UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA
COLEGIADO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

Laércio de Souza

SARA - Sistema de Alocação de Recursos Aeroportuários

Projeto de Graduação apresentado ao Departamento de Informática da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Orientador: **Prof. M.Sc. Jadir Eduardo Souza Lucas**

VITÓRIA
2021

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA
COLEGIADO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

Laércio de Souza

SARA - Sistema de Alocação de Recursos Aeroportuários

COMISSÃO EXAMINADORA

_____
Prof. M.Sc. Jadir Eduardo Souza Lucas

_____
Prof. phD. José Gonçalves Pereira Filho

_____
Prof. Ph.D. Rodrigo Laiola Guimarães

Vitória, _____ de 2021

Aos meus pais, José e Maria por sempre acreditarem no meu potencial e nunca medirem esforços para que eu pudesse avançar nos estudos.

A minha irmã Elizabeth que sempre esteve comigo nos momentos mais difíceis

Ao Senhor Anísio Silva, sem o apoio do qual eu não chegaria até aqui.

Agradecimentos

A Deus pelo zelo que tem com minha vida e por ter me dado os recursos físicos e emocionais para vencer as dificuldades que surgiram pelo caminho.

A minha Esposa Denecy pelo companheirismo, pelos momentos em que precisou privar da minha companhia, pelo apoio incondicional e por sempre ter acreditado em mim.

A minha filha Andressa pelo tempo que esteve privada da minha companhia por ela ser a razão de eu nunca desistir nas horas mais angustiantes.

Aos professores do Projeto Universidade para Todos dos anos 1998, 1999, 2000 e 2001 assim como toda a equipe de apoio e os patrocinadores, pelo apoio, ensino e incentivo, sem os quais não conseguiria vencer o Vestibular. Aqueles aulões nos fins de semana e véspera de vestibuar, os quais além de divertirem, revisavam conteúdo e reduziam o stress, foram fundamentais para o sucesso.

Aos Companheiros de trabalho da Infraero no Aeroporto de Vitoria-ES pelas trocas de turno e pela boa amizade que foram fundamentais nesta caminhada.

A minha família, meus pais e meus irmãos e irmãs que foram meu porto seguro nos momentos de maior angustia e dificuldade além de sempre me incentivar a alçar os voos mais altos.

A Maria Vânia Amorim e família por sempre terem acreditado em mim e me incentivado.

A cada um dos professores pelo tempo que dedicaram inclusive fora da sala de aula para sanar minhas duvidas e também por seu alto nível de exigência nos trabalhos e provas, fundamentais a qualidade da minha formação.

A cada um dos alunos com quem formei grupo de trabalho e com aqueles que levarei comigo por toda a vida.

A todos aqueles que mesmo sem saber me ajudaram a angariar este titulo de Bacharel em Ciência da Computação.

MUITO OBRIGADO.

Há homens que lutam um dia e são bons, há outros que lutam um ano e são melhores, há os que lutam muitos anos e são muito bons. Mas há os que lutam toda a vida e estes são imprescindíveis.

(Berthold Brecht)

LISTA DE FIGURAS

LISTA DE SIGLAS

ANAC – Agência Nacional de Aviação Civil

API – Application Programming Interface

ASEB – Aeroportos Sudeste do Brasil S.A.

BDO – Banco de Dados Operacional

CDP – Componente de Domínio do Problema

COA – Centro de Operações Aeroportuárias

CRUD – Acronimo do Inglês Create, Read, Update and Delete

DAC – Departamento de Aviação Civil

DAO – Acronimo do Ingês Data Access Object

EER – Diagrama Entidade-Relacionamento Extendido

ER – Entidade e Relacionamentos

GUI – Grafical User Interface

HOTRAN – Horário de Transporte

IDE – Integrated Development Environment

IEEE – Institute of Eletrical and Eletronic Engineers

IHC – Interface Humano Computador

INFRAERO – Empresa Brasileira de Infraestrutura Aeroportuária

JDBC – Java Database Connectivity

JPA – Java Persistence API

MVC – Model, View, Controller

MYSQL – Sistema de Gerenciamento de Banco de Dados

ORM – Objecting-Relacional Mapping

RAD – Rapid Aplication Development

RBAC – Regulamento Brasileiro de Aviação Civil

SARA – Sistema de Alocação de Recursos Aeroportuários

SBVT – Aeroporto de Vitória – Eurico de Aguiar Salles

SISO – Sistema Inegrado de Soluções Operacionais

SIV – Sistema Informativo de Voos

TWR – Torre de Controle de Aeródromo

UML – Unifield Modeling Language

SUMÁRIO

RESUMO

O objetivo deste trabalho é desenvolver um software para alocação de recursos às aeronaves que pousam e decolam no Aeroporto de Vitoria, onde sempre que possível utiliza-se software livre para desenvolvimento dos subprodutos, além disso o programa sera implementado aplicando alguns dos conhecimentos de engenharia de software. O resultado é um programa que atende à proposta inicial e pode permitir que o raciocínio empregado em seu desenvolvimento seja aplicado a outros modais de transporte onde seja necessária uma boa politica de alocação de recursos.

Palavras chave: Aeroporto, alocação, cliente, dao, estacionamento, heurísticas, jdbc, meta-heurísticas, mysql, mvc, otimização, patio, recursos.

1 Introdução

As aeronaves representam uma das grandes maravilhas tecnológicas e seu uso gera um impacto sem precedentes na atividade humana, seja em tempos de paz seja em tempos de guerra a forma com que as pessoas, os produtos e alguns animais são trasportados de um local a outro ganhou muito em velocidade e eficiência com o advento da Industria Aeronáutica. O principal produto desta industria, os aviões, são equipamentos com uma enorme tecnologia embarcada e por isso mesmo seu valor atinge facilmente a ordem de grandeza de centenas de milhões de reais.

É inegável que a aviação reduz distancias e aproxima pessoas, estes são pressupostos fundamentais em tempos de economia globalizada e intensa atividade comercial, a qual já não respeita fronteiras. Neste contexto, o transporte aéreo se coloca como fundamental para permitir as trocas comerciais e o desenvolvimento das sociedades, principalmente por agregar segurança e velocidade ao transporte. Destaca-se que o transporte aéreo é considerado o modal mais seguro de transporte.

Este modal de transporte é realizado com aeronaves, dos mais variados tamanhos e capacidades, as quais geralmente saem de um aeródromo em direção a outro. Desta forma, pode-se usufruir de uma serie de facilidades oferecidas pelos aeródromos dentre as quais poderíamos citar: pista de pouso e decolagem, pista de taxiamento, local para estacionamento, equipamentos para carga e descarga de bagagens, abastecimento de combustível, abastecimento de água, coleta de desejos, equipe de manutenção, equipe de combate a incêndios, sinalização para estacionamento, pontes de embarque, salas de embarque e desembarque, portões de embarque, esteiras de bagagem, serviço de táxi, serviço de emergência medica, apoio nas emergências aeronáuticas, comunicação, torre de controle, serviço aduaneiro e vigilância sanitária entre outros.

Os custos envolvidos na atividade aeronáutica são extremamente elevados, tanto na construção da infraestrutura quanto na sua operação e manutenção, por isso, para garantir a viabilidade dos negócios, a atividade operacional tem como um de seus requisitos relevantes a otimização do uso dos recursos disponíveis.

Observa-se, que os recursos necessários à realização eficiente e segura das atividades aeronáuticas encontram-se prioritariamente nos aeródromos e aeroportos onde as operações são realizadas. Aqui, pretendemos tratar da alocação de alguns destes recursos. Discorremos brevemente sobre o problema, com foco no aeroporto de Vitória-ES e propomos uma maneira de solucionar o mesmo automaticamente, através do uso de software.

De inicio, julgamos importante destacar os conceitos de aeródromo e de aeroporto, conforme definidos no RBAC 01 EMENDA 08 no qual temos:

Aeródromo significa uma área delimitada em terra ou na água destinada, no todo ou em parte, para pouso, decolagem e movimentação em superfície de aeronaves; inclui quaisquer edificações, instalações e equipamentos de apoio e de controle das operações aéreas, se existirem. Quando destinado exclusivamente a helicópteros, recebe denominação de heliponto(ANAC, 2021) .

Aeroporto é um aeródromo público dotado de edificações, instalações e equipamentos para apoio às operações de aeronaves e de processamento de pessoas e/ou cargas(ANAC, 2021).

A razão de existir dos aeroportos esta em oferecer recursos e facilidades às aeronaves que realizam voos de pouso ou decolagem, assim como garantir aos passageiros um bom nível de conforto e segurança.

1.1 Recursos e Facilidades

No contexto deste trabalho cabeceiras de pouso e decolagem, posição de estacionamento, portões de embarque e esteiras de restituição de bagagens são os recursos utilizados pelas aeronaves e passageiros ao realizar um voo, os quais o sistema propõe-se a gerenciar a alocação.

O aeroporto oferece diversas facilidades aos usuários a saber: lojas de conveniência, lanchonetes, restaurantes, locadoras de veículos, inspeção de passageiros, serviço medico de emergência, órgãos públicos, balcão de check-in , estacionamento de veículos, serviço de táxi e outros. Estas fogem ao escopo deste trabalho.

Até 02/01/2020 o Aeroporto de Vitoria-ES era administrado pela Infraero porém a partir do dia 03/02/2020 passou a ser administrado pela ASEB uma empresa do grupo Zurich Airport.

Em diversas circunstancias observa-se que os recursos disponíveis para a realização de uma atividade ou projeto são escassos e ampliar a quantidade de tais recursos é inviável no curto espaço de tempo.

Tratando especificamente do cliente dos aeroportos capixabas, observa-se que os processos de embarque e desembarque precisam ser ágeis e com o mínimo deslocamento a pé possível, por outro lado, ao analisar as necessidades das companhias aéreas, observa-se que numero de slots disponíveis é escasso assim como os demais recursos.

A experiencia tem mostrado que o sucesso do negócio aviação comercial, depende em elevada medida da otimização do uso de horários de pouso e decolagem (slots), posição de estacionamento no pátio de manobras, esteiras de bagagem, portões de embarque, balções de check-in e demais facilidades disponibilizadas aos usuários do aeroporto.

Portanto, apresentamos o Sistema de Alocação de Recursos Aeroportuários (SARA) o qual ambiciona colaborar para a otimização do uso das principais facilidades relacionadas à operacionalidade do aeroporto ao mesmo tempo em que reduz as chances de erro humano na alocação destes recursos.

1.2 Declaração de capacidade do Aeroporto

Segundo a Declaração de capacidade do Aeroporto de SBVT(INFRAERO, 2018) o aeroporto de Vitoria-ES possui duas pistas de pouso e decolagem a saber:

- Pista 06/24 com dimensões 1750 x 45 metros.
- Pista 02/20 com dimensões 2028 x 45 metros.

As duas pistas tem a peculiaridade de não serem paralelas e tão pouco se cruzarem. Entendemos que isto representa uma vantagem operacional pois permite operações de pouso e decolagem simultâneas, desde que em pistas diferentes.

O aeroporto possui um terminal de passageiros com aérea de 31.850 m2 e 1790 vagas para estacionamento de veículos (INFRAERO, 2018).

Além disto, conta ainda com 4 pátios com as seguintes características:
- Patio 1 – 54.480,00 m^{2-} possui 6 posições disponíveis para pernoite de aeronaves até código C2. Temos ainda 05 posições remotas disponíveis para aeronaves até código C2 e duas posições auxiliares que comportam ate aeronave de código D2 (INFRAERO, 2018).

- Pátio 2 – 21.675,00 m^{2-} disponibiliza 6 posições remotas para pernoite de aeronaves ate o código C2(INFRAERO, 2018).

- Pátio 3 – 20.145,00 m^{2-} exitem 06 posições para aeronaves de código C1 com até 29 metros de envergadura (distancia entre as pontas de asas). Temos ainda 05 posições para aeronaves de Código C2 com ate 27,05 m de envergadura. Ou então 01 posição para atendimento de aeronave de código D2. Temos que sete posições deste patio podem ser utilizadas por aeronaves de asa rotativa (Helicóptero) (INFRAERO, 2018).

- Pátio 4 – 1571,00 m^{2-} Conta com 5 posições para estacionamento de helicópteros com asa rotativa com rotor de até 20m de diâmetro (INFRAERO, 2018).

Destaca-se ainda que o aeroporto possui 06 pontes de embarque, 05 esteiras para restituição de bagagens, 06 portões de embarque (INFRAERO, 2018).

Quanto ao numero de passageiros transportados, o aeródromo de Vitoria-ES classifica-se como sendo da Classe III pois transporta um numero de passageiros superior a 1000.000 e inferior a 5.000.00 conforme padronizado no RBAC 153 EMENDA 04 aprovado conforme resolução 517 de 14 de maio de 2019.

Os códigos das aeronaves mencionados acima são conforme (ANAC, 2020) na tabela características físicas e Operacionais de Aeronaves. Este é um documento para consulta e caso o usuário necessite de informações mais precisas, deve consultar o fabricante da aeronave sobre a qual deseja informações mais detalhadas.

A ANAC é uma das agencias reguladoras federais do Brasil. Esta agencia foi criada pela lei 11.182 de 27 de setembro de 2005 e tem a competência de regular e fiscalizar as atividades da aviação civil e da infraestrutura aeronáutica e aeroportuária conforme consta em seu site institucional" (ANAC, 20/03/2021) , e passou a atuar em 2006 em substituição ao Departamento de Aviação Civil (DAC).

1.3 - Objetivos

O objetivo geral deste trabalho é projetar e desenvolver um sistema de software capaz de realizar a alocação automatizada de recursos aeroportuários (SARA) aplicando os conhecimentos adquiridos durante a graduação e agregando outros conhecimentos obtidos fora do meio acadêmico mas que possuem interface com aqueles. Serão necessários conhecimentos de projeto e desenvolvimento de software, implementação da interface com o usuário, projeto e implementação do Banco de Dados, levantamento de requisitos, análise de requisitos, documentação de requisitos, programação utilizando a linguagem java SE 8 e Mysql para implementação do Banco de Dados.

Como objetivos específicos podem ser citados:

a) Percorrer os principais passos necessários ao desenvolvimento de um projeto completo de software envolvendo as principais atividades necessárias e utilizando prioritariamente software livre.

b) Apresentar o sistema como uma proposta de melhoria do Banco de Dados Operacionais (BDO) com o qual trabalhava-se no Aeroporto de Vitoria.

c) Solidificar os conhecimentos de Engenharia de Software, projeto de algoritmos e desenvolvimento java utilizando swing e JDBC.

1.4 Organização deste Texto

O Capitulo 1 faz um breve explanação sobre a industria aeronáutica. define aeródromo e aeroporto, faz uma explanação das características do aeroporto de Vitoria-ES, aborda seus recursos e facilidades oferecidas às aeronaves e aos clientes, além disso, define algumas características operacionais, aborda os objetivos principais e secundários e situa o leitor no contexto do problema que deseja-se resolver.

Já o Capitulo 2 aborda o sistema SISO/BDO com foco na características operacionais de interesse, situa o problema a ser resolvido dentro do campo de estudo Programação Linear e Otimização, analisa dados estatísticos do movimento de aeronaves, aborda alguns requisitos de qualidade e a forma proposta de Interação Humano Computador.

No Capitulo 3 abordam-se as tecnologias usadas na implementação da solução proposta, destacam-se ainda os pontos fortes de cada tecnologia e as razões de sua escolha.

O Capitulo 4 aborda o desenvolvimento da solução e por isso trata dos artefatos produzidos, a modelagem conceitual, paradigmas de desenvolvimento, alguns diagramas da UML, implementação da persistência, padrões arquiteturais ‘e de projeto, são apresentados alguns diagramas e um exemplo de prototipação.

Finalmente o Capitulo 5 foi reservado para as considerações finais, onde são apresentadas as conclusões e propostas de trabalhos futuros.

2 Histórico

O Sistema Integrado de Soluções Operacionais e Banco de Dados Operacionais (SISO/BDO) trata-se de um projeto levado a cabo pela Infraero com o objetivo de disponibilizar informações operacionais, as quais, serviriam de suporte à tomada de decisões gerenciais além de disponibilizar informações de interesse do publico que utiliza os Aeroportos da Rede Infraero (INFRAERO, 2004). Este sistema concentra em uma única base de dados um enorme gama de informações operacionais e armazena também informações de relevância histórica, pode se dizer que este é o sistema responsável por gerir o fluxo de voos do aeroporto o qual inclusive controla as informações encaminhadas ao Sistema informativo de voos (SIV).

Ao constatar que o sistema BDO implantado no aeroporto de Vitória é bastante dependente do operador, surgiu a ideia de propor uma melhoria em um dos processos componentes do sistema qual seja a alocação de recursos aeroportuários.

Esta alocação no BDO é baseada na decisão do operador do Centro de Operações Aeroportuárias (COA), o qual se vale da analise do contexto operacional e também da lista de voos previstos para o dia, o que aliado a visualização da disponibilidade de posições, permite realizar a alocação dos recursos. O processo é dinâmico e por isso mesmo sujeito a falhas. Durante a madrugada, o operador do COA lança no sistema os recursos previstos para uso pela aeronave considerando uma situação ótima, a qual raramente se confirma. Tal situação ocorre pois não é sempre que os voos chegam e saem no horário previsto no HOTRAN.

O operador do COA é um funcionário capacitado para controlar as comunicações de natureza operacional, planejar e executar a alocação de recursos aeroportuários, operar o SISO/BDO, realizar o levantamento de dados estatísticos, colaborar com a segurança patrimonial, prestar atendimento ao publico, fazer anúncios de utilidade publica além de possuir atuação de destaque no caso de Emergência Aeronáutica.

O HOTRAN é um documento “que formaliza as outorgas para a exploração de linhas aéreas regulares internacionais e domésticas de passageiros e/ou carga e da Rede Postal pelas empresas de transporte aéreo, com os respectivos horários, números de voos, frequências, tipos de aeronaves e oferta de assentos” (ANAC, 27 de abr. de 2021). As informações contidas no HOTRAN são fundamentais para a realização do planejamento das atividades operacionais e a respectiva alocação dos recursos necessários aos voos.

O problema da alocação de recursos aeroportuários, se assemelha ao problema da alocação de salas o qual também é um problema NP COMPLETO que vem sendo solucionado utilizando meta-heurísticas entre as quais Simulated Annealing (Dowsland 1998, Abramson 1991), Busca Tabu (Burke et al. 2001, Costa 1994, Hertz 1992) e Programação Genética (Ueda et al. 2001, Santos et al. 1997, Erben and Keppler 1996, Rich 1996) que vem sendo aplicadas em problemas de programação de horários (SOUZA, 2002).

2.1 Movimento Aeroportuário

Na Tabela 1 exibimos o quantitativo de pousos e decolagens conforme dados obtidos no Anuário Estatístico Operacional da Infraero envolvendo um periodo de 12 anos entre 2007

e 2019. Este documento reúne uma serie de dados englobando geralmente os 4 últimos anos e traz informações de todos os aeroportos da rede Infraero. Destacamos para compor a tabela abaixo apenas os dados referentes ao aeroporto de Vitoria (SBVT) o qual é objeto de nosso estudo.

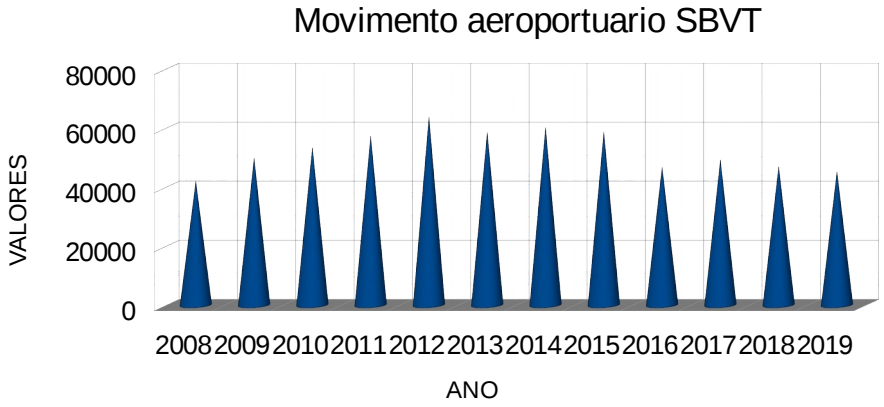
Os dados tabulados abaixo oferecem ao leitor uma noção do quão intenso é o movimento aeroportuário além disso permite inferir a utilidade de sistemas informatizados para apoiar a tomada de decisões operacionais e gerenciais. O Banco de Dados Operacionais é uma importante ferramenta para geração e obtenção de dados estatísticos assim como apoio a tomada de decisões gerenciais.

Tabela 1: Movimentação de Aeronaves

ANO	POUSOS + DECOLAGENS
2007	39778
2008	41936
2009	49807
2010	53360
2011	57293
2012	63777
2013	58504
2014	60144
2015	58760
2016	46737
2017	49201
2018	46950
2019	45204

Podemos apresentar os dados acima visualmente conforme o Gráfico 1. Neste gráfico, implementado com os valores são extraídos da Tabela 1 estão relacionados o somatório de pousos e decolagens a cada ano.

Gráfico 1 – Movimento aeroportuário



No gráfico 1 observamos um crescimento do Movimento Aeroportuário no periodo entre 2008 e 2012 o qual coincide com uma melhora na situação econômica do pais, posteriormente entre 2013 e 2015 observamos uma certa estabilidade seguida de queda em 2016 mantendo estável até 2019. Neste intervalo de 2016 a 2019 houve uma

alteração do MIX de aeronaves que operavam regularmente no aeroporto com a chegada de aeronaves de maiores envergadura e capacidade em substituição a outras de menor porte.

Nos momentos de crescimento econômico, conforme verificado entre 2008 e 2012, o fluxo de aeronaves pode apresentar um crescimento acelerado e, dentre os desafios enfrentados pela administração aeroportuária esta a necessidade de adequar o MIX de aeronaves aos recursos que o aeroporto tem a oferecer, lembrando sempre que os recursos são limitados e só é possível atender simultaneamente um determinado numero de aeronaves de uma certa categoria, sem perder de vista a necessidade de otimizar o desempenho operacional e consequentemente maximizar proveito econômico.

Uma possível solução é ampliar a infra estrutura, porém esta nem sempre é viável pois exige um elevado dispêndio financeiro por parte da administração publica, exige ainda percorrer as fases de um processo licitatório e enfrentar diversos interesses políticos e econômicos os quais tem potencial inclusive de inviabilizar as obras necessárias. Especificamente no Aeroporto de Vitoria, estas foram algumas dificuldades enfrentadas para a construção do novo aeroporto cuja atividade se arrastou por 16 anos, conforme amplamente noticiado na imprensa. Por isso, estratégias de gerenciamento dos recursos representam um importante ferramental quando se vislumbra ampliar os ganhos econômicos oriundos da exploração de um aeroporto e ao mesmo tempo oferecer um atendimento voltado a satisfazer as necessidades do Cliente.

Os clientes dos aeroportos são os passageiros que embarcam e desembarcam, as companhias aéreas que realizam voos regularmente utilizando o aeroporto, as empresas de táxi aéreo, as forças armadas, as empresas especializadas na importação e exportação de carga aérea, empresas de UTI no ar, companhias de petróleo a exemplo da Petrobras entre outros.

Observa-se que o transporte aéreo exerce um forte impacto no desenvolvimento econômico seja no turismo, seja no transporte de carga, [...] “essas pontes aéreas virtuais, permitem o fluxo econômico de bens, investimentos, pessoas e ideias” (IATA, 2019). Neste aspecto o trasporte aéreo é um importante indutor do desenvolvimento econômico pois agiliza o transporte de bens e de pessoas, desta forma facilita a realização de variados tipos de negócios dos quais podemos destacar o turismo, comercio e o transporte de produtos de alto valor agregado.

2.2 Contextualização do Problema

Durante o estudo de otimização combinatória e programação linear, tivemos o contato com os conceitos de heurística e meta-heurística que compõe o ferramental utilizado para solucionar problemas para os quais não se conhece um algoritmo de tempo polinomial. Estes são problemas definidos como NP COMPLETO.

Heurística é um procedimento mental simples que ajuda a encontrar respostas adequadas, embora várias vezes imperfeitas, para perguntas difíceis. A palavra tem a mesma raiz que eureka. (WIKIPÉDIA, 21/04/2021).

Meta heurísticas podem ser entendidas como métodos heurísticos utilizados para resolver problemas de otimização, para os quais não se conhece uma solução eficiente

(WIKIPÉDIA, 21/04/2021). Esses métodos não garantem uma solução ótima mas geralmente oferecem boa aproximação da solução a um custo aceitável.

O problema da alocação de recursos aeroportuários é muito parecido com o conhecido problema da alocação de salas de aula, um problema de otimização da classe NP COMPLETO ou seja um problema para o qual não se conhece um algoritmo que encontre uma solução ótima em tempo polinomial.

Considerando que na aviação os horários de voos são afetados pelas intempéries da natureza, pela necessidade de manutenção não programada, pela quantidade de passageiros embarcando e desembarcando, pelo tempo de carga e descarga, pela disponibilidade de pessoal e equipamentos, entre outros observa-se que os horários previstos nem sempre se concretizam, havendo variação de alguns minutos, algumas horas e inclusive cancelamento de operações, além disso, ocorrem operações não programadas a exemplo voos alternados. Entende-se que é possível obter uma boa alocação de recursos utilizando a estratégia de prioridade de uso de recursos, conforme é proposto neste trabalho. Diante disso, abrimos mão de utilizar meta-heurísticas conhecidas em busca de uma solução quase ótima, mesmo sabendo que os problemas NP COMPLETO geralmente são solucionados lançando mão desse ferramental.

2.3 Requisitos de Qualidade

O sistema em desenvolvimento, poderá ser utilizado por pessoas com vistas a obter informações e controle da movimentação de aeronaves assim como prover uma alocação eficiente dos recursos aeroportuários. Espera-se que os usuários do sistema não tenham que passar horas lendo manuais ou em uma sala de treinamento, para aprender utilizá-lo, ou seja, usabilidade é um requisito não funcional importante.

É esperado que o sistema seja fácil de usar e aprender, eficiente e eficaz, apresente uma boa aceitabilidade ou seja, o usuário precisa perceber que o sistema facilita seu trabalho pois desta forma desejara utilizá-lo, para isso, o sistema deve adotar conceitos e metáforas adequados ao meio aeroportuário o que equivale a apresentar um modelo conceitual simples e claro. Ao desenvolver um sistema, é preciso ter em mente que a tecnologia atrapalha as pessoas e o que elas querem fazer (BENYON, 2011). É fundamental reduzir este impacto quando se projeta software de qualidade, sendo assim, decidimos prover o sistema de uma Interface Gráfica com o Usuário.

2.4 Interface Homem Maquina

Ao estudar a Interação Humano Computador (IHC) encontramos alguns subsídios conceituais, os quais devem nortear o desenvolvimento de Interfaces Gráficas com Usuário (GUI), descobrimos que uma premissa básica da usabilidade é “deixar o controle na mão do usuário”, conforme preconiza a 3.ª Heurística de Nielsen (NIELSEN, Jakob. Ten usability heuristics. 2005).

Sabe-se que as atividades aeroportuárias apresentam a premissa implícita de satisfazer as necessidades dos clientes e neste contexto entende-se o usuário, no nosso caso as companhias aéreas, como sujeitos habilitados a realizarem as escolhas com o objetivo de melhor satisfazer suas próprias necessidades.

Baseado nestas constatações, surgiu a ideia de permitir às companhias aéreas a oportunidade de fazer escolhas a saber: dizer quais recursos gostaria de usar dentro de determinados contextos principalmente os relacionados à segurança das operações obedecendo ao que preconiza o RBAC 153, melhor aproveitamento dos recursos, redução de consumo de combustível e maior lucratividade.

Por isso, decidimos que a cada empresa aérea estará associada uma lista de preferencia por uso de determinado recurso e dai faremos a alocação automática dos recursos com base nesta lista preferencial, obedecendo a disponibilidade dos recursos.

2.5 Desenvolvimento do Trabalho

Sabe-se que o projeto e o desenvolvimento de um software envolve as seguintes atividades:

- Levantamento de requisitos;
- Analise de requisitos;
- Elaboração do diagrama casos de uso;
- Elaboração do diagrama de classes;
- Elaboração do protótipo das telas usada na *Interface gráfica com o usuário*
- Elaboração do diagrama EER usando o Mysql
- Implementação da Interface com o usuário.
- Implementação de uma ConnectionFactory para a conexão com o Banco de Dados
- Implementação do CRUD usando o padrão DAO e JDBC
- Implementação das Regras de Negócio
- Teste do Sistema

A realização deste trabalho consiste visitar as atividades acima listadas e realizar boa parte destas no curso do desenvolvimento do software a que nos propomos. Entendemos que realizar todas as tarefas supra citadas com maestria é tarefa para uma equipe de desenvolvimento composta por diversos especialistas , equipe esta da qual não se dispõe, ainda assim, ao realizar o percurso supra citado pretende-se produzir um software de boa qualidade e ampliar os conhecimentos do processo de software.

3. Base Teórica

A escolha das tecnologias empregadas para levar a cabo um projeto é uma etapa importante pois gera impacto em termos qualidade e produtividade. Razão pela qual realizamos abaixo uma breve explanação sobre tais tecnologias.

O presente trabalho será composto do projeto da Lógica do Negócio, projeto da IHC (Interação Humano Computador) e projeto da Persistência em Banco de Dados.

3.1 O Sistema Gerenciador do Banco de Dados

Durante o curso da disciplina Banco de Dados foram abordados os Sistemas Gerenciadores de Bancos de Dados Relacionais e citado o Mysql Workbench, entretanto não foram possíveis avaliações praticas do processo de implementação do banco de dados utilizando java JDBC. Conhecimento esse que foi complementado com pequisa na internet, onde sites como: Dev Media, youtube, Caelum e muitos outros foram fundamentais na consolidação dos conhecimentos requeridos para desenvolver adequadamente o projeto do Banco de Dados.

Ao escolher um dos Sistemas Gerenciadores de Bancos de Dados foi considerado importante ao SGBD: suportar a linguagem SQL, ser um software livre, usar diagramas EER, oferecer geração de código SQL a partir de um diagrama, suportar o modelo relacional , além de ser amplamente utilizado em grandes e múltiplas corporações.

Constatou-se ainda que o Mysql Workbench reúne todas as ferramentas necessárias ao desenvolvimento do Banco de Dados em um único ambiente de desenvolvimento integrado (WIKIPÉDIA, 07/09/2019), isto representa um ganho de desempenho na fase de desenvolvimento do projeto. O Mysql reúne todas essas características, o que aliado ao fato de já ter-se algum conhecimento sobre este SGBD foram fatores determinantes em sua escolha. Neste projeto foi utilizado o Mysql Workbench Community versão 6.3.

3.2 A Linguagem de Programação

A linguagem de programação escolhida foi Java SE 1.8. apesar de não ser um software livre, é amplamente utilizada pela comunidade e possui bastante documentação disponível na internet além de apresentar uma API (Application Programming Interface) moderna e bastante funcional. Pretende-se tirar proveito dos recursos disponíveis para manipular tipos de dados, inclusive utilizar a API de data e hora.

Java é uma linguagem de programação Orientada a Objetos, portável, amplamente utilizada no desenvolvimento de jogos Online, aplicativos para Android, páginas na Internet, documentos interativos, aplicativos desktop, ou seja, Java é bastante aplicada na industria de software.

3.3 O Ambiente de Desenvolvimento Integrado

Quando trabalha-se com programação, o uso de um IDE (Integreted Development Environment) agrega produtividade ao processo de desenvolvimento de software. Considerando que o Eclipse é uma das ferramentas mais utilizadas, foi escolhido o Eclipse Version: 2019-03 (4.11.0).

O Eclipse é uma IDE que apresenta uma boa curva de aprendizado, sendo considerada fácil, sua tecnologia baseada em plugins, oferece grande suporte e flexibilidade aos desenvolvedores de software razão pela qual é uma das mais utilizadas na indústria de software.

3.4 O Plugin Windowsbuilder Pro

Um plugin fundamental neste projeto é o Windowsbuilder pro, um plugin instalável no Eclipse e que visa facilitar o projeto de Interfaces Gráficas com o Usuário. Sua utilização agiliza o processo de criação das telas do aplicativo, responsáveis por facilitar a interação com o usuário.

Segundo o Eclipse.org informa:

> "O WindowBuilder é composto pelo SWT Designer e pelo Swing Designer e facilita a criação de aplicativos Java GUI sem gastar muito tempo escrevendo código. Use o designer visual WYSIWYG (*What You See Is What You Get*) e as ferramentas de layout para criar formulários simples para janelas complexas; o código Java será gerado para você. Adicione controles com facilidade usando o recurso de arrastar e soltar, adicione manipuladores de eventos aos seus controles, altere várias propriedades dos controles usando um editor de propriedades, internacionalize seu aplicativo e muito mais.
>
> O WindowBuilder é construído como um plug-in para o Eclipse e os vários IDEs baseados em Eclipse (RAD, RSA, MyEclipse, JBuilder, etc.). O plug-in cria uma árvore de sintaxe abstrata (AST) para navegar no código-fonte e usa o GEF (Graphical Editing Framework) para exibir e gerenciar a apresentação visual.
>
> O código gerado não requer nenhuma biblioteca personalizada adicional para compilar e executar: todo o código gerado pode ser usado sem o WindowBuilder Pro instalado. O WindowBuilder Pro pode ler e escrever quase qualquer formato e fazer engenharia reversa da maioria dos códigos Java GUI (Graphical User Interface) escritos à mão. Ele também suporta edição de código de forma livre (faça alterações em qualquer lugar ... não apenas em áreas especiais) e a maioria dos re-fatores (você pode mover, renomear e subdividir métodos sem problemas)" (ECLIPSE, 2021).

Percebe-se , conforme descrito acima e no site Eclipse, que o uso do Windowsbuilder facilita o processo de desenvolvimento das telas e agrega produtividade devido a geração automática de código Java para criação da GUI.

Apesar de não ser o objetivo deste trabalho criar uma interface com o usuário de excelência, observamos a importância de recordar o uso intenso de metáforas na implementação da interação homem x maquina (BARANAUSKAS,2003) e dai a necessidade uso das metáforas adequadas a promover o interesse do usuário em utilizar o software em desenvolvimento.

Neste mesmo diapasão, recorremos aos ensinamentos de (NIELSEN, JAKOB, 1985) pois através do contato com as 10 heurísticas de Nielsen encontra-se um norte para o desenvolvimento de interfaces.

Nilsen trata da "visibilidade do status do sistema, correspondência entre o sistema e o mundo real, liberdade e controle do usuário, consistência e padrões, prevenção de erros, reconhecer ao invés de lembrar, flexibilidade e eficiência, estética e design minimalista,

*diagnostico e recuperação de erros, ajuda e documentação[...]". Um maior detalhamento pode ser encontrado no artigo "*Heurísticas de Nielsen: 10 Dicas para melhorar a Usabilidade de sua Interface" disponível em *(AELA.IO, 2019).*

É evidente que neste trabalho não foram aplicadas integralmente os conceitos acima mas reconhecemos que os mesmos representam uma importante direção a seguir quando o assunto é usabilidade.

O projeto da interface com o usuário, deve seguir paralelamente durante todo o processo de desenvolvimento do software e visa que o usuário não precise de um manual de instruções para utilizar o sistema conforme preconiza (MACEDO, 2017). Aqui o principal requisito ao qual pretende-se atender é a usabilidade.

3.5 Diagramas UML

*Uma parte fundamental do curso de Bacharel em Ciências da Computação é o estudo da disciplina Engenharia de Software. O professor Ricardo Falbo foi o titular desta disciplina, onde é dado ao aluno a oportunidade de um contato introdutório à modelagem de software através de diagramas UML (*Unified Modeling Language)*.*

Aqui os diagramas foram um norte para o processo de desenvolvimento de software pois julgamos importante aplicar os conhecimentos de Engenharia de Software quando um dos requisitos é a qualidade do Software.

*Conforme (*LUCIDCHART, *2019) os diagramas UML são classificados em:*

- *Diagrama de Classes*
- *Diagrama de componentes*
- *Diagrama de Estruturas compostas*
- *Diagrama de Objetos*
- *Diagrama de Pacotes*
- *Diagrama de Atividade*
- *Diagrama de Comunicação*
- *Diagrama de Visão Geral da Interação*
- *Diagrama de Sequencia*
- *Diagrama de tempo*
- *Diagrama de Casos de Uso*

O diagrama de casos de uso é uma representação dos atores e quais casos de uso cada ator realiza. Ele permite visualizar como se dá a interação entre esses elementos conceituais dotando os steakholders de uma compreensão do que o sistema faz e quem são os responsáveis por cada etapa do processo de solução do problema, o qual o software pretende resolver. Estes diagramas"são importantes para organização e modelagem dos comportamentos do sistema" (Booch, et al, 2000).

O diagrama de classes é uma representação conceitual dos elementos contrutivos que serão utilizados na implementação do software juntamente com a descrição visual de como estes elementos se relacionam para gerar os objetos tratados no software. Através desse diagrama temos uma visão macroscópica das classes e de como elas se relacionam para atingir os objetivos do software. Esse diagrama é utilizado para fornecer uma visão estática do sistema conforme recomenda (Booch, et al, 2000).

Apesar da UML permitir da criação dos diversos diagramas citados acima, escolhemos os diagrama de Casos de Uso, o diagrama de Classe e o diagrama de Pacotes, os quais se revelaram necessários e suficientes para o desenvolvimento do software proposto.

Para geração dos diagramas UML, foi usada a ferramenta Astah Comunity 7.1.0 /f2c212 Model Version 37, a qual foi recentemente descontinuada, porém isto não é um grande problema haja vista existência de outras ferramentas em software livre para modelagem UML por exemplo ArgoUML e Dia. Existem também plugins Eclipse dentre os quais pode ser citado o Papyrus e diversas ferramentas online.

3.6 Modelagem do Banco de Dados

*O modelo de entidades e relacionamentos (ER) é uma abstração do esquema do banco de dados ilustrada graficamente através de um diagrama o qual representa os conceitos de entidades, atributos e como estes elementos estão relacionados (*MALNOR, Cody et al. 2018*).*

*O modelo de entidades e relacionamentos estendido, adotado pelo MYSQL, foi proposto como uma extensão do modelo ER (TEORY, et al, 1986) acrescentando conceitos do paradigma de orientação a objetos a saber: relacionamentos entre classe e subclasse, herança de tipo, hierarquias de generalização e especialização assim como as respectivas restrições (*MALNOR, Cody et al. 2018*).*

Foi necessário construir um Diagrama de Casos de Uso e um Diagrama de Classes. O Diagrama de Classes foi uma base para a criação do diagrama EER (Diagrama de Entidade e Relacionamento Estendido) no Mysql Workbench e geração do script para a criação do Banco de Dados Relacional que implementamos como parte deste Trabalho de Conclusão de Curso.

4 Desenvolvimento da Ferramenta

O processo de desenvolver um software traz embutido diversas tomadas de decisões, deve-se escolher entre outras coisas padrões de projeto e padrões arquiteturais, geralmente é necessário a criação de modelos com propósitos variados.

4.1 Requisitos e Modelagem Conceitual

Uma atividade muito importante do processo de desenvolvimento de software é o levantamento e documentação dos requisitos do software.

Requisitos de software são definidos pela norma IEEE-90 como sendo:

- *Uma capacidade que um usuário necessita para resolver um problema ou atingir um objetivo;*
- *Uma capacidade que deve ser atendida ou possuída por um sistema ou componente de um sistema para satisfazer um contrato, padrão, especificação ou outro documento formalmente imposto;*
- *O conjunto de todos os requisitos que formam a base para o desenvolvimento subsequente de um software ou componentes de um software;*

No caso do Sistema de Alocação de Recursos Aeroportuários os principais requisitos são:

- O sistema deverá permitir o cadastro de aeronaves, o cadastro de posições de estacionamento, a alocação da aeronave a posição mais conveniente.
- O sistema deve conter uma lista de prioridades de posições para cada companhia aérea.
- A aeronave devera ser alocada na posição mais prioritária que esteja livre no horário confirmado de pouso desde que a posição comporte tal aeronave.
- O sistema deve alocar aeronaves asa fixa dos grupos I.

O critério de alocação de aeronaves do grupo I será conforme já definido acima.

As aeronaves do grupo II são divididas em duas categorias a
saber:
i) Aeronaves de asa fixa
ii) Aeronaves de asa móvel (Helicóptero)
Podendo ser aeronaves civis ou militares

As aeronaves do Grupo I são classificadas ainda por tipo de voo:
i) Regular
ii) Não Regular
iii) Charter
iv) Fretamento
v) Translado

- As aeronaves do grupo II precisam solicitar previamente um dia e horário para pouso sendo que neste caso seu pouso será considerado com coordenação e terá sua permanecia garantida, conforme acordado com a administração aeroportuária. Deve ser informado a matrícula da aeronave, o equipamento, o nome do comandante a hora de pouso e a hora de decolagem. Será definida uma posição para a alocação da aeronave que esteja vazia nesta data e hora.

- O sistema deve prover um banco de dados onde armazenaremos os dados das aeronaves alocadas a cada posição e as posições vazias para fins de efetivar a politica de alocação, além de todos os dados persistentes de interesse para operacionalização do sistema.

- O sistema não permitirá a alocação de um recurso durante o tempo em que estiver alocado a algum voo.

Observamos que temos na verdade ao menos duas politicas de alocação diferentes conforme a aeronave seja do grupo I ou II. Aqui trataremos da alocação de aeronaves que realizam voos do grupo I pois neste grupo estão os voos comerciais de grande porte os quais impactam diretamente o turismo e a economia capixaba.

Em outro momento, poderíamos definir critérios para alocação de aeronaves asa móvel e também aeronaves militares, e outras aeronaves do grupo II, por exemplo prioridade ou adotar heurística ou meta-heurística que forneça uma boa solução para o problema de alocação de recursos.

De uma aeronave desejamos saber:

- A matricula
- A qual grupo ela pertence
- Asa fixa ou asa móvel
- Envergadura(comprimento e largura)
- PMD (Peso máximo de decolagem)

Para as aeronaves do grupo I, após alocar a posição de estacionamento, deverão ser alocados também as esteiras de bagagem, e o portão que será utilizado para embarque, logo:

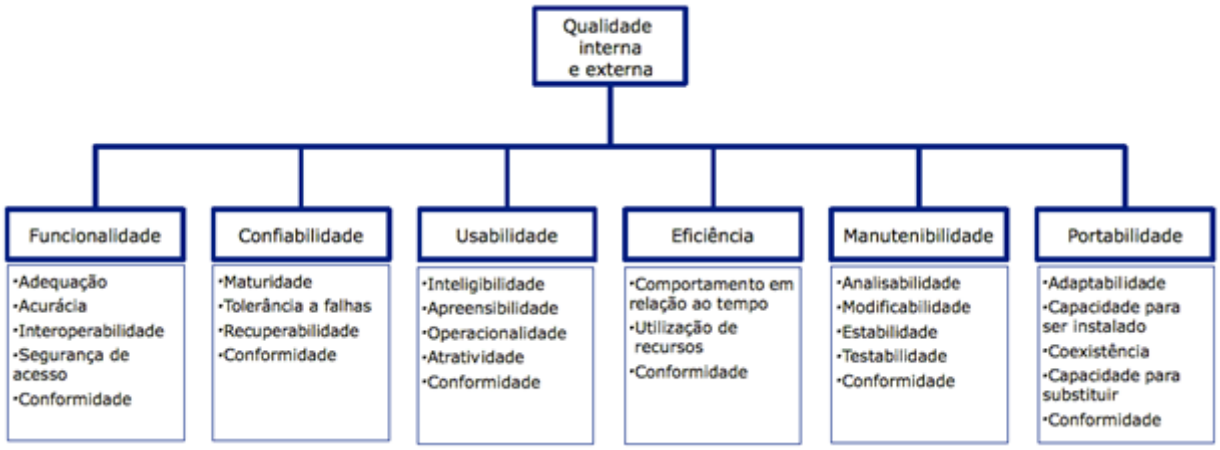
- O sistema deverá permitir o cadastramento de portão de embarque e esteiras de bagagem
- O sistema deverá permitir alterações de alocação realizadas manualmente e atualizar o banco de dados convenientemente.
- O sistema deverá permitir a execução simultânea em várias estações de trabalho.
- O sistema deve apresentar boa manutenibilidade e usabilidade.
- O sistema deve permitir a geração de relatórios gerenciais.
- O sistema deve ser independente de plataforma.
- O sistema deve exibir uma relação (tabela) dos voos previstos para um determinado dia conforme Hotran, assim como permitir que os dados sejam atualizados dinamicamente.

Os requisitos acima, são os funcionais, além desses, o sistema deve apresentar ainda alguns dos requisitos não funcionais, preconizados pela norma ISO-9126 a saber:

- **Usabilidade:** entendida como facilidade de aprender e usar o sistema;
- **Funcionalidade:** o sistema deverá funcionar em diferentes arquiteturas;
- **Confiabilidade:** conjunto de atributos que evidenciam a capacidade do software de manter seu nível de desempenho sob condições estabelecidas durante o periodo de tempo estabelecido.
- **Manutenibilidade:** atributos de software que evidenciam o esforço necessário para modificá-lo, remover defeitos ou adaptá-lo a mudanças ambientais.
- **Portabilidade:** conjunto de atributos que evidenciam a capacidade do software em ser transferido de um ambiente para outro.

A figura 1 detalha os requisitos acima (RUGGERO. RUGGIERI, 2016).

Figura 1 – Requisitos de Qualidade

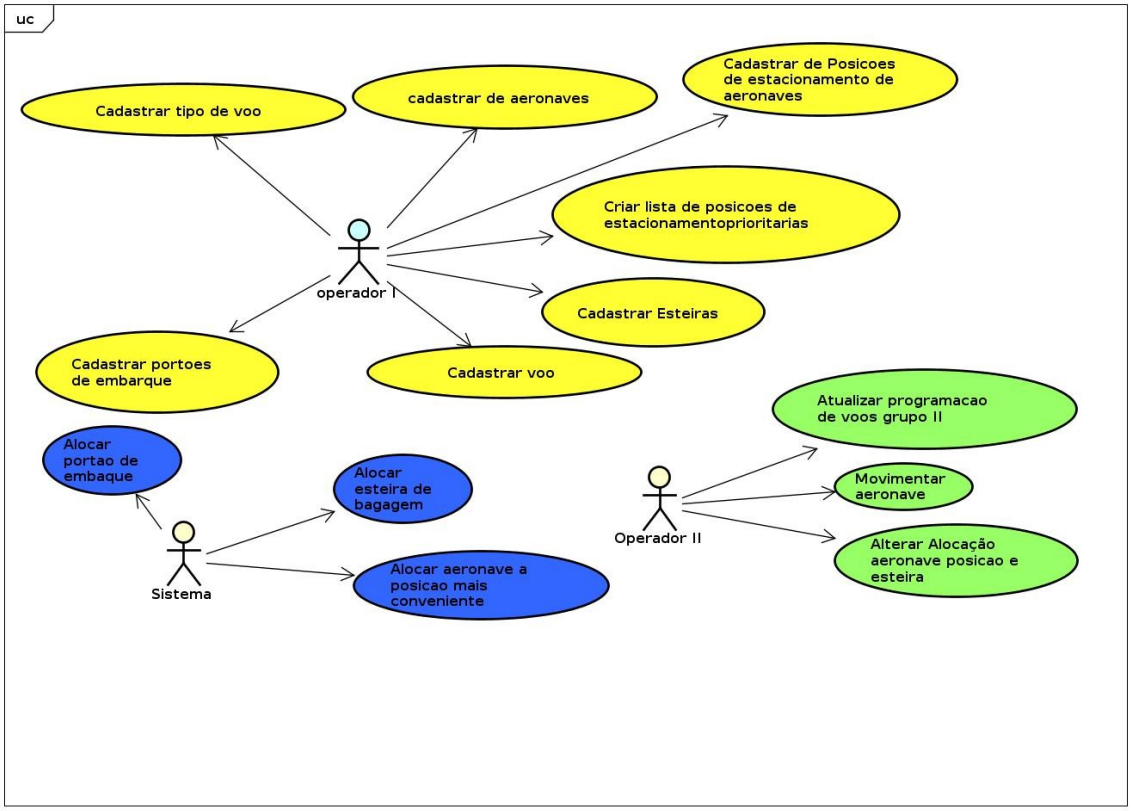


Neste trabalho, adotamos o conceito de Qualidade de Software definido como “conformidade a requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo software profissionalmente desenvolvido”(PRESSMAN, 2011).

Um dos principais objetivos no desenvolvimento desta ferramenta consiste em apresentar uma proposta de melhoria em um software já existente, o chamado SISO/BDO. Trataremos aqui da alocação automatizada de recursos aeroportuários.
Recursos aeroportuários no contexto deste projeto, são pistas de pouso e decolagem, posições de estacionamento, pontes de embarque e desembarque, heliponto, cabeceiras de pouso/decolagem, portão de embarque, portão de desembarque, esteira de devolução de bagagens.

Uma etapa muito importante do processo de desenvolvimento de software é a analise e especificação de requisitos a partir da qual elaboramos o diagrama de casos de Uso da UML, conforme (Figura 2). Para tanto lançamos mão do Software Astah Community para gerar o artefato supracitado.

Figura 2 – Diagrama de Casos de Uso



Segundo (Booch, et all 2000) o Diagrama de Casos de Uso registra os aspectos estáticos do sistema, dessa forma, explicita as funcionalidades que o mesmo deve oferecer para satisfazer as necessidades dos stakeholders.

O diagrama de casos de uso acima mostra que o programa é operacionalizado por 3 atores a saber:

Sistema – responsável por alocar automaticamente a aeronave à posição mais conveniente, conforme definido pelo proprietário, além disto é ele quem aloca o portão de embarque e a esteira de bagagens para a aeronave.

Temos ainda dois operadores, os quais são designados no diagrama como sendo Operador I e Operador II. Esses operadores são pessoas responsáveis pela implementação de casos de uso específicos.

Operador I – é quem realizará as operações de cadastro:

- cadastro de aeronaves;
- cadastro de categoria;
- cadastro de esteira;
- cadastro de frequência;
- cadastro de hotran;
- cadastro de pista;
- cadastro de portões de embarque;
- cadastro de posições no heliponto;
- cadastro de posição no patio;
- cadastro de preferencia no uso dos recursos;

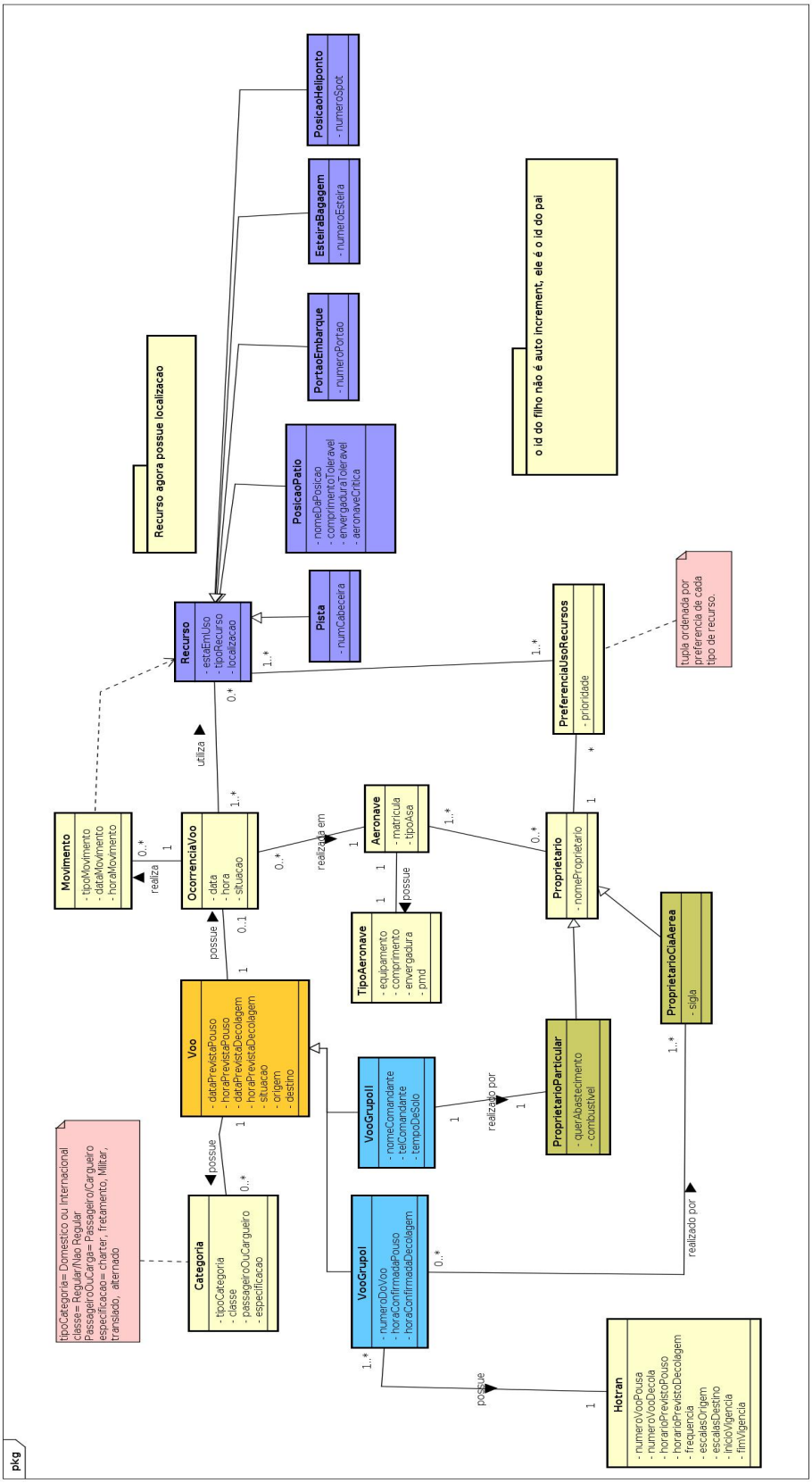
- cadastro de proprietário companhia aérea;
- cadastro de proprietário particular;
- cadastro de recurso;
- cadastro de recurso em ocorrência de voo;
- cadastro de recursos por proprietário;
- cadastro de tipo de aeronave;
- cadastro de voo;
- cadastro de voo grupo I;
- cadastro de voo do grupo II;

Operador II – é quem realizará as operações de atualização do banco de dados através dos seguintes casos de uso:

- atualizar a programação de voos do grupo II;
- registrar a movimentação de aeronave através do cadastro de movimentos;
- alterar a alocação da aeronave se necessário.
- cadastro de ocorrência de voo;

O diagrama de classes (Figura 3) é outro diagrama fundamental para este projeto, ele demonstra as principais classes que devem ser implementadas e serve de base para a tomada de decisões referentes a camada de persistência. No diagrama abaixo além de exibir as classes também esta demonstrado a maneira como estas se relacionam assim como a cardinalidade dos diversos relacionamentos. Observa-se ainda as relações de herança, muito comuns no paradigma Orientado a Objetos.

Figura 3 – Diagrama de Classes

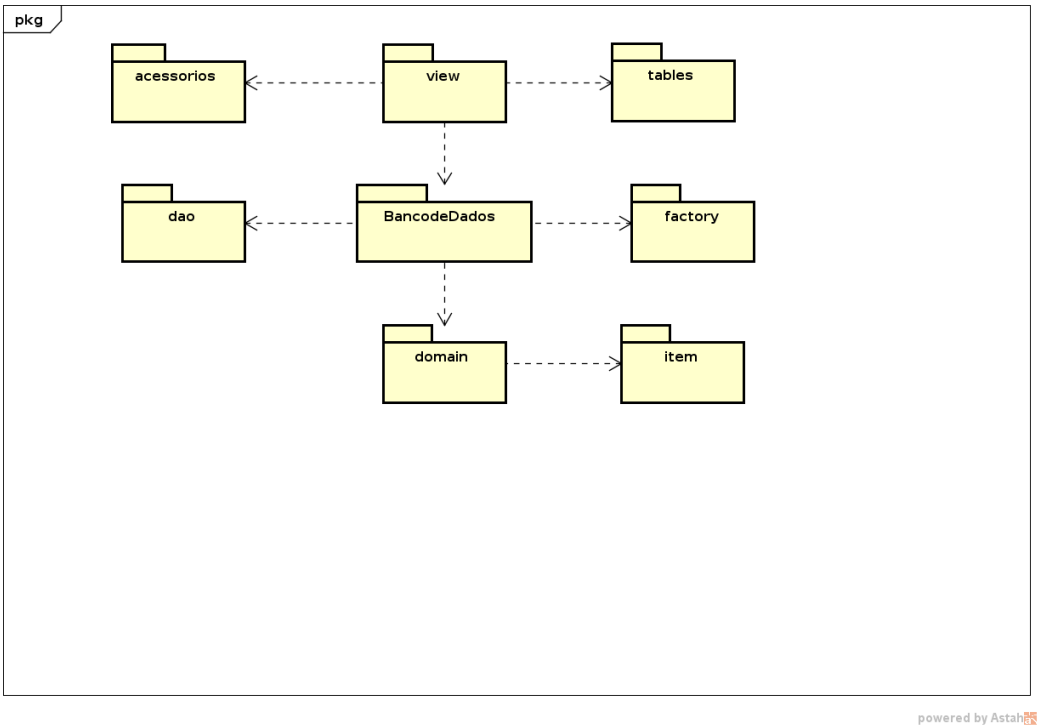


4.2– Projeto e Implementação

Para que o sistema pudesse ser implementado, diversas decisões foram tomadas em relação ao arcabouço cientifico e tecnológico disponível para a obtenção do produto de software.

O diagrama de pacotes permite uma macro visão do sistema conforme pode ser constatado na (Figura 4).

Figura 4 – Diagrama de pacotes



O sistema foi organizado conforme o diagrama de pacotes acima, o objetivo é agrupar as classes conforme sua responsabilidade dentro do sistema e assim facilitar a manutenibilidade, a usabilidade e o reuso de código. Em cada pacote as classes que possuem funções semelhantes foram devidamente agrupadas.

No pacote Domain reunimos as classes de domínio (diretamente relacionadas ao problema que queremos resolver).

No pacote item temos uma classe que trata de um subitem especifico de uma das classes do domain, sua necessidade foi constatada em tempo de desenvolvimento.

Para implementar a persistência usamos um SGBD, o pacote DAO reúne as classes que implementam a manipulação do CRUD – Criação, Consulta, Atualização e Destruição respectivamente Create, Read, Update e Delete.

No pacote factory encontra-se a classe responsável pela conexão com o banco de dados baseado no Design pattern Factory.

Considerando que Java foi a linguagem de programação escolhida para implementar o sistema, para implementar a persistência utilizamos a API JDBC pois julgamos importante estudar um pouco esta API nativa do java para num futuro ter um padrão de comparação com os frameworks de persistência tal qual “JPA, Groovy SQL, Hibernate, Spring JDBC, myBatis, Torque, Castor, Java Object Oriented Querying” (FRANZINI, 2017).

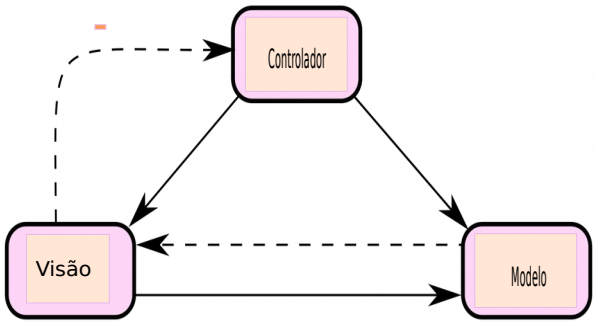
No pacote view temos as classes que gerenciam tudo que é exibido na tela para o usuário. Esta classe utiliza serviços fornecidos pelas classes dos pacotes acessórios e tables. A classe acessórios funciona com um deposito temporário de dados já no pacote tables estão as classes que implementam as tabelas utilizadas pelo pacote view.

Com a organização explanada acima, buscamos implementar no sistema desktop o padrão arquitetural MVC o qual foi criado em 1979 por Trygve Reenskaug (Xerox PARC).

4.2.1 O padrão MVC

É importante observar que este padrão foi a base teórica para o agrupamento das classes utilizado no projeto. Dentre as vantagens do padrão arquitetural MVC esta o “reuso de código, melhora da manutenibilidade, permite alterar a GUI sem necessidade de alterações na logica de negócios, permite designers e desenvolvedores trabalhem em paralelo” (WIKIPÉDIA, 2021). Entendemos que desta forma agregamos produtividade e manutenibilidade ao processo de desenvolvimento de software.

Figura 5 – O padrão MVC



Na (Figura 5) temos um diagrama clássico do padrão arquitetural MVC. No modelo são agrupadas as classes que descrevem a Lógica do Negócio assim como o acesso aos dados. Na visão são agrupadas as classes que tratam da Interface com o usuário e o Controlador é o responsável por gerenciar o fluxo de comunicação entre as classes.

Ao cursar a disciplina projeto de sistemas de software, tivemos um primeiro contato com o padrão MVC (Model-View-Controler) através da apostila do professor Ricardo Falbo, e neste projeto , constatou-se a oportunidade de adotar este padrão arquitetônico.

O padrão MVC é dividido em três partes com responsabilidades especificas a saber:

- Model: é composto por objetos diretamente ligados à aplicação, conforme preconiza (GAMMA et al., 1995). É onde estão inseridas as classes diretamente ligadas as regras de negócio do projeto.

- View: é onde estão implementados os designs da GUI (Grafic User Interface) ou seja são as classes que implementam as telas/janelas de interação dos usuários com o programa.

- Controller: é onde definimos a resposta do sistema às interações do usuário com o mesmo.

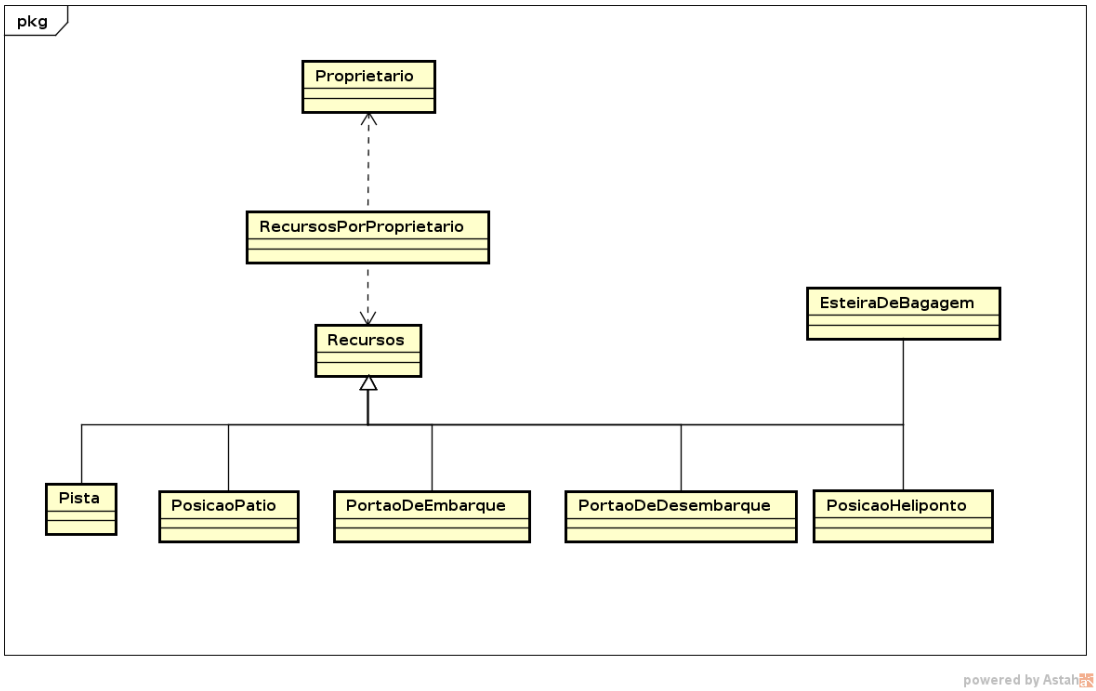
Esta separação de responsabilidades visa tornar o sistema mais flexível e facilitar o reuso de código.

É importante perceber que essas partes não estão isoladas no sentido próprio da palavras pois é na sua interação que surge o sistema propriamente dito. Na verdade, existe uma comunicação entre as diversas partes do sistema sempre que ocorre uma mudança de estado do sistema(GAMMA et al., 2009).

4.2.2 Diagrama de Classes do CDP

No diagrama da (Figura 6), o qual é uma parte do diagrama de classes do componente de domínio do problema(CDP) destacamos uma parte fundamental para a solução que pretendemos apresentar. Neste diagrama observamos que a existe uma relação de dependência entre a classe recursos por proprietário e as classes proprietário e recursos. Esta classe é de extrema importância pois uma vez definidos os recursos por proprietário a alocação de recursos fica imensamente facilitada pois dar-se-á conforme a preferencia do proprietário mediante disponibilidade do recurso. O sistema não devera permitir a alocação de um recurso já alocado, exceto a pista cujo critério de alocação é estar em uso.

Figura 6 - Recursos por proprietário - Diagrama de classes do(CDP)

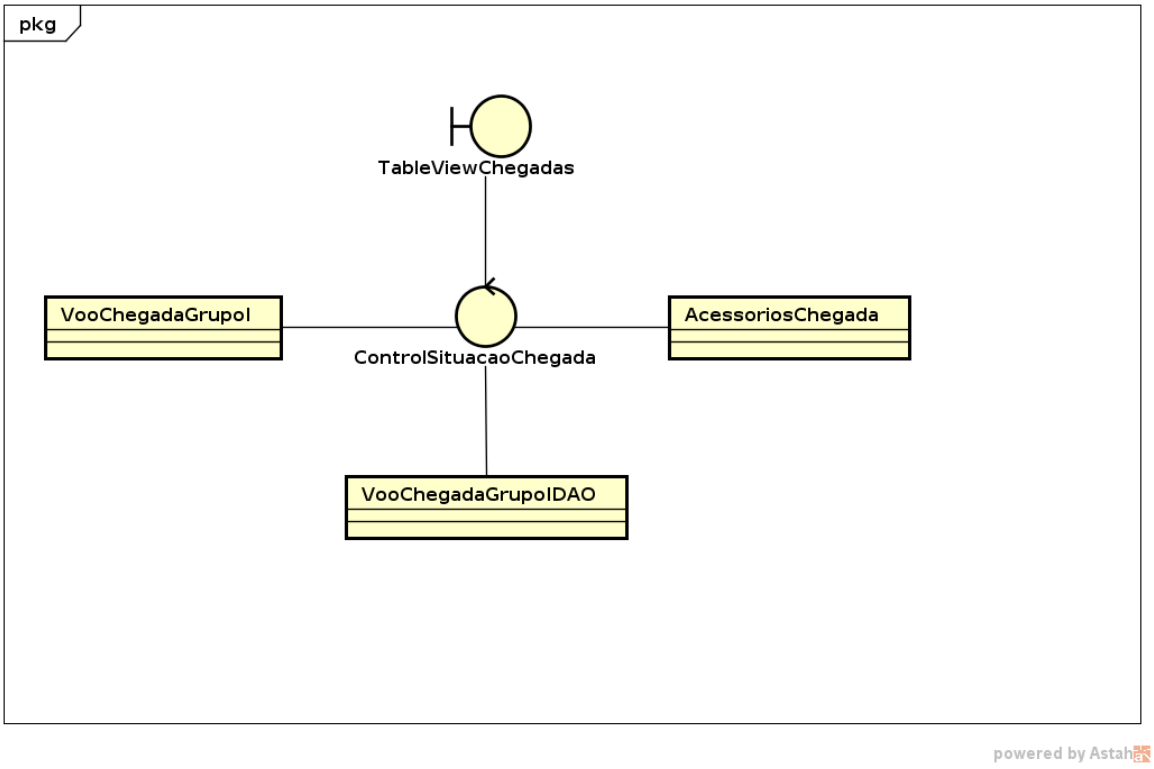


Os recursos que serão alocados pelo sistema são: cabeceira, posição no pátio, esteira de bagagem, portão de embarque. Salientamos ainda que a cabeceira de pouso em uso é definida pela torre de controle e a definição é feita considerando a direção do vento no

momento do pouso. Já a numeração das cabeceiras indicam a inclinação da pista em relação ao norte magnético da terra e podem mudar com o tempo, dependendo da mudança desta inclinação (FARIAS, 2020). No aeroporto de Vitoria-ES a pista antiga, hoje chamada pista 02, teve a numeração das cabeceiras alteradas devido a essa mudança de inclinação. Com isso a cabeceira 05 passou a ser designada cabeceira 06 e a cabeceira 23 passou a ser chamada cabeceira 24.

O aeroporto de Vitoria-ES possui torre de controle, no caso é ela quem diz qual a cabeceira em uso para pouso e decolagem. Lembrando que cada pista possui duas cabeceiras. De posse da informação da cabeceira em uso a cada instante, o sistema aloca convenientemente a pista e os demais recursos conforme os critérios estabelecidos.

Figura 7 – Controller Voo Chegadas



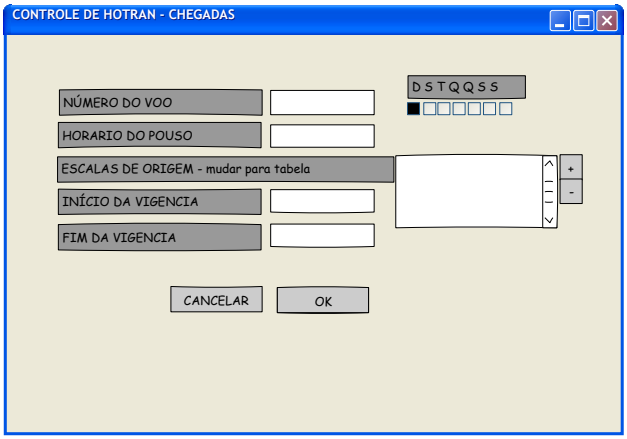
A (Figura 7) é um esboço das interações que ocorrem as quais permitem controlar os voos chegando no aeroporto. O controle é realizado atraves da tabela de chegadas e esta depende das classes AcessoriosChegada, depende dos voo de chegada e depende ainda de interagir com o Banco de dados. Da mesma forma, o controle de voos Partida lança mão da mesma ideia e a interface com o usuário é bem parecida, razão pela qual abrimos mão de explicitar em um diagrama conforme acima.

Vale destacar que esta esboçado na (Figura 7) um dos componentes essenciais da Interação com o usuário o qual através de um menu drop-down permite a navegação por diversas partes do sistema e a correspondente atualização das informações contidas não só na tabela mas também no próprio artefato usado na persistência. Estas atualizações ocorrem em resposta a eventos gerados pelos componentes responsáveis pela *Interação Humano Computador (IHC).*

4.3 Protótipo

Antes de realizar a implementação da interface gráfica com o usuário, utilizamos a ferramenta Pencil para projetar esboço de algumas partes GUI e desta forma facilitar a comunicação das ideias que seriam utilizadas na implementação da Interação Humano Computador. Vale destacar o aprendizado aqui obtido uma vez que o uso de prototipação permite aos stakeholders terem uma ideia visual do software em desenvolvimento. Ainda que não represente a visão final do projeto os protótipos são importantes pois facilitam o processo de comunicação. Um exemplo de protótipo é apresentado na (Figura 8).

Figura 8 - Protótipo



4.4 Base de Dados

É no Banco de Dados que estão as informações necessárias para gerar inicialmente a Tabela de voos que pousam ou decolam do aeroporto em uma determinada data. Utilizamos o mecanismo de query do MYSQL para popular a tabela. É atraves desta tabela que ocorre a interação do usuário com o sistema nas operações inerentes á chegada de um voo. As situações de cada voo no tempo são manipuladas e atualizadas pelo usuário através de um menu drop-down e telas disponibilizadas pelo sistema.

É através da interação que o usuário participa da atualização das informações do sistema conforme muda a situação dos voos.

Os clientes do aeroporto, visualizam nas telas do SIV(Sistema de Informações de Voo) disponíveis no saguão do aeroporto das informações relevantes ao voo de seu interesse. Tais informações dependem da situação do voo que pode estar previsto para pousar em um determinado horário, confirmado para pouso, atrasado, alternado, cancelado e outras. Estas situações são influenciadas por vários fatores dos quais podemos destacar: necessidade de manutenção não programada, condições meteorológicas, atrasos em etapas anteriores, tempo necessários para embarque e desembarque de carga e passageiros, fluxo de pessoas no aeroporto, tempo necessário ao abastecimento da aeronave só para citar alguns.

Os acessórios chegadas e também os acessórios partida, são objetos de vida curta, seus atributos são armazenados provisoriamente atraves de tabelas do Banco de Dados e conforme mudam as situações do voo eles se transformam em voos e serão armazenados na tabela de Voos Chegada ou Voo Partida conforme o caso. Para que isso ocorra é necessário conhecer a matricula da aeronave que ira pousar ou decolar pois o sistema precisa desta informação para buscar no Banco de Dados outras informações necessárias para atualizar a situação dos voos na tabela respectiva e assim persistir corretamente esses dados e atualizar a alocação de recursos conforme proposto neste trabalho. Conforme indicado, ao persistir um Voo Chegada o objeto acessoriosChegada ou acessoriosPartida equivalente é excluído do Banco de Dados.

4.5 Recursos e Facilidades

Ao pousar, um voo deverá utilizar uma cabeceira especifica de uma pista de pouso e decolagem também especifica, posteriormente deve ser conduzido através da pista de táxi até a posição de estacionamento. Na posição de estacionamento ocorre o desembarque de passageiros, carga, bagagem assim como o abastecimento, embarque de carga, passageiros e bagagens além de pequenas manutenções.

Ao desembarcar os passageiros são conduzidos à sala de desembarque, no patio 01 geralmente através de pontes de embarque e desembarque mas havendo necessidade podem ser conduzidos de ônibus ou a pé conforme haja ou não disponibilidade de veiculo e motorista para o transporte.

Na sala de desembarque os passageiros se dirigem à esteira de bagagens para recolher suas bagagens e ao sair da sala desembarque completa-se o ciclo de pouso.

No caso dos voos que irão decolar, o passageiro chega ao aeroporto, realiza o despacho de bagagens e caso não tenha feito o check-in eletrônico o faz nos balcões disponibilizados pelas companhias aéreas. Posteriormente dirigem-se à sala de embarque onde ocorre a inspeção de passageiros sendo posteriormente encaminhados ao portão de embarque que será utilizado pelo seu voo e através deste portão se dirige a respectiva aeronave quando a cia aérea liberar o embarque dos passageiros.

Para decolar o comandante da aeronave solicita autorização à torre de controle (TWR), a qual é responsável pelas operações em solo, e sendo autorizada deixa a posição em que esta estacionada e numa operação conhecida como push back e conduzida até a faixa de rolagem. Na faixa de rolagem o comandante solicita autorização para o inicio do taxiamento até a cabeceira em uso para decolagem. Chegando a cabeceira de decolagem, o comandante realiza os testes de praxe, e estando tudo conforme, solicita à TWR autorização para decolar. Se for autorizado o comandante iniciará a corrida e decolará rumo ao aeroporto de destino.

A atuação da TWR visa garantir a segurança o que é uma premissa básica das operações aeroportuárias. É sabido que cada item transportado possui via de regra alto valor agregado o que eleva muito a responsabilidade de todos os envolvidos nas atividades operacionais haja vista que as pessoas mais influentes da sociedade utilizam com muita frequência este modal de transporte. Além disto, um acidente ou incidente aeronáutico geralmente acarretam enormes prejuízos financeiros, atrasos de outros voos e fechamento do aeroporto. Dessa forma acidentes e incidentes aeroportuários geram um

grande impacto local e podem afetar uma parte considerável da malha aérea gerando atrasos, cancelamentos de voos, voos alternados, necessidade de acomodação dos passageiros em hotel ou em outros voos muitas vezes importantes negócios não são fechados além do impacto psicológico gerado nos familiares das vítimas.

Percebe-se diante do exposto acima que segurança é um atributo fundamental nas operações aéreas e por esta razão a ANAC possui uma legislação específica onde trata deste tema, desta forma o modal aéreo detém o titulo de meio de transporte mais seguro.

Outra grande preocupação da administração aeroportuária é com o fluxo de informações de interesse dos clientes razão pela qual disponibiliza diversos terminais de video, estrategicamente posicionados para fornecer ao passageiro as informações relevantes de seu voo entre as quais: situação do voo, horário de pouso e decolagem, portão de embarque e outros. Além disso o sistema SISO/BDO permite a disseminação de informações sonoras de interesse publico geralmente relacionadas as operações aeroportuárias e à segurança pessoal e patrimonial. Questões de saúde e outras consideradas relevantes pela administração.

Para manter essas informações, os dados são persistidos em um Banco de Dados.

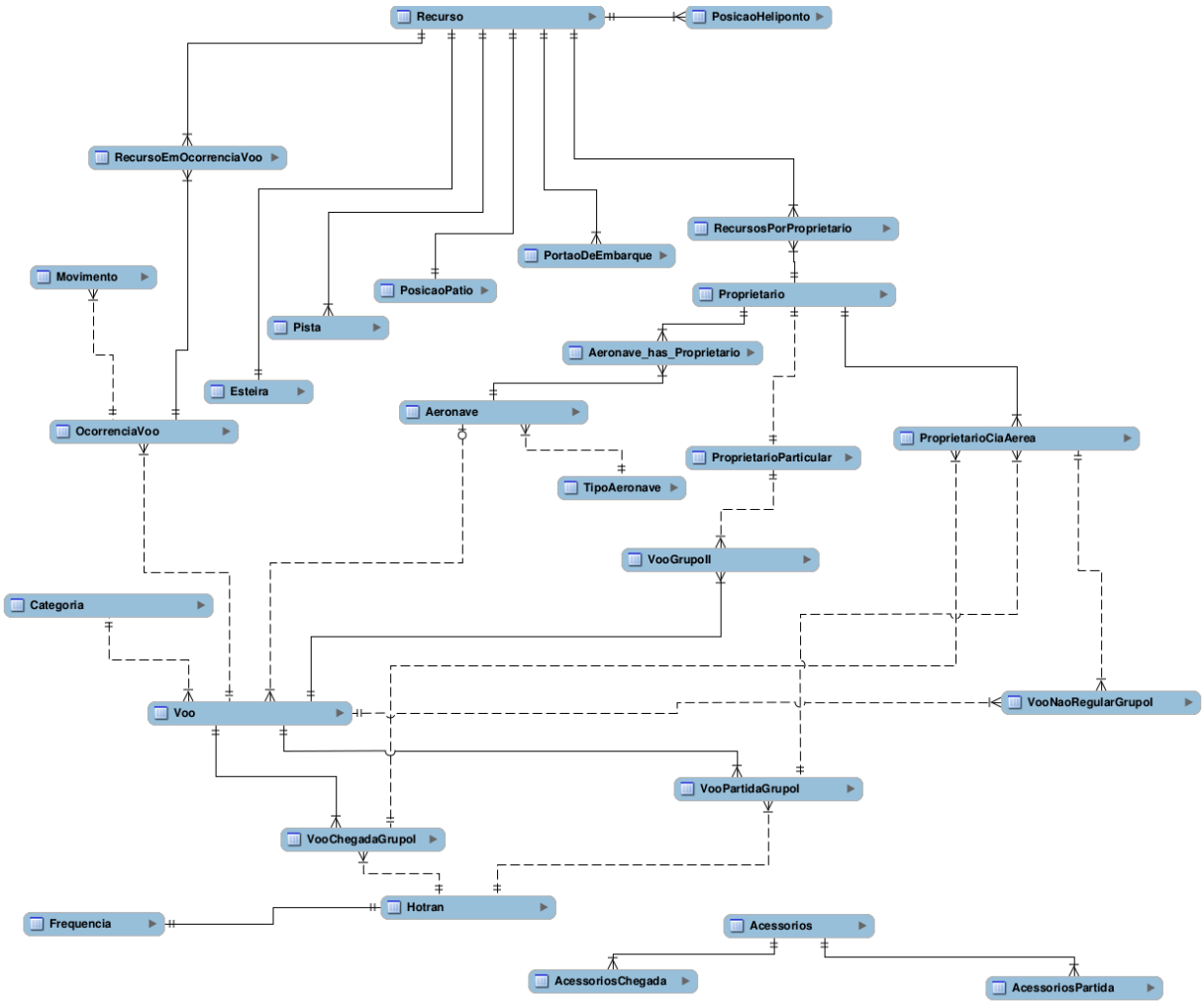
4.6 Modelagem de Dados

Na (Figura 9) visualizamos o Diagrama de Entidades e Relacionamentos Estendido. Este diagrama foi implementado usando o Mysql Workbench. Esta é uma importante ferramenta oferecida pelo SGBD para a modelagem de dados, pois uma vez implementado o diagrama é possível gerar automaticamente o código que da origem as tabelas do Banco de Dados.

Ao implementar a persistência, não lançamos mão de nenhum Framework de Mapeamento Objeto Relacional (ORM), mesmo sabendo que o Hibernate e o JPA são amplamente utilizados para esta finalidade. Em vez disso, adotamos a estrategia de utilizar na subclasse o mesmo valor do campo chave usado na superclasse e assim implementamos a herança, tão comum no paradigma Orientado a Objetos.

Omitimos intencionalmente os atributos das tabelas do Banco de Dados, representados na (Figura 9), pois o objetivo aqui e demonstrar quais tabelas foram implementadas e como se dá o relacionamento entre elas.

Figura 9 – Diagrama EER



4.7 Padrão de Projeto

Outra disciplina que merece destaque é Tópicos Especiais em Engenharia de Software III ministrada pelo professor Vitor Souza onde tivemos contato com o padrão de projeto DAO(Data Access Object).

O padrão DAO emana da necessidade de separarmos a lógica de negócios da logica da persistência, isso torna possível mudar a lógica de persistência sem causar impacto na logica de negócios além de melhorar a legibilidade do código-fonte impacta positivamente a manutenibilidade.

Toda a logica de acesso ao banco de dados foi implementada utilizando o padrão DAO, isto facilita bastante a implementação do CRUD e inclusive auxilia na implementação da interface gráfica com o usuário, assim como a logica de resposta a eventos.

Criamos uma ConnectionFactory a qual implementa o design patern Factory (GAMMA, et all. 2009), aqui o objetivo é promover o reuso do código que fornece uma conexão com o Banco de Dados e ao mesmo tempo favorecer a manutenibilidade.

O sistema em desenvolvimento utilizara intensamente de conexão com o Banco de Dados logo, reduzir a reescrita de código para obtenção desta conexão é certamente desejável, haja vista que agrega qualidade e produtividade ao processo desenvolvimento de Software.

4.8 Observação Experimental

Ao trabalhar com banco de dados, constatamos que as consultas baseadas em strings devem ser utilizadas com cautela, pois os espaços em branco são contabilizados como parte integrante da string. Dependendo da forma como é feita a entrada dos dados via interface gráfica, usando por exemplo um campo de texto(JtextField) espaços em branco podem ser adicionados sem que usuário perceba e desta forma, se não houver o tratamento adequado, tem-se um bug difícil de ser descoberto, principalmente quando o programador não vivenciou esta situação anteriormente. Entende-se que o tratamento adequado das strings agregará confiabilidade ao sistema em desenvolvimento.

5. Considerações Finais

Durante o desenvolvimento deste trabalho percebemos logo no inicio sua função pedagógica, o trabalho de conclusão de curso oferece ao graduando a oportunidade de aparar algumas arestas que ocorreram no processo de ensino e aprendizagem. Foi possível agregar novos conhecimentos e experiencia pois buscamos na literatura as informações necessárias pois não tínhamos todas as respostas, restou patente que o objetivo da graduação não é esgotar o conhecimento sobre assuntos específicos mas capacitar o graduando para que nos momentos de necessidade consiga encontrar as respostas.

5.1 – Conclusões

Para que fosse possível a realização deste trabalho, com relativo sucesso, percorremos as principais etapas do método científico a partir do momento em que identificamos um problema para o qual gostaríamos de propor uma solução: automatizar a alocação de recursos às aeronaves que realizam voos regulares chegando ou partindo do aeroporto de Vitoria-ES.

De inicio, o trabalho consistiu em identificar a classe a qual o problema pertence, onde baseado nos estudos de Programação Linear e Otimização verificamos que tratava-se de um problema da classe NP.

Apos identificar a categoria do problema, fomos buscar na literatura compreender algumas formas de solucionar o referido, e a partir dai criamos uma estrategia de solução. Saliente-se que não buscamos uma solução ótima, mas em vez disso percebemos que poderíamos propor uma boa solução. Assim são resolvidos problemas complexos para os quais seria inviável garantir uma solução ótima.

Para resolver o problema conforme propomos, observamos a possibilidade de desenvolver um sistema desktop, utilizando uma linguagem de programação orientada a objetos. Java SE foi a linguagem escolhida.

Constatamos ainda a necessidade de realizar a persistência de dados usando um SGBD relacional. Neste caso especifico abrimos mão de utilizar um framework de persistência baseado em Mapeamento Objeto Relacional pois julgamos importante conhecer a API JDBC do Java.

Valorizamos também a questão da usabilidade e por esta razão implementamos a Interação Homem Maquina a partir da criação de uma GUI utilizando Swing.

Vale salientar que procuramos utilizar alguns dos conhecimentos de Engenharia de Software razão pela qual um diagrama de classes foi a base para implementação do software e também para a geração do Banco de Dados atraves do diagrama EER o qual implementamos usando o Mysql Workbench.

Procuramos utilizar alguns Design Patterns com o objetivo de agregar qualidade e profissionalismo à solução que propomos, por isso, durante o desenvolvimento do software requisitos de qualidade foram uma importante baliza a nos guiar.

Constatamos que o padrão MVC permite uma boa organização do código e desta forma favorece a manutenibilidade, além disso, a utilização de uma Conection Factory favorece o reuso de código pois concentra em uma única classe a conexão com o Banco de Dados.

Sempre que possível, procuramos utilizar software livre pois entendemos que esta é uma tendencia no meio acadêmico, além disso, os custos da utilização de software proprietário tornariam inviável a implementação do software conforme propomos, haja vista as diversas tecnologias utilizadas a saber: IDE, SGBD, Linguagem de Programação, software de prototipação de uma parte da GUI, software para criação dos diagramas UML, software gerador de gráficos e software para desenho.

Portanto, atraves deste trabalho de conclusão de curso foi possível implementar um software o qual pode ser proposto como uma melhoria do Software Banco de Dados Operacionais, utilizado pela Infraero como ferramenta de controle operacional, especificamente no que concerne a alocação dos recursos aeroportuários conforme definimos como sendo o objetivo principal deste trabalho. Salientamos que foi possível utilizar-se dos ensinamentos de Engenharia de Software, Programação Linear e Otimização, Banco de Dados Relacional, SQL, Linguagens de Programação, paradigmas de programação Orientado a Objetos, Programação Orientada a Eventos, Metodologia de Pesquisa, Lógica, Estruturas de Dados. Desta forma, apresentamos o Sistema de Alocação de Recursos Aeroportuários o qual entendemos nos permite cumprir com os requisitos deste Trabalho de Conclusão do Curso de Ciência da Computação.

5.2 – Trabalhos Futuros

Sugerimos como trabalhos futuros dotar o sistema SARA de capacidade de aprendizado utilizando Inteligencia Artificial, converter o sistema de desktop para sistema Web ou Mobile, implementar uma alocação eficiente de recursos para voos do Grupo II, ou seja voos que chegam esporadicamente e não tem Hotran. Poder-se-ia Implementar o Banco de Dados utilizando ORM, poderíamos ainda aplicar a estrategia proposta aqui para outros modais de transporte por exemplo portos e pátios de carga e descarga de veículos, pode-se ainda realizar um estudo comparativo entre a solução hora apresentada com outras soluções baseadas em meta-heurísticas. Além disso, pode-se realizar um estudo das melhores praticas de alocação de recursos aplicadas à aviação civil.

REFERÊNCIAS

[1] ANAC, Regulamento Brasileiro de Aviação Civil - RBAC 01 Emenda 8 – Definições, Regras de Redação e Unidades de Medida para Uso nos Normativos da ANAC. 12 de jan. de 2021. Disponível em: https://www.anac.gov.br/assuntos/legislacao/legislacao-1/rbha-e-rbac/rbac/rbac-01 acesso em: 24 de abr. de 2021.

[2] INFRAERO, 19 DE ABR.2018 Declaração de Capacidade de SBVT. Disponível em: http://www4.infraero.gov.br/media/675069/declaracao-de-capacidade-sbvt-w18.pdf. Acesso: 24 de abr. de 2021.

[3] ANAC. Nov. 2020. Código de Referência de Aeronaves. Disponível em: https://www.anac.gov.br/assuntos/setor-regulado/aerodromos/downloads/aeronaves-e-codigo-de-referencia acesso: 20 de mar. De 2021.

[4] ANAC Agencia Nacional de Aviação Civil. 20 de mar.de 2021. Institucional. Disponível em: https://www.anac.gov.br/acesso-a-informacao/institucional acesso em: 20 de mar. De 2021.

[5] ABRAMSON, David. Constructing school timetables using simulated annealing: sequential and parallel algorithms. Management science, v. 37, n. 1, p. 98-113, 1991.

[6] SOUZA, Marcone Jamilson Freitas et al. Métodos de pesquisa em vizinhança variável aplicados ao problema de alocação de salas. XXII Encontro Nacional de Engenharia de Produção ENEGEP, Fortaleza, Brasil, 2002.

[7] MALNOR, Cody et al. Validação de modelos ER. In: Anais do XXVI Workshop sobre Educação em Computação. SBC, 2018.

[8] PRESMAN,, R. S. (2011). Engenharia de Software. 7ª ed., São Paulo, SP: Makron Books.

[9] GAMMA, Erich. **Padrões de projetos: soluções reutilizáveis**. Bookman editora, 2009.

[10] AELA.IO Heurísticas de Nielsen: 10 Dicas para melhorar a Usabilidade de sua Interface. 17 de jul. 2019. Disponivel em: https://medium.com/aela/10-heur %C3%ADsticas-de-nielsen-dicas-para-melhorar-a-usabilidade-de-sua-interface-35ef86a7fb41 acesso em: 22 de março de 2021.

[11] LUCIDCHART, O que é um diagrama UML? 21 de setembro de 2019. Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-uml acesso em : 21 de setembro de 2019.

[12] RUGGERO RUGGIERI ,Análise sobre a ISO 9126 – NBR 13596. 24 de out. de 2016. Disponível em: https://www.tiespecialistas.com.br/analise-sobre-iso-9126-nbr-13596/ acesso em 22 de dez. De 2019.

[13] FRANZINI, Java Frameworks Persistencia em Banco de Dados Relacionais. 13 de mar. de 2017. Disponível em: https://fernandofranzini.wordpress.com/2017/02/23/java-frameworks-persistencia-em-banco-de-dados-relacionais/ acesso: 22 de mar. De 2021.

[14] WIKIPEDIA, MVC. 15 de mar. de 2021. Disponível em: https://pt.wikipedia.org/wiki/MVC. Acesso em: 15 de mar. De 2021.

[15] FARIAS. Por que as pistas de pouso às vezes mudam de numeração das cabeceiras? 01 de out. de 2020. Disponível em: https://www.aeroin.net/pistas-pouso-mudam-numeracao-cabeceiras/ acesso: em 16 de mar. De 2021.
[16] ABRAMSON, David. Constructing school timetables using simulated annealing: sequential and parallel algorithms. Management science, v. 37, n. 1, p. 98-113, 1991.

[17] BOOCH, G.; Rumbaugh,J.;Jacobson, I. UML guia do Usuário tradução de Fabio Freitas da Silva . Rio de Janeiro: Elsevier, 2000. 12ª reimpressão ISBN 85-352-0562-4.

[18] MACEDO. 02 de ago. de 2017. 10 heurísticas de Nielsen para o design de interface. Disponível em: https://brasil.uxdesign.cc/10-heur%C3%ADsticas-de-nielsen-para-o-design-de-interface-58d782821840 acesso: 24 de abr. de 2021.

[19] TODESCO, Fabiana; MÜLLER, Carlos. OTIMIZAÇÃO DA DESIGNAÇÃO DE VÔOS AOS GATES NO AEROPORTO INTERNACIONAL DE SÃO PAULO/GUARULHOS.

[20] IATA, 2003 – O VALOR DO TRANSPORTE AEREO NO BRASIL - A IMPORTÂNCIA DO TRANSPORTE AÉREO PARA O BRASIL, acessado em 03/04/2021.

[21] ECLIPSE, 26 de abr. de 2021. WindowBuilder - é um designer de GUI Java bidirecional poderoso e fácil de usar. Disponível em:https://www.eclipse.org/windowbuilder/ acesso: 26 de abr. de 2021.

[22] VIEIRA, H. C. R.; BARANAUSKAS, Maria Cecília Calani. Design e avaliação de interfaces humano-computador. Campinas: Unicamp, 2003.

[23] INFRAERO, jul. 2004 SISO/BDO Sistema Integrado de Solução Operacional e Banco de Dados Operacional. Disponível em: https://licitacao.infraero.gov.br/arquivos_licitacao/2011/SRSE/013_ADSE-3_SRSE_2011_TP/Anexo%201%20SO_BDO_Caderno_Funcional_julho2004.pdf acesso: 26 de abr. de 2021.

[24] ANAC, 27 de abr .de 2021. ANACPÉDIA . Disponível em: https://www2.anac.gov.br/anacpedia/por_por/tr3258.htm acesso: 27 de abr. 2021.

[25] NOGUEIRA, Kamila Borges; AGUIAR, Paulo Henrique Cabral. Aplicação do sistema formiga para gerência do sequenciamento de taxiamento em aeroportos. 2013.