos** de acesso ao banco de dados para funções CRUD (consultas e transações no banco de dados).

O mapeamento objeto-relacional fica a cargo do framework Hibernate que veremos a seguir.

3.5 HIBERNATE

O Hibernate é um framework para linguagem Java, utilizado para mapeamento de objetos em um banco de dados. Este mapeamento facilita principalmente quando é necessário fazer alterações na estrutura do banco de dados, diminuindo a quantidade de código que deve ser escrita. O Hibernate necessita de um arquivo XML

para guardar as informações do banco de dados, nele constam informações como nome do banco de dados, usuário, senha, e cada tabela que será mapeada. O Hibernate utiliza o JPA (Java Persistence API), API presente no Java Development Kit para mapeamento objeto-relacional.

O Hibernate pode ser utilizado para diversos bancos de dados relacionais como Mysql, Oracle DB, PostgreSQL, entre outros, dando ao desenvolvedor um ganho significativo de tempo ao escrever as linhas de código, é possível implementar funções CRUD com muito mais rapidez, ajudando na manutenibilidade do software desenvolvido.

A figura abaixo mostra a implementação do código que faz o mapeamento objeto-relacional da tabela Usuarios do banco de dados utilizando a API JPA do Java:

Figura 3. Implementação do código de mapeamento objeto-relacional de Usuários no SIGEAT

```
@Entity
public class Usuarios extends Pessoas implements Serializable {

    @Column(nullable = false, length = 15)
    private String login;

    @Column(nullable = false, length = 15)
    private String senha;

    @Column(length = 20)
    private String perfil;

    public String getLogin() {
        return login;
    }

    public void setLogin(String login) {
        this.login = login;
    }

    public String getSenha() {
        return senha;
    }

    public void setSenha(String senha) {
        this.senha = senha;
    }

    public String getPerfil() {
        return perfil;
    }
}
```

Fonte: Próprios autores

Note que a notação @Entity acima da classe indica que ela é uma classe de mapeamento de uma entidade do banco de dados, todas entidades devem ser mapeadas desta maneira. O uso da palavra extends indica que a **herança** está sendo empregada, isto é muito importante dentro do mapeamento objeto-relacional pois temos várias tabelas que possuem colunas em comum. As **variáveis** possuem a notação @Column indicando que representam as colunas de uma tabela de banco de dados, os valores após a notação representam as **constraints** destas colunas. O código foi implementado utilizando boas práticas de programação como **encapsulamento** e herança muito importantes em POO (Programação Orientada a Objeto). Todo este mapeamento faz parte do modelo no pacote bean.

Abaixo temos a figura 4, que mostra a implementação do código que faz a chamada às bibliotecas do Hibernate:

Figura 4. Chamadas às bibliotecas do Hibernate para persistência de dados

```
public class UsuariosDAO implements IUsuariosDAO {

    @Override
    public Usuarios save(Usuarios usuario) {

        EntityManager em = new ConnectionFactory().getConnection();

        try {
            em.getTransaction().begin();

            //Se o usuario ja possui um id
            if (usuario.getId() != null) {

                em.merge(usuario);

            } else {

                em.persist(usuario);

            }

            em.getTransaction().commit();

        } catch (Exception e) {
            //Desfazer alterações
            em.getTransaction().rollback();
            e.printStackTrace();
        } finally {
            em.close();
        }
    }
}
```

Fonte: Próprios autores

Note que com poucas linhas de código temos um método que faz a inserção e a atualização de um registro. A palavra `implements` indica que este código implementa uma interface na qual os métodos são apenas declarados, o uso da interface é uma boa prática de programação de POO (Programação Orientada a Objeto). O Hibernate pode desfazer a transação caso aconteça alguma falha durante seu processo.

Para o bom desenvolvimento de um software além das boas práticas descritas acima devemos usar práticas de Engenharia de Software, vamos conhecer algumas destas práticas que são importantes no desenvolvimento de software.

3.6 ENGENHARIA DE SOFTWARE

Os sistemas de software são abstratos e intangíveis. Eles não são restringidos pelas propriedades dos materiais, nem governados pelas leis da física ou pelos processos de manufatura. Isso simplifica a engenharia de software, porque não há limites naturais para o potencial do software, no entanto devido esta falta de restrições físicas, os sistemas de software podem se tornar extremamente complexos de modo muito rápido, difíceis de entender e caros de alterar (SOMMERVILLE, 2011).

A engenharia de software é focada no processo de desenvolvimento de software através de princípios, práticas e metodologias que foram evoluindo com o passar do tempo. Tudo isso contribui para que o software apresente os melhores resultados sendo confiável, eficiente e eficaz. Isso facilita qualquer alteração futura que o software precise receber, sem que seja muito caro e trabalhoso.

Abaixo vamos abordar algumas práticas de engenharia de software que foram utilizadas no desenvolvimento deste trabalho começando pela linguagem de modelagem unificada.

3.6.1 LINGUAGEM DE MODELAGEM UNIFICADA(UML)

A linguagem de Modelagem Unificada (UML) de métodos anteriores utilizados em análise de projetos de sistemas orientados à objeto, os principais métodos que deram origem a esta linguagem foram Booch, OMT e OOSE. (Costa, 2001)

O conceito da UML, define um conjunto básico de notações e diagramas, que conseguem representar múltiplas perspectivas estruturais, dinâmicas e estáticas comportamentais do sistema em análise e desenvolvimento. (Costa, 2001)

Podemos citar os diagramas de Uses Cases, Classes, interações, atividades e de estado e transição como alguns dos diagramas aplicados nesta linguagem. (Costa, 2001)

A UML, não é considerada uma linguagem de programação e sim uma linguagem de modelagem, que tem por objetivo auxiliar os engenheiros de software a definirem características dos sistemas como, comportamento, dinâmica dos processos, estrutura lógica, e até mesmo necessidades físicas do equipamento. (GUEDES, 2009)

A real necessidade de modelar um software, e conseguir determinar toda a necessidade de um sistema de informação, conseguindo envolver análises de questões completamente complexas como análise de requisitos, prototipação, tamanho do projeto, complexidade, prazos, custos, documentação, manutenção, entre outros. (GUEDES, 2009).

Outra ferramenta importante no desenvolvimento de software é o gerenciamento de versões que pode ser feito através do Git, que foi desenvolvido por Linus Torvalds, criador do kernel(núcleo) do Linux.

3.6.2 GIT

O Repositório GIT, é uma ferramenta utilizada para desenvolvimento de projetos, pelo fato de ser mais orientado a decisões internas que refletem diretamente no modo de utilização. O GIT oferece inicialmente um detalhamento razoável de pontos chaves, podendo ser utilizado de forma mais rápida e confiante (SILVERMAN, 2013).

De acordo com Silverman (2013), um projeto GIT, contém um histórico completo do projeto desde a sua concepção, um repositório, consiste em uma coleção de arquivos e diretórios, que podemos chamar de “commits”.

O GIT é um sistema aberto, e pode ser usado para controlar qualquer versão de qualquer formato, as suas vantagens são a velocidade, não precisar ter acesso direto ao servidor, gerencia e unifica modificações simultâneas ao mesmo arquivo.

A organização da equipe nas tarefas é uma parte muito importante, podendo ser feito através do aplicativo Trello.

3.6.3 TRELLO

Para o gerenciamento de projeto, devido seu visual simplificado, gratuidade e flexibilidade, o aplicativo Trello, é uma plataforma simples e pode ser personalizada para organizar quaisquer atividades. (SOUZA e MELLO, 2018)

O aplicativo Trello apresenta uma proposta de ser suficientemente flexível para cobrir uma variedade grande de tipos de projeto, desde um desenvolvimento de software até o planejamento de um evento, ele procura deixar você mais focado naquilo que importa. (GOULART, GODININHO, MARTINO, 2018)

O Trello otimiza a eficiência dos fluxos de trabalho, e permite incluir anexos aos cartões de tarefas e o compartilhamento de arquivos externos entre as equipes. O Trello também permite integrar com os principais serviços de nuvem como, google drive, dropbox e one drive. (GOULART, GODININHO, MARTINO, 2018)

Cada funcionalidade de um software deve ser testada após sua implementação, isto pode ser feito de maneira automatizada através dos testes unitários.

3.6.4 TESTES UNITÁRIOS

O teste de software é um processo realizado para avaliar se as especificações e funcionamento estão corretos para o ambiente o qual foi projetado (NETO, 2007).

O nível dos testes unitários tem como objetivo, explorar a menor unidade do projeto, procurando por falhas que são causadas por defeitos de lógica e de implementação, separadamente em cada módulo. (NETO, 2007).

A biblioteca para realização de testes unitários do Java é chamada de JUnit, normalmente já pode ser encontrada dentro das principais IDEs da linguagem. A figura 5 demonstra os testes realizados para retornar os clientes.

Figura 5. Teste unitário do método que retorna todos os clientes registrados

```
public void stage4_testFindAll() {
    System.out.println("\ntestFindAll() rodando...\n");

    ClientesDAO dao = new ClientesDAO();

    System.out.println("Procurando...\n");
    List<Clientes> clientes = dao.findAll();

    // Verificando acertos
    assertFalse(clientes.isEmpty());

    System.out.println("\nTodos os dados encontrados\n");

    // Mostrando dados
    for (Clientes c : clientes) {
        System.out.println("\nID : " + c.getId());
        System.out.println("\nNome : " + c.getNome());
        System.out.println("\nEndereco : " + c.getEndereco());
        System.out.println("\nTelefone : " + c.getTelefone());
        System.out.println("\nEmail : " + c.getEmail());
        System.out.println("-----");
    }

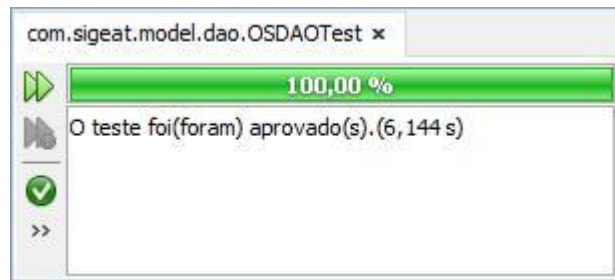
    System.out.println("\nPassou\n");

    System.out.println("testFindAll() concluido.\n");
}
```

Fonte: Próprios autores

Está sendo verificado se não estamos recebendo uma lista de clientes vazia, o que acarretaria em um erro no teste, erros indicam que o método testado está com problemas no seu código e deve ser verificado. Os testes não são capazes de identificar erros de mau uso por parte do usuário do software, bem como é impossível testar todas entradas ou saídas possíveis, dependendo do método a ser testado isso seria computacionalmente inviável. A figura 6 mostra o resultado do teste do SIGEAT.

Figura 6. Resultado aprovado de um teste unitário do SIGEAT



Fonte: Próprios autores

4 METODOLOGIA

A seguir é apresentada a metodologia para o desenvolvimento deste trabalho, ela tem como base diversos processos de engenharia de software.

4.1 ANÁLISE E LEVANTAMENTO DE REQUISITOS

Abaixo temos uma persona para especificação de requisitos do SIGEAT:

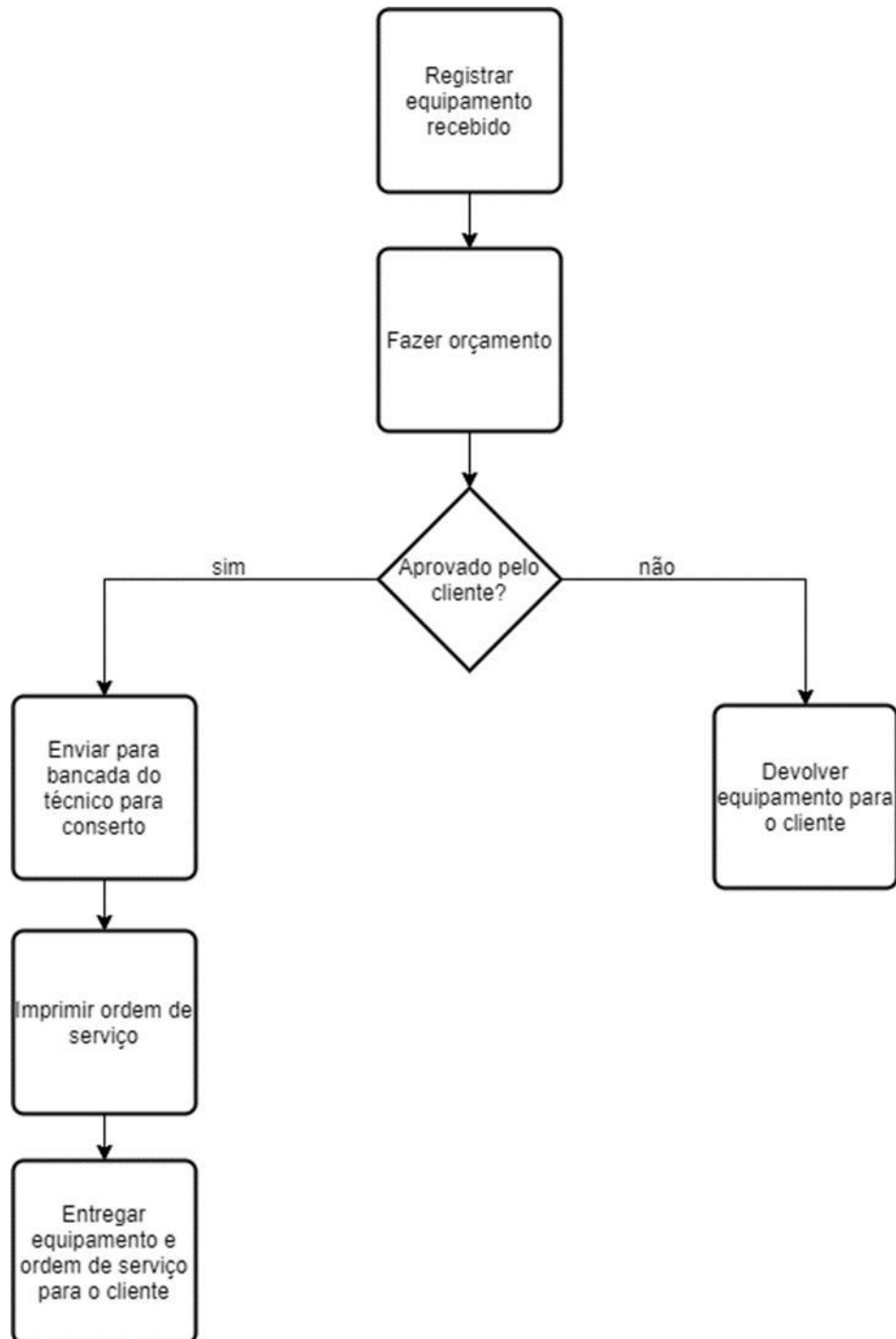
Paulo é um gerente de uma assistência técnica especializada em equipamentos de informática, como computadores, notebooks, smartphones, tablets, monitores, impressoras e videogames. Ele possui ao todo 4 funcionários, além dele mesmo, uma atendente e três técnicos. Ele gostaria de automatizar todo processo de sua assistência técnica para agilizar seu trabalho. Todo equipamento que chegar deve passar pelo seguinte processo:

- Deve ser registrado em uma base de dados informando seu defeito para análise e o cliente que solicitou o orçamento. Também deve ser registrado qualquer dano que o equipamento possua antes de ser recebido.
- Se aprovado o orçamento, o equipamento vai para a bancada para ser reparado por um dos técnicos, isto deve ser registrado, bem como as correções que foram feitas. Se necessitar de correções que não foram acordadas com o cliente, deve passar pela etapa de aprovação novamente, tendo que atualizar seus dados.
- Após o término no conserto o equipamento pode ser entregue ao cliente bem como a ordem de serviço com uma garantia de 90 dias do conserto feito. Conforme o código de defesa do consumidor, todas as informações devem estar claras neste documento.

Paulo gostaria de poder gerar relatórios para sua própria análise, entretanto somente ele pode ver estes relatórios, seus funcionários não podem ter acesso a esse tipo de informação no sistema.

Abaixo temos uma figura (7) de fluxograma ilustrando todo o processo que um equipamento passa na assistência técnica de Paulo.

Figura 7. Fluxograma de processo da assistência técnica de Paulo.



Fonte: Próprios autores

Restrições do Sistema:

- Não pode haver nenhuma OS sem cliente vinculado.
- O sistema deverá gerar automaticamente data e hora da emissão da OS.
- Somente o gestor pode ter acesso ao relatório de serviços.

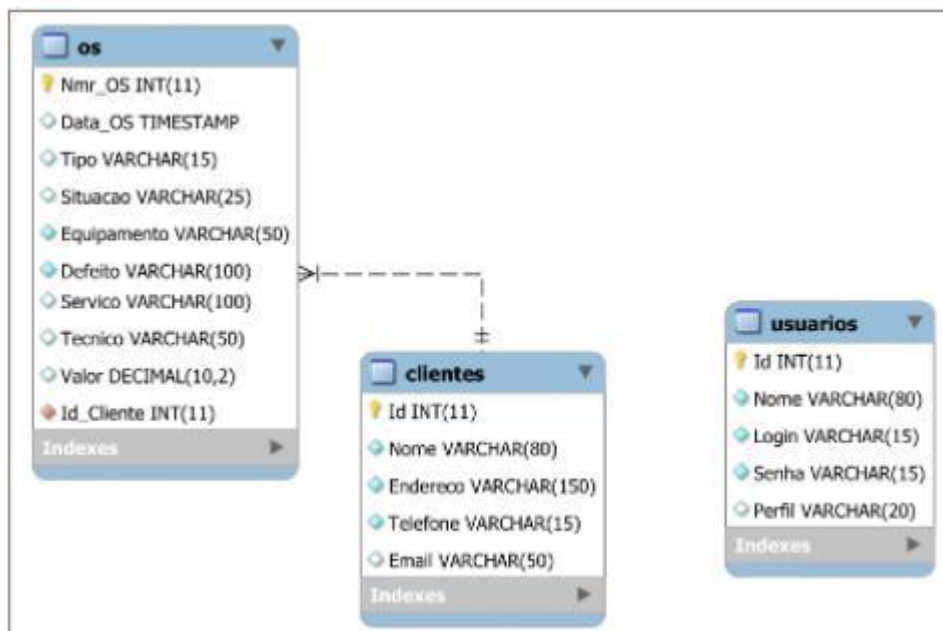
4.2 MODELO DO BANCO DE DADOS

Para atender os requisitos especificados, o SIGEAT conta com um modelo de dados com 3 entidades:

1. A entidade usuarios gerencia os usuários do sistema, não concedendo permissão de acesso total a qualquer usuário; 2. A entidade clientes gerencia todos os clientes, contendo seus principais dados para agilizar o processo de atendimento; e 3. A entidade os gerencia as ordens de serviço para os equipamentos que entrarem na assistência técnica. Toda os deve estar vinculada à um cliente.

Na figura abaixo temos um diagrama entidade-relacionamento representando a estrutura da base de dados do SIGEAT.

Figura 8. Diagrama Entidade-Relacionamento do SIGEAT



Fonte: Próprios autores

4.3 DIAGRAMA DE CASOS DE USO

O diagrama de caso de uso é uma ferramenta da UML muito importante no desenvolvimento de softwares. Cada caso de uso representado contém um requisito

funcional do sistema. Ele acompanha o desenvolvimento do software do início à conclusão sendo ferramenta de consultas, acertos, discussões, reuniões, alterações em requisitos e desenhos (MEDEIROS, 2004).

Os atores são componentes muito importantes dentro dos casos de uso, eles podem representar os usuários do sistema(pessoas), outros sistemas, ou entidades externas (não somente pessoas). Em um diagrama de casos de uso o ator é representado por um “boneco palito”, logo abaixo é indicado um nome para ele.

O caso de uso representa as ações que serão feitas por um ator dentro do contexto do sistema, ou seja, os requisitos funcionais. Ele é representado por uma elipse contendo o nome do caso de uso, esta elipse deve ser ligada ao ator no qual o caso de uso se aplica.

Abaixo temos figuras ilustrando como deve ser a relação entre um ator (figura 9a) e um caso de uso dentro de um diagrama de casos de uso (figura 9b):

Figura 9a. Representação de um ator em um diagrama de casos de uso.

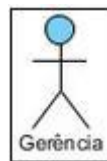
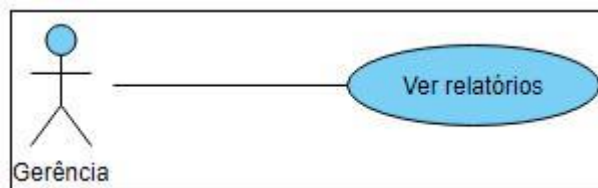


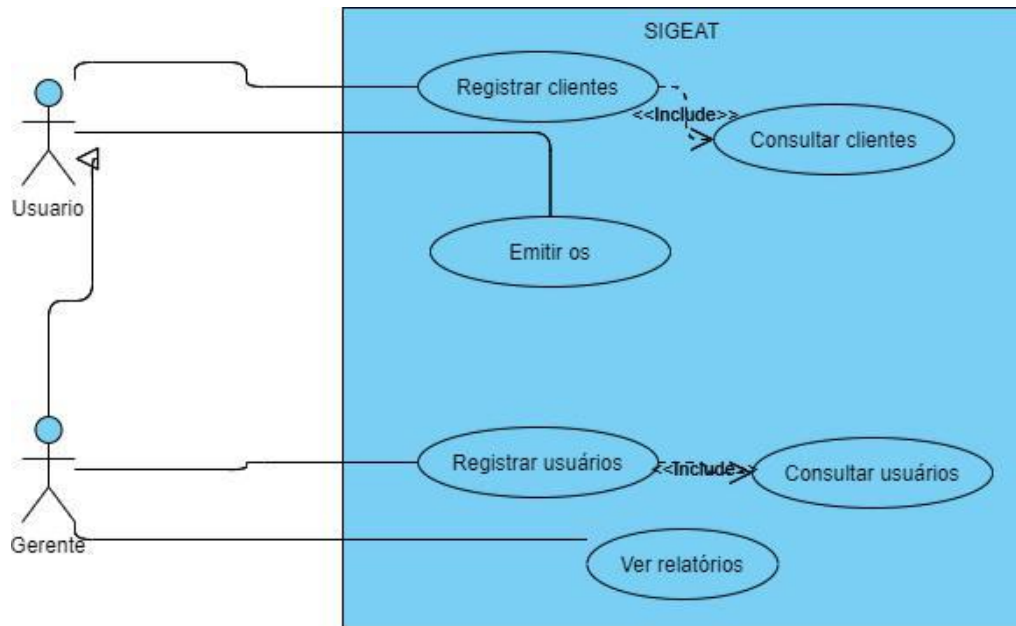
Figura 9b. Caso de uso associado a um ator.



Fonte: Próprios autores

O diagrama de caso de uso do SIGEAT resume as interações entre os usuários e o sistema. Logo abaixo temos uma figura representando este diagrama de casos de uso:

Figura 10. Diagrama de casos de uso do SIGEAT.



Fonte: Próprios autores

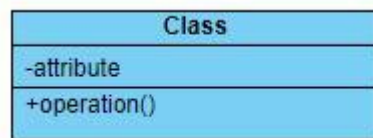
O Gerente possui todos os casos de uso de Usuário. O uso do <Include> representa uma dependência entre casos de uso, o caso de uso no fim da seta depende do anterior, não podendo ser executado sem que o anterior já tenha sido executado pelo menos uma vez.

4.4 DIAGRAMA DE CLASSES

O diagrama de classes é um dos mais importantes e mais utilizados da UML. Seu principal enfoque está em permitir a visualização de classes que compõem o sistema e seus respectivos atributos e métodos, bem como demonstrar como as classes do diagrama se relacionam, complementam e transmitem informações entre si (GUEDES, 2009).

No diagrama de classes temos algumas representações importantes de símbolos. A classe é representada por um retângulo que é subdividido em partes, temos o nome da classe, seus atributos e suas operações(métodos). Abaixo temos uma figura representando a estrutura de uma classe em um diagrama de classes.

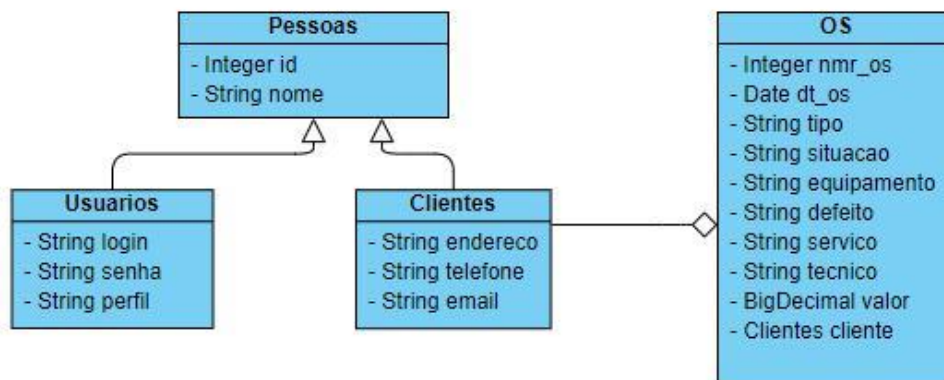
Figura 11. Representação de uma classe em um diagrama de classes.



Fonte: Próprios autores

Abaixo temos o diagrama de classes do SIGEAT, as classes não se encontram devidamente ligadas pois o diagrama está sendo apresentado em partes. Na figura 12 temos representado o bean da camada de modelo (mapeamento objeto-relacional).

Figura 12. Classes de mapeamento objeto-relacional do SIGEAT.

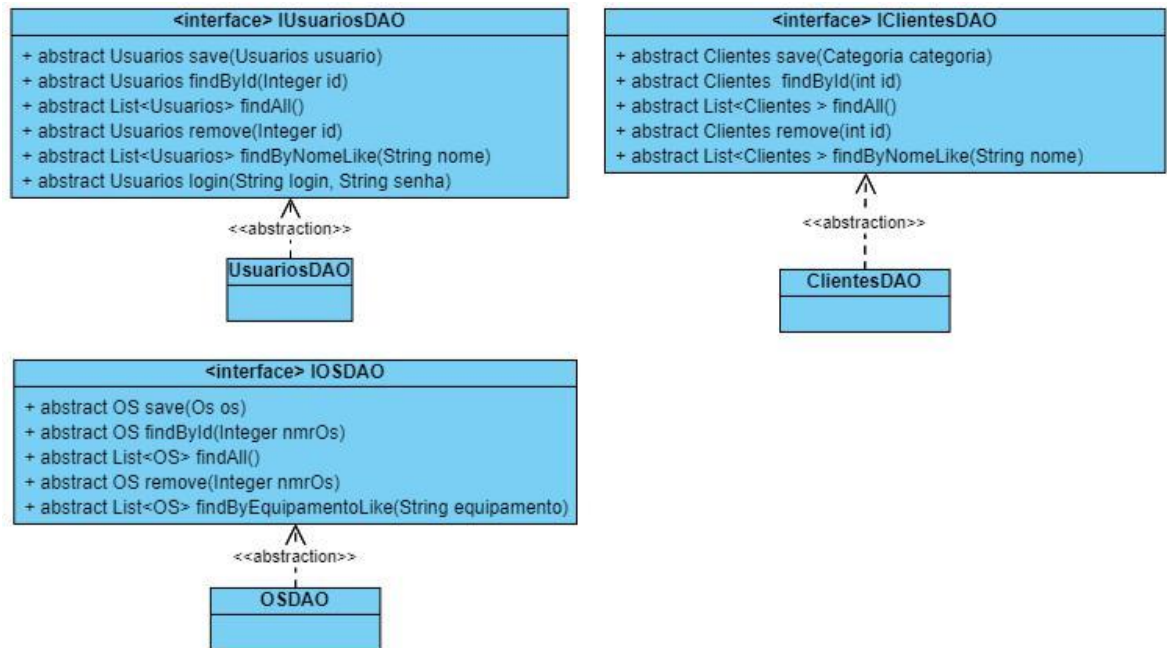


Fonte: Próprios autores

As classes Usuarios e Clientes herdam atributos da classe Pessoas (Superclasse). A classe OS agrega a classe Clientes.

Na figura 13 temos representados os objetos de acesso aos dados (DAO) da camada de modelo.

Figura 13. Classes de acesso aos dados.

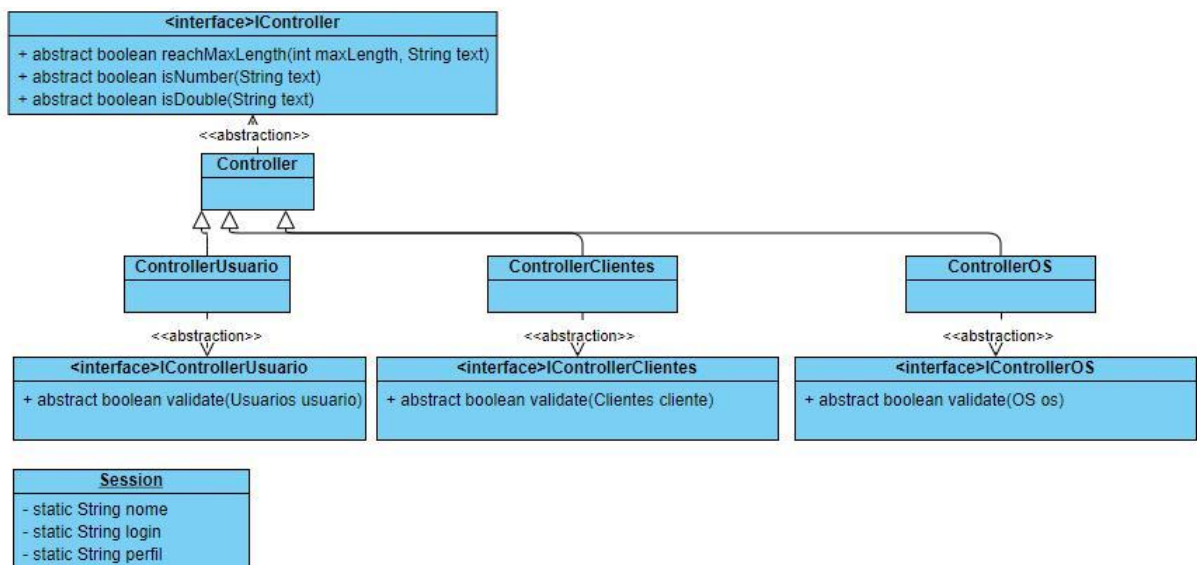


Fonte: Próprios autores

Como observado na figura os métodos de acesso aos dados foram declarados em uma interface pública e implementados em classes, cada objeto-relacional possui sua interface e sua classe. Por convenção foi adicionada a letra “I” ao nome de cada interface para não haver eventuais ambiguidades entre as classes e as interfaces.

Na figura 14 temos representada a camada de controle do SIGEAT.

Figura 14. Classes de controle do SIGEAT.

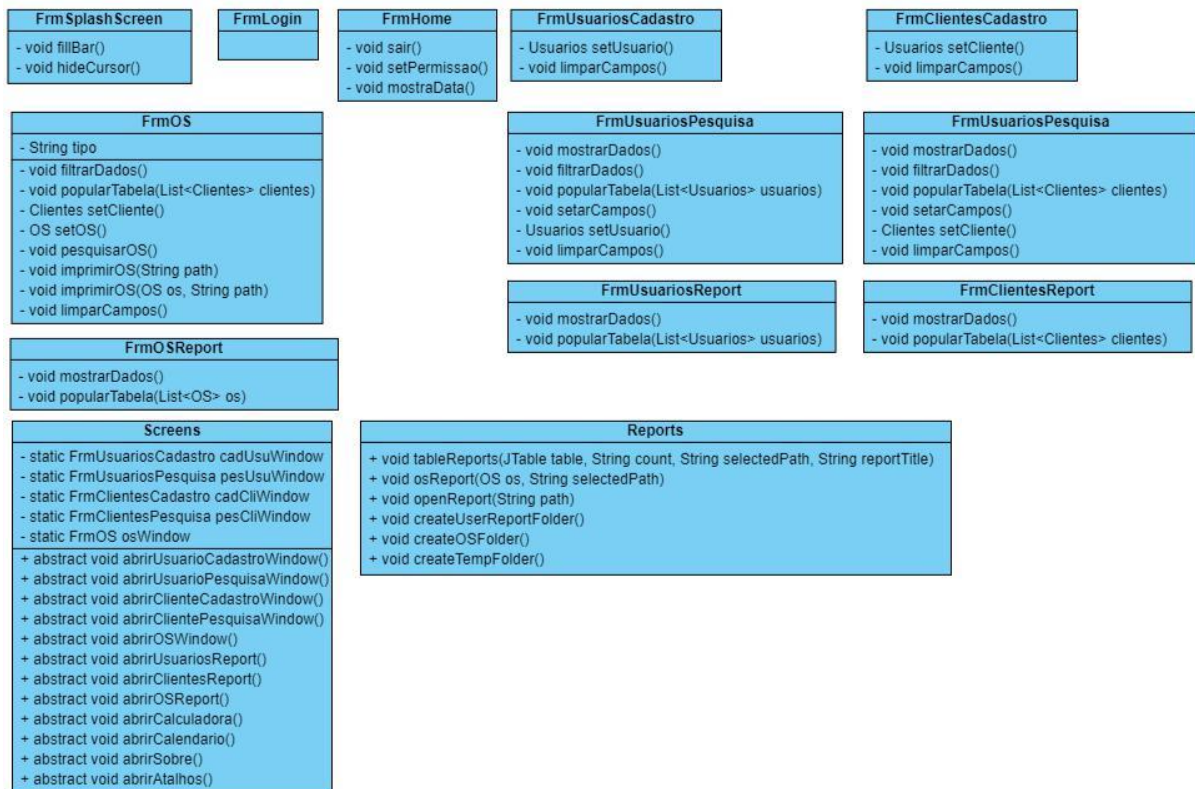


Fonte: Próprios autores

Temos diversas interfaces sendo implementadas, cada objeto-relacional possui sua classe de controle, também é utilizada a herança para métodos em comum, temos uma classe abstrata para armazenar alguns dados da sessão do usuário no sistema.

Na figura 15 temos representada a camada de aplicação do SIGEAT.

Figura 15. Classes da camada de aplicação do SIGEAT.



Fonte: Próprios autores

A camada de aplicação contém todos os formulários e elementos visuais do SIGEAT, sendo responsável pelo que é exibido ao usuário, também possui duas classes, uma responsável pelas janelas da aplicação e outra responsável pelos relatórios. Na classe FrmOS temos o conceito de **polimorfismo** aplicado, onde há sobrecarga de um método. Mais detalhes e documentações podem ser vistos no repositório do SIGEAT.

5 RESULTADOS/DISCUSSÕES

A seguir temos os resultados do processo de desenvolvimento deste trabalho, todas as funcionalidades serão apresentadas bem como o software desenvolvido.

5.1 INTRODUÇÃO AO SIGEAT:

O SIGEAT (Sistema de Gerenciamento de Assistências Técnicas) é um software desenvolvido utilizando a linguagem de programação Java, o sistema de banco de dados para armazenamento e consulta de registros utiliza o MySQL, sendo uma solução focada em assistências técnicas em equipamentos de informática. Será apresentada a versão 1.0.0 do sistema, várias funcionalidades poderão ser adicionadas futuramente, porém o software já se mostra completamente funcional nesta sua primeira versão. Software sob licença GPL 3.0.

5.2 APRESENTAÇÃO DO SIGEAT

Para o correto funcionamento do SIGEAT são necessários o cumprimento de alguns requisitos, o usuário deve ter instalado em seu computador o JRE(Java Runtime Environment) 8 e o MySql 5.5 ou superior, detalhes podem ser encontrados no repositório do projeto no Github.

Abaixo serão apresentadas algumas *screenshots* do SIGEAT, apresentando suas principais funcionalidades.

Na figura 16 é apresentada a tela de login do SIGEAT, somente usuários cadastrados no banco de dados do sistema podem utilizá-lo, existe um usuário padrão para que o sistema possa ser acessado pela primeira vez, este usuário padrão tem total acesso do sistema, por questões de segurança não deve ser compartilhado com outros usuários.

Figura 16. Tela de login do SIGEAT.



Fonte: Próprios autores

Na figura 17 é apresentada a tela de cadastros de novos usuários, apenas usuários administradores podem visualizar esta tela.

Figura 17. Tela de cadastro de novos usuários do SIGEAT.

The screenshot shows the 'Novo Usuário' window in the SIGEAT application. The title bar includes 'Usuários, Clientes, Serviços, Relatórios, Ferramentas, Ajuda, Opções'. The main content area has a form with the following fields: 'Nome*' (text), 'Login*' (text), 'Senha*' (password), and 'Perfil*' (dropdown menu currently showing 'usuário'). At the bottom of the form are two buttons: 'Cadastrar' and 'Limpar'.

Fonte: Próprios autores

Na figura 18 é apresentada a tela de pesquisa de usuários do sistema, nesta tela também é permitida a atualização de dados de usuários e também exclusão, assim como a tela anterior apenas usuários administradores podem visualizar esta tela.

Figura 18. Tela de pesquisa de usuários do SIGEAT.

The screenshot shows the 'Pesquisar Usuário' window in the SIGEAT application. The title bar includes 'Usuários, Clientes, Serviços, Relatórios, Ferramentas, Ajuda, Opções'. The main content area has a search form with a 'Nome' field and a magnifying glass icon. Below the search form is a table listing users with columns: ID, Nome, Login, Senha, and Perfil. The table contains four rows of user data. Below the table are fields for 'ID', 'Nome*', 'Login*', 'Senha*', and 'Perfil*' (dropdown menu currently showing 'usuário'). At the bottom of the form are three buttons: 'Atualizar', 'Limpar', and 'Excluir'.

ID	Nome	Login	Senha	Perfil
1	Paulo Silva	psilva	123	administrador
2	Pamella Silva	pam1234	123	usuário
3	Anderson Martins	anderson	45789	usuário
4	Genivaldo Santana	genivaldo	456789	usuário

Fonte: Próprios autores

Na figura 19 é apresentada a tela de cadastro de novos clientes, qualquer usuário com acesso ao sistema pode visualizar esta tela.

Figura 19. Tela de cadastro de clientes do SIGEAT.

The screenshot shows a web application window titled 'SIGEAT'. The main content area is titled 'Novo Cliente'. It contains four text input fields labeled 'Nome*', 'Endereço*', 'Telefone*', and 'Email'. Below these fields are two buttons: 'Cadastrar' (blue) and 'Limpar' (blue). The top of the window shows a menu bar with 'Usuários', 'Clientes', 'Serviços', 'Relatórios', 'Ferramentas', 'Ajuda', and 'Opções'. The top right corner displays the user 'admin', the date 'Sábado, 13/11/2021', and the time '21:54:49'.

Fonte: Próprios autores

Na figura 20 é apresentada a tela de pesquisa de clientes, nesta tela também é permitida a atualização de dados de clientes e também exclusão, assim como na tela anterior qualquer usuário com acesso ao sistema pode visualizar esta tela.

Figura 20. Tela de pesquisa de clientes do SIGEAT.

The screenshot shows a web application window titled 'SIGEAT'. The main content area is titled 'Pesquisar Cliente'. It features a search bar labeled 'Nome:' with a magnifying glass icon. Below the search bar is a table with the following data:

ID	Nome	Endereço	Telefone	Email
1	Marcelo Antônio	Rua Maria 594	3333-4444	marcelo@ema...
2	Ana Paula de Oliveira	Rua Pinheiro...	3544-5555	anapaula555...
3	Ulisses Passos	Rua Adriana...	3210-4455	
4	Karla Santana	Rua Dora, 44	5588-9999	karla@ema...
5	Amélia da Silva	Rua Oléu, 45	9999-0000	amelias2@e...
6	Augusto Henrique	Rua Tere, 66	8888-5522	augustinho2...

Below the table, there is a 'Total: 6' label. Underneath the table are five input fields: 'ID:', 'Nome*', 'Endereço*', 'Telefone*', and 'Email'. At the bottom are three buttons: 'Atualizar' (blue), 'Limpar' (blue), and 'Excluir' (red). The top of the window shows the same menu bar and user information as Figure 19.

Fonte: Próprios autores

Na figura 21 é apresentada a tela de OS (ordens de serviço), esta é a principal tela do SIGEAT, responsável por cadastrar, atualizar, pesquisar e excluir os dados referentes aos equipamentos dos clientes para conserto, como apontado nas restrições do sistema, não é possível cadastrar uma OS sem um cliente vinculado. Qualquer usuário com acesso ao sistema pode visualizar esta tela.

Figura 21. Tela de OS do SIGEAT.

Fonte: Próprios autores

Na figura 22 temos uma amostra de um relatório de clientes, este relatório foi gerado em formato pdf e não pode ser gerado por usuários comuns.

Figura 22 :Relatório de clientes.

23/11/2021 - 18:40:15

SIGEAT
demonstração

Clientes Cadastrados

Relatório de consulta à base de dados

Id	Nome	Endereço	Telefone	Email
1	José da Silva	Rua Quatro 234 - A-City	3366-4555	
2	Angela Muniz	Rua Oito 344 - D-City	92220-1000	angelm09@gmail.com
3	Carlos Faundes	Rua Um 123 - A-City	96665-4566	carlosfag10@gmail.com
4	Paulo Fontana	Rua Cinco 234 - A-City	99995-8888	pfontana55@gmail.com
6	Marcia Araújo	Rua Cinco 234 - A-City	5522-8855	


Total : 5

Fonte: Próprios autores

Na figura 23 temos uma amostra de uma ordem de serviço (OS), documento que deve ser impresso e acompanhar o equipamento recebido e no final do processo deve ser entregue ao cliente como certificado de garantia do serviço executado.

Figura 23. Documento de ordem de serviço.

23/11/2021 - 21:30:23



SIGEAT

demonstração

Ordem de serviço

Número da OS: 1

Tipo: Orçamento

Data da OS: 22/11/2021

Cliente: José da Silva

Equipamento: Notebook Acer

Defeito: Travando

Serviço: Formatação

Técnico: Fabio Faria

Valor: R\$100,00

GARANTIA DE 90 DIAS - NÃO COBRE MAU USO

Fonte: Próprios autores

6 CONSIDERAÇÕES FINAIS

O ato de computar, apesar de ser inerente à própria história da humanidade, foi sofrendo modificações e melhorias que visavam facilitar essa ação. Essa evolução chegou a um tal ponto em que foram criados mecanismos elétricos para isso, os computadores. O avanço tecnológico, somado ao barateamento da produção levou à popularização da tecnologia, que por sua vez possibilitou uma verdadeira revolução na vida cotidiana das pessoas.

Essa democratização tecnológica, contudo, teve também como consequência o aumento de problemas relacionados a estes mecanismos. Neste trabalho, o ponto focal entre estes problemas foi a demanda por resolução de problemas relacionados à sua estrutura e funcionamento que precisem de reparos e consertos em lugares especializados, as assistências técnicas. A procura por este tipo de estabelecimento

cresceu exponencialmente, em consonância com o número de equipamentos. Com o aumento da demanda, as assistências técnicas passaram a enfrentar o desafio de gerenciar o negócio de forma a otimizar tempo e custos, sem abrir mão da excelência no atendimento e execução dos serviços.

Este nicho de mercado é composto, em sua maioria, por empresas de pequeno porte, com poucos funcionários, sendo que poucos têm real conhecimento na área de gestão de negócios ou de fluxo de processos, sendo, via de regra, pessoal especializado da parte técnica apenas.

Diante deste cenário, foi proposta a construção de um software, que ajude neste gerenciamento, e com base em um tripé básico de gestão: dados referentes aos clientes, dados dos usuários internos, e às ordens de serviço associadas à ambos. Este software foi nomeado SIGEAT (Sistema de Gerenciamento de Assistências Técnicas). A construção do software se deu de forma orgânica, baseada em experiências empíricas dos componentes do grupo

Ao final da fase de testes do software, chegou-se à conclusão de que ele atende a todos os requisitos levantados na fase de concepção, tornando possível adicionar aumentar os padrões de qualidade e de possibilidade de informações aos clientes referentes aos seus produtos e, ao mesmo tempo, reduzindo ao máximo possíveis transtornos aos clientes, elevando a relação de confiança e fidelidades destes para com a assistência técnica de Paulo.

Contudo, é preciso também fazer melhor avaliação da cultura organizacional de onde se possa vir aplicar este software, não apenas na assistência de Paulo. O compromisso com os clientes, juntamente com a quebra de paradigmas estabelecidos e a criação de novas estratégias, dentre elas a utilização deste software, é que trarão as vantagens competitivas. A utilização do software por si, ajuda, mas não resolve todas as questões.

Por fim, pode-se dizer que os objetivos propostos neste trabalho foram alcançados, criando-se uma ferramenta poderosa no auxílio à gestão de assistências técnicas, havendo, todavia, espaço para melhorias e atualizações.

7 REFERÊNCIAS

- BERTOLINI, C. et. al, **LINGUAGEM DE PROGRAMAÇÃO I**. Primeira edição. Universidade Federal de Santa Maria. Rio Grande do Sul, 2019.
- COSTA, C. **Gestão & Produção**. A aplicação da linguagem de modelagem unificada (UML) para o suporte ao projeto de sistemas computacionais dentro de um modelo de referência, v.8, n.1, abr. 2001
- DA SILVA, I. **Desenvolvendo com o Hibernate**. Devmedia, 2009. Disponível em: <<https://www.devmedia.com.br/desenvolvendo-com-hibernate/14756>>. Acesso em: 09 out. 2021.
- MEDEIROS, E. **Desenvolvendo Software com UML 2.0 : Definitivo**. São Paulo: Pearson, 2004.
- DIANA, D. **História e Evolução dos Computadores**. Toda Matéria, 2019. Disponível em: <<https://www.todamateria.com.br/historia-e-evolucao-dos-computadores/>> . Acesso em: 30 set. 2021.
- GOULART, A. GODINHO, T. MARTINO, L. **COMUNICON2018**. “Seu segundo cérebro”? Aplicativos de produtividade, disciplina pessoal e precarização do trabalho. ESPM, 2018.
- GPJ Informatica, **História da computação**, 2018. Disponível em: <<https://gpjinformatica.wordpress.com/2018/05/22/historia-da-computacao/>>. Acesso em: 30 set. 2021.
- GUEDES, G. **UML 2**. Uma abordagem prática. Primeira edição. São Paulo: Novatec, 2009.
- MAGALHÃES, Ivan Luizio; PINHEIRO, Walfrido Brito. **Gerenciamento de serviços de TI na prática: uma abordagem com base na ITIL: inclui ISO/IEC 20.000 e IT Flex**. 2007. São Paulo - SP: Novatec.
- MELLO, Anna Carolina; SOUZA, Luiz Henrique Gomes de. **Solução Simplificada para o Monitoramento e Controle de Projetos Utilizando a Ferramenta Trello**. Boletim do Gerenciamento, [S.l.], v. 2, n. 2, out. 2018. ISSN 2595-6531. Disponível em: <<https://nppg.org.br/revistas/boletimdogerenciamento/article/view/35>>. Acesso em: 02 nov. 2021.
- Tecnologia e Informação, **A história do Java.**, 2019. Disponível em: <https://tecnologiaeinformacao.netlify.app/_pages/java-b/intro.html>. Acesso em: 01 out. 2021.
- TIC Domicílios. (2020). **Pesquisa sobre o uso das tecnologias de informação e comunicação nos domicílios brasileiros: TIC Domicílios 2020**. Disponível em: <<https://cetic.br/pt/pesquisa/domicilios/indicadores/>>. Acesso em 29 out. 2021.

NAVATHE, S. B., RAMEZ, E. **Sistemas de Bancos de Dados**. 4. ed. São Paulo: Pearson, 2005.

NETO, A. **Revista Engenharia de Software**. Introdução a Teste de Software. Primeira edição, 2007)

PACIEVITCH, Y. **MySQL**. InfoEscola, 2011. Disponível em: <<https://www.infoescola.com/informatica/mysql/>>. Acesso em: 02 out. 2021.

SILVERMAN, R.. **Git Guia Prático**. Primeira edição. São Paulo: Novatec, 2013.

SOMMERVILLE, I. **Engenharia de Software**. 9. ed. São Paulo: Pearson, 2011.

ZUCHER, V. **O que é padrão MVC? Entenda arquitetura de softwares!**. Le Wagon, 2020. Disponível em: <<https://www.lewagon.com/pt-BR/blog/o-que-e-padrao-mvc>>. Acesso em: 06 out. 2021.

GLOSSÁRIO

Atributos: Dados que fazem parte de um objeto.

Classe: Modelo usado para criar objetos dentro do paradigma de POO.

Constraints: Restrições de dados que serão aceitos em uma coluna de um banco de dados relacional.

CRUD: Operações básicas de um banco de dados, create (adicionar novo registro), read (ler registros), update (atualizar registro) e delete (excluir registro).

DAO: Data Access Object (Objeto de Acesso aos Dados). Objeto instanciado com o propósito de acessar uma base de dados.

Encapsulamento: Um dos pilares da programação orientada a objeto, onde uma classe não pode alterar diretamente os atributos de outra classe.

Frameworks: Conjunto de bibliotecas e componentes contendo interfaces, classes, atributos e métodos que realizam uma determinada tarefa e podem ser implementados em vários projetos.

Herança: Um dos pilares da programação orientada a objeto, onde uma classe pode passar seus atributos e métodos para outra classe.

IDE: Integrated Development Environment (Ambiente de Desenvolvimento Integrado). Conjunto de ferramentas de desenvolvimento de software para uma ou mais linguagens de programação.

Métodos: Trechos de códigos que representam comportamento ou ações de um objeto, são implementados em uma classe.

Polimorfismo: Um dos pilares da programação orientada a objeto, onde um mesmo método pode ter diferentes implementações.

POO: Programação Orientada a Objeto. Paradigma de programação que aproxima o manuseio das estruturas de um programa ao manuseio de coisas no mundo real.

Transações: Operações que fazem alterações em um banco de dados.

Variáveis: Valores que são armazenados na memória de um computador durante a execução de um programa.

ANEXOS

ANEXO A – Repositório do SIGEAT

<https://github.com/MrX456/tcc-sigeat>