UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO DE CIÊNCIAS EXATAS, NATURAIS E DA SAÚDE
DEPARTAMENTO DE COMPUTAÇÃO

Marcelo Côgo

Desenvolvimento de um Sistema para Automação da Elaboração de Quadro de Horários e Disciplinas

ALEGRE
2018

MARCELO CÔGO

Desenvolvimento de um Sistema para Automação da Elaboração de Quadro de Horários e Disciplinas

Trabalho de conclusão de curso apresentado ao Departamento de Computação do Centro de Ciências Agrárias da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. D.Sc. Helder de Amorim Mendes.

ALEGRE - ES

2018

Desenvolvimento de um Sistema para Automação da Elaboração de Quadro de Horários e Disciplinas

Trabalho de Conclusão de Curso apresentado ao Departamento de Computação do Centro de Ciências Exatas, Naturais e da Saúde da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 26 de Novembro de 2018.

COMISSÃO EXAMINADORA

Prof. D.Sc. Helder de Amorim Mendes
Universidade Federal do Espírito Santo
Orientador

Prof. D.Sc. Geraldo Regis Mauri
Universidade Federal do Espírito Santo

Prof. M.Sc. Jacson Rodrigues Correia da Silva
Universidade Federal do Espírito Santo

Dedico todo esse trabalho aos meus pais, José Edir e Maria das Graças, por todo incentivo, apoio e força que me deram.

AGRADECIMENTOS

Primeiramente, agradeço a Deus por tudo o que Ele tem feito em minha vida ao longo desses últimos anos.

Agradeço aos meus pais pelo apoio e por me incentivarem nos momentos que mais precisei. Sem eles não seria possível a realização deste trabalho.

Aos amigos da República os Pedreiros com quem compartilhei momentos de alegria, tristeza e ansiedade, sendo parte da minha formação.

À Ana Caroline, pelo companheirismo, motivação e pelo amor incondicional, que mesmo nos momentos de minha ausência dedicados aos estudos me incentivou nas horas que mais precisei ajudando a superar todos os obstáculos.

Ao professor orientador Helder, por toda a atenção e comprometimento ao longo do desenvolvimento deste trabalho e por todo conhecimento passado ao longo do curso. Aos demais professores do departamento pela contribuição na minha formação.

RESUMO

O processo de geração da grade horária de uma escola é complexo e influencia a rotina diária de todos os envolvidos durante um período letivo. Os coordenadores de cursos de instituições de ensino encontram muitas dificuldades na hora de elaborar o quadro de horários de disciplinas de seus cursos. Vários critérios pedagógicos dificultam esse processo que normalmente é feito de forma manual, exigindo muito trabalho e o comprometimento de várias pessoas. Neste trabalho, desenvolveu-se uma aplicação que permite automatizar a elaboração do quadro de horários e disciplinas tomando como base o centro de Ciências Exatas, Naturais e da Saúde da Universidade Federal do Espírito Santo (CCENS-UFES), com o auxílio de um otimizador meta-heurístico de horários que pode ser integrável ao sistema, possibilitando assim, gerar horários de forma rápida e menos onerosa.

Palavras-chave: Programação De Horários, Desenvolvimento de *Software*, Meta-heurística, Otimizador de Horários.

Development of a System for Elaboration of Timesheet and Subjects

ABSTRACT

The process of generating a school's schedule is complex and influences the daily routine of all involved during a school term. The coordinators of courses in educational institutions find it very difficult to draw up the timetable of their course subjects. Several pedagogical criteria hamper this process, which is usually done manually, requiring a lot of work and the commitment of several people. In this work, an application was developed that automates the elaboration of schedules and subjects based on the Exact, Natural and Health Sciences Center of the Federal University of Espírito Santo (CCENS-UFES), with the aid of an optimizer heuristic of schedules that can be integrated into the system, making it possible to generate schedules quickly and less costly.

Keywords: Time Schedule, Software Development, Heuristics, Time Optimizer.

LISTA DE DIAGRAMAS

Diagrama 1 - Diagrama de Caso de Uso. 27

Diagrama 2 - Modelo Entidade Relacionamento 29

LISTA DE FIGURAS

Figura 1. Camadas da Engenharia de *Software*. 15

Figura 2 - Diagrama de Fluxo de Atividades do Projeto. 22

Figura 3 - Fluxo de eventos e informações em uma arquitetura MVC. 23

Figura 4 - Tela de Login. 31

Figura 5 - Mensagem de erro de login 32

Figura 6 – Dashboard 33

Figura 7 - Tela de Cadastro de Períodos. 34

Figura 8 - Listagem de cursos 34

Figura 9 - Cadastro de curso 35

Figura 10 - Listagem de disciplinas 35

Figura 11 - Cadastro de disciplina 36

Figura 12 - Listagem de Locais 37

Figura 13 - Horários de aula por sala 38

Figura 14 - Listagem de Professores 38

Figura 15 - Cadastro de Professor 39

Figura 16 - Tela de ajuste de horário 40

Figura 17 - Cadastro de turmas. Aba Turmas 40

Figura 18 - Aba Disciplinas/Professores/Salas 41

Figura 19 - Aba Horários 42

Figura 20 - Tela home. 43

LISTA DE TABELAS

Tabela 1 - Requisitos não funcionais 26

LISTA DE ABREVIATURAS E SIGLAS

ALNS – Adaptive Large Neighborhood Search

CCA-UFES – Centro de Ciências Agrárias da Universidade Federal do Espírito

CCENS – Centro de Ciências Exatas, Naturais e da Saúde

CS – Clustering Search

CSS - Cascading Style Sheets

ECS – Evolutionary Clustering Search

FAMINAS – Faculdade de Minas

FO – Função Objetivo

GRASP – Greedy Randomized Adaptive Search Procedure

IDE - Integrated Development Environment.

ILS – Iterated Local Search

MER – Modelo Entidade Relacionamento

MVC - Model, View, Controller

OO – Orientação a Objetos

PHE – Programação de Horários Escolares

PPH- Problema de Programação de Horários

PPHDU – Problema de Programação de Horários de Disciplinas em Universidades

SA – Simulated Annealing

SGBD - Sistema Gerenciador de Banco de Dados

UC - User Case

UFES – Universidade Federal do Espírito Santo

UML - Unified Modeling Language

UNIRIO – Universidade Federal do Estado do Rio de Janeiro

VND – Variable Neighborhood Descent

SUMÁRIO

1 INTRODUÇÃO 12

1.1 O Problema e sua Importância 12

1.2 Objetivos 14

1.2.1 Objetivo geral 14

1.2.2 Objetivos específicos 14

2 REVISÃO DA LITERATURA 14

2.1 Engenharia de *Software* 14

2.1.1 Processo de Software 15

2.1.2 Sistemas 16

2.1.3 Testes De *Software* 17

2.2 Otimização Combinatória 18

2.3 Softwares para automação da elaboração de horários 19

3 METODOLOGIA 20

3.1 Levantamento de Requisitos 20

3.2 Ferramentas 20

4 RESULTADOS 24

4.1 Análise de Requisitos 24

4.1.1 Requisitos Funcionais 24

4.1.2 Requisitos não-funcionais 26

5 CONCLUSÃO 44

6 REFERÊNCIAS 45

1 INTRODUÇÃO

Elaborar o quadro de horários em instituições de ensino, com grande número de cursos e disciplinas, é uma tarefa árdua e que pode levar muito tempo para ser realizada. Este problema é conhecido como Problema de Programação de Horários (PPH), e é um problema clássico de otimização combinatória, pois trata-se da alocação de recursos em determinados horários, satisfazendo restrições, e minimizando ou maximizando uma função matemática, segundo Schaerf (1999) e Souza (2000).

A alocação dos horários das disciplinas tem forte influência na atividade de uma instituição de ensino durante um período letivo, sendo necessário que todos os envolvidos se adaptem à mesma. Segundo Santos e Souza (2007), a natureza dos requisitos de um problema de programação de horários pode ser dividida em três categorias, sendo estas organizacionais, pedagógicos e pessoais. As organizacionais tratam questões como gestão de recursos e cumprimento da legislação atual da instituição, como por exemplo, a aula de Estrutura de Dados I deverá ser realizada em um laboratório de informática. Os requisitos pedagógicos, uma vez cumpridos, podem melhorar o desempenho nas aulas; isto é, maior duração de aula para que seja possível realizar determinadas atividades em uma disciplina. Por fim, as categorias pessoais estão relacionadas com as necessidades pessoais dos docentes, por exemplo, um determinado professor prefere que suas aulas em determinado dia da semana ocorram no período vespertino.

Uma grande diversidade de problemas associados a esse assunto, como apresentados por Berardino (2013), Mariano (2014) e Carvalho (2015) já foram relatados, e diferentes soluções foram propostas, sendo estas tipicamente dependentes das situações específicas de uma instituição. Assim, não há uma solução universal que possa ser adotada em todos os casos.

1.1 O Problema e sua Importância

A grande quantidade de restrições de horários das disciplinas e a dificuldade de satisfazer todas as preferências de horários dos professores de diferentes departamentos fazem com que a obtenção da grade horária final para o semestre seja uma tarefa complexa. Atualmente esse trabalho é realizado de forma manual na maioria das instituições de ensino, resultando em uma demanda concentrada de esforço e

comunicação. Além disso, a probabilidade de ocorrerem erros é alta, podendo gerar insatisfação dos interessados, professores e alunos.

Mariano (2014) apresentou uma proposta de solução para o problema da elaboração de grade horária para um departamento no Centro de Ciência Agrárias da Universidade Federal do Espírito Santo (CCA-UFES), e classificou o problema em dois grupos de restrições, as fortes e as fracas.

Segundo Santos e Souza (2007):

Restrições Fortes: devem ser satisfeitas a qualquer custo. O não atendimento desse tipo de restrição inviabiliza a solução;

Restrições Fracas: são aquelas cuja satisfação é desejável, mas caso não seja possível atendê-las, a solução não é inviabilizada.

As principais restrições fortes do problema de programação de horários aplicado à realidade do CCA-UFES são as seguintes, segundo Mariano (2014):

1. Conflitos de professor: um professor não poderá ministrar mais de uma disciplina no mesmo dia e horário;

2. Conflitos de turmas: uma turma não poderá assistir a mais de uma aula no mesmo dia e horário;

3. Conflitos de salas: uma sala de aula não poderá estar reservada para mais de uma disciplina no mesmo dia e horário;

O problema de programação de horários é um problema de fácil entendimento, mas de difícil solução, e é necessário um grande esforço para encontrar uma solução quando feita de forma manual.

A partir disso, viu-se a necessidade de um sistema que a partir das informações dos cursos, disciplinas, professores alocados nas ofertas das disciplinas, salas, preferências e restrições de horários, possibilite a montagem da grade de horários para um período letivo de uma instituição de ensino auxiliando o trabalho do responsável pela sua confecção uma vez que podem ocorrer inconsistência quando feita de forma manual.

1.2 Objetivos

1.2.1 Objetivo geral

O objetivo principal deste trabalho consiste no desenvolvimento de um sistema *desktop* para a entrada de dados, montagem e visualização de quadros de horários de professores e disciplinas que possa ser integrável com ferramentas de geração automática de horários.

1.2.2 Objetivos específicos

Para atingir o objetivo geral, é necessário cumprir alguns objetivos específicos:

a) Projeto e implementação de um módulo para a entrada de dados, cadastro e edição de professores, disciplinas e cursos.
b) Projeto e implementação de um módulo para a montagem e edição da grade de horários.
c) Implementação da integração com a ferramenta para geração automática da grade de horários.

2 REVISÃO DA LITERATURA

Para o desenvolvimento do trabalho é necessário compreender duas áreas de conhecimento: Engenharia de *Software*, para auxiliar o desenvolvimento do sistema e, Otimização Combinatória, para compreender o funcionamento do processo de otimização de horários e assim realizar a integração do sistema com o otimizador heurístico.

2.1 Engenharia de *Software*

A engenharia de *software* é uma disciplina da engenharia que se ocupa de todos os aspectos da produção de *software*, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, depois que ele entrou em operação (SOMMERVILLE, 2007).

PRESSMAN (2006) trata a engenharia de *software* como uma tecnologia em camadas conforme a Figura 1. Toda iniciativa de engenharia de *software* dever ser apoiada por um compromisso com a qualidade. Acima da camada da qualidade encontram-se os processos, logo acima, os métodos e, acima destes, as ferramentas. Foram concebidos vários modelos de processos de *software* e nenhum pode ser

considerado o ideal, devido às suas divergências. Entretanto, para SOMMERVILLE (2007), todos compartilham de atividades fundamentais como especificação, projeto e implementação, validação e evolução.

Figura 1. Camadas da Engenharia de Software. Fonte: Adaptada de Sommerville (2007)

2.1.1 Processo de Software

Sommerville (2007) define que um processo de *software* é um conjunto de atividades e resultados associados que produz um produto de *software*. Existem quatro atividades fundamentais, que são comuns a todos os processos de *software*: especificação, desenvolvimento, validação e evolução. Respectivamente ocorre a definição do *software* a ser produzido e as restrições para a sua operação, logo após é feita sua projeção e programação bem como se faz necessária a validação onde é verificado para garantir os desejos do cliente, por fim o *software* é modificado para se adaptar às mudanças e requisitos do usuário final.

Existem alguns modelos teóricos desenvolvidos que buscam descrever a forma com que as fases seguem e interagem. Em outras palavras, um modelo é uma filosofia do andamento das fases, e não uma descrição de como cada atividade deve ser executada.

Segundo Sommerville (2007) e Pressman (2006), existem diversos modelos de processo de *software*. Os mais conhecidos são:

- Modelo sequencial linear ou modelo em cascata: considera as atividades apresentadas anteriormente e as representa como fases separadas de processo, como especificação de requisitos, projeto de *software*, implementação, teste e assim por diante. Depois que cada estágio é concluído, ele é aprovado e o desenvolvimento prossegue para o estágio seguinte.
- Desenvolvimento Iterativo: esta abordagem intercala as atividades de especificação, desenvolvimento e validação. Um sistema inicial é desenvolvido rapidamente com base em especificações muito abstratas. É então refinado com as informações do cliente, para produzir um sistema que satisfaça as

necessidades deste. O Sistema pode, então, ser entregue. Como alternativa ele pode ser reimplementado, utilizando uma abordagem mais estruturada, para produzir um sistema mais robusto e mais fácil de ser mantido.

- Engenharia de *Software* baseada em componentes: esta técnica supõe que partes do sistema já existam. O processo de desenvolvimento concentra-se mais na integração dessas partes do que no seu desenvolvimento a partir do início.

2.1.2 Sistemas

No que diz respeito especificamente aos sistemas computacionais, estes podem ser enumerados como Sistemas de *Software On-line*, Sistemas de *Software* de Tempo Real, Sistemas de *Software* de Apoio à Decisão e Sistemas de Planejamento Estratégico e Sistemas de *Software* Baseados no Conhecimento.

Sistema de *Software* On-line é um tipo de sistema onde os dados são trocados de forma remota, em que existe uma distância entre o emissor e o emitente da informação. São aqueles que recebem entradas diretamente do local onde estas são criadas. São também os sistemas em que as saídas, ou os resultados do processamento são encaminhados diretamente para onde solicitados (YORDON, 1990).

Os Sistemas de Tempo Real são caracterizados principalmente pelo tempo de resposta que eles necessitam, que é extremamente alto, ou de acordo com YORDON (1990) “um sistema de processamento em tempo real pode ser definido como aquele que controla um ambiente pelo recebimento de dados, seu processamento e apresentação dos resultados com rapidez, suficiente para afetar o ambiente naquele momento”.

Sistemas de *Software* de Apoio à Decisão e Sistemas de Planejamento Estratégico ajudam as empresas a seguirem um rumo. Alguns destes sistemas são planilhas eletrônicas, sistemas de análise de estatísticas entre outros. No início, quando os sistemas computacionais foram implantados em empresas, eles cuidavam da parte operacional da empresa. No entanto, nos últimos anos a necessidade de um melhor posicionamento no mercado acabou facilitando a difusão dos sistemas de apoio à decisão que é definido por YORDON (1990) como sistemas de processamento “que auxiliam gerentes e outros profissionais funcionários do conhecimento de uma organização a tomarem as decisões inteligentes e bem informadas sobre vários aspectos da operação”. Alguns destes sistemas são planilhas eletrônicas, sistemas de análise de estatísticas entre outros.

Os Sistemas Baseados no Conhecimento começaram a ser estudados com uma parte da inteligência artificial. Estes sistemas tentam encontrar informações dentro de dados que foram coletados pelas empresas durante anos. Os sistemas baseados no conhecimento podem ser descritos como sistemas que "contêm grande quantidade de conhecimentos variados que eles trazem para utilização em determinada tarefa" (YORDON,1990).

2.1.3 Testes De *Software*

Segundo Sommerville (2007), o processo de teste de *software* tem duas metas distintas:

1. Demonstrar ao desenvolvedor e ao cliente que o *software* atende aos requisitos. Para *softwares* sob encomenda, isso significa que deve haver pelo menos um teste para cada requisito contido nos documentos de usuário e de sistema. Para produtos de *software* genéricos, isso significa que deve haver testes para todas as características do sistema.
2. Descobrir falhas ou defeitos no *software* que apresenta comportamento incorreto, não desejável ou em não conformidade com sua especificação. O teste de defeitos está relacionado à remoção de todos os tipos de comportamentos indesejáveis de sistema, como travamentos, interações indesejáveis com outros sistemas, cálculo incorretos e corrompimento de dados.

Os testes não podem demonstrar que um sistema é livre de defeitos ou que ele se comportará conforme o esperado em todos os casos, segundo Dijkstra (1969). Portando, a meta do teste de *software* é convencer os desenvolvedores e clientes do sistema de que o *software* é bom o suficiente para o uso operacional.

O teste de sistema envolve a integração de dois ou mais componentes que implementam funções ou características do sistema e depois o teste desse sistema integrado. Em um processo de desenvolvimento iterativo, o teste de sistema se concentra no teste de um incremento que será entregue ao cliente.

Sommerville (2007) define que existem duas fases distintas de teste do sistema, sendo estes testes de integração e testes de releases. Os de integração são aqueles onde a equipe de testes deve acessar o código fonte do sistema. Quando um problema é encontrado, a equipe de integração tenta encontrar a origem do problema e identificar os componentes que devem ser reparados. Os testes de integração geralmente estão relacionados à descoberta de defeitos no sistema. Os testes de releases são aqueles nos

quais uma versão do sistema que poderia ser liberada aos usuários é testada. Nessa etapa consiste em validar se o sistema atende aos requisitos e em assegurar que o sistema é confiável. Os clientes são envolvidos nos testes de releases, que em algumas vezes é chamado teste de aceitação. Se o release for bom o suficiente, o cliente poderá aceita-lo para o seu uso.

2.2 Otimização Combinatória

A Otimização Combinatória é um ramo da ciência da computação e da matemática aplicada que estuda problemas de otimização em conjuntos finitos. Em um problema de otimização temos uma função objetivo e um conjunto de restrições, relacionados às variáveis de decisão dos problemas. Os valores possíveis para as variáveis de decisão são delimitados pelas restrições impostas por elas, formando assim um conjunto de soluções possíveis a um problema. O problema pode ser de minimização ou de maximização da função objetivo, ou seja, o que se deseja otimizar.

Há muitas classificações para um problema de otimização, as que apresentam métodos exatos e eficientes de resolução e outras que levarão à necessidade de métodos não exatos (heurísticas/meta-heurísticas), uma vez que sua formulação e resolução exatas levariam a uma complexidade intratável.

A programação de horários de universidades é um problema clássico de otimização combinatória. Encontrar uma solução ótima ou soluções consideradas boas que se aproximam do ótimo, geralmente não é uma tarefa fácil e nem sempre é possível de ser realizada em tempo hábil. As meta-heurísticas são ferramentas que buscam encontrar soluções de boa qualidade em tempo computacional razoável para problemas de otimização complexos (CHAVES, 2009). Elas são estratégias comumente utilizadas para resolver problemas NP – Difíceis por oferecerem melhores soluções e geralmente com tempo de processamento menor do que por outros tipos de técnicas. De forma geral, utilizam combinação de escolhas aleatórias e conhecimento histórico (dos resultados anteriores adquiridos pelo método) para se guiarem e realizar suas buscas pelo espaço de pesquisa em vizinhanças dentro do espaço de pesquisa, o que evita paradas prematuras em ótimos locais.

No trabalho "*An algorithm for constructing University timetables*", Csima e Gotlieb (1964) foram os percursores na automação do processo de gerar quadros de horários. Os autores propuseram um método envolvendo matrizes binárias.

Abramson (1991) aponta como solução do problema de programação de horários, a utilização do *Simulated Annealing* de forma paralela. Por sua vez Colorni et al. (1998) propuseram três distintas soluções. Na primeira, empregam o *Simulated Annealing*, e na segunda os autores aplicam a Busca Tabu. Por fim, na terceira solução, exploram o uso do Algoritmo Genético com e sem busca local.

Ramos (2002) desenvolveu um Sistema Automático de Geração de Horários para a UFLA (Universidade Federal de Lavras), utilizando Algoritmos Genéticos. O sistema não obteve bons resultados devido às inúmeras restrições do problema e na dificuldade de se combinar parâmetros para que uma boa solução fosse encontrada.

Mariano (2014) resolveu o PPHDU (Problema de Programação de Horários de Disciplinas em Universidades) aplicando a meta-heurística Adaptive Large Neighborhood Search (ALNS) ao Departamento de Computação do CCA-UFES. O autor obteve resultados melhores quando comparados com o quadro de horários elaborado de forma manual.

Carvalho (2015) resolveu o PPHDU (Problema de Programação de Horários de Disciplinas em Universidades) referente ao CCA-UFES, aplicando as metas-heurísticas *Simulated Annealing* (SA) e *Clustering Search* (CS). Sendo assim o método produzido por este trabalho será utilizado como o otimizador heurístico de horários a ser integrado ao sistema.

2.3 Softwares para automação da elaboração de horários

Berardino (2013) desenvolveu um *software* para geração da Grade Horária do curso de Bacharelado em Sistemas de Informação da Universidade Federal do Estado do Rio de Janeiro (UNIRIO), e obteve resultados satisfatórios. Porém, sua utilização era muito trabalhosa, já que não apresentava uma interface única que fizesse todo o processo. A utilização de diferentes paradigmas de solução, tais como Excel, Java e Prolog trouxeram desafios não só dentro do próprio paradigma como também de integração entre eles.

Existem muitas ferramentas na web voltadas para a elaboração da grade de horários de professores e disciplinas, porém com foco no ensino básico, que contemplam colégios de ensino médio ou às escolas de ensino fundamental. Como exemplo dessas ferramentas podemos citar o GridClass (http://gridclass.com.br) e Cronos (https://www.cronostimetable.com). Dentre suas principais vantagens se destacam sistema 100% WEB, horário sem conflitos, junção e/ou divisão de turmas para determinadas aulas, customização do sistema, solicitação de disponibilidade

automatizada com acompanhamento remoto (envio de e-mail aos professores com os horários disponíveis e possibilidade de cadastro da disponibilidade), cadastro de disponibilidade flexível e gestão da disponibilidade dos professores.

3 METODOLOGIA

A metodologia deste trabalho está dividida em três partes, sendo levantamento e análise de requisitos, ferramentas utilizadas e, por fim, os métodos utilizados no processo de desenvolvimento do sistema.

3.1 Levantamento de Requisitos

Por meio de entrevistas, junto a coordenadores de cursos do CCENS - UFES, foram levantadas informações do processo de elaboração das grades de horários e as dificuldades encontradas em sua confecção. Além das entrevistas, foi realizada uma coleta de dados em outros sistemas e trabalhos anteriores como os propostos por Mariano (2014) e Carvalho (2015).

3.2 Ferramentas

Para a modelagem do sistema foi utilizada a Linguagem de Modelagem Unificada (*Unified Modeling Language* - UML) que exprime na forma de diagramas padronizados uma noção abstrata do comportamento do sistema. Adotado o diagrama de caso de uso, foi definida sua estrutura com o intuito de mostrar as funcionalidades da aplicação.

Para a modelagem do banco de dados foi utilizado o modelo de entidade e relacionamento (MER) que fornece uma representação abstrata do banco de dados.

Optou-se, na construção do sistema pelo uso do ambiente de desenvolvimento integrado Netbeans IDE 8.2, pois além de possuir um amplo número de recursos avançados, uma extensa linha de ferramentas, modelos e exemplos facilita o desenvolvimento de interfaces gráficas integrando muito bem as bibliotecas Java Swing e JavaFx. Bibliotecas estas que permitem criar aplicações gráficas com uma vasta seleção de controles visuais que possibilitam a criação dos mais diversos tipos de aplicações. A biblioteca JavaFX possibilita, ainda, a separação entre a lógica do programa e a interface. Esta pode ser estruturada por uma linguagem de marcação criada especificamente para a plataforma Java denominada FXML.

No auxílio do desenvolvimento da interface gráfica foi utilizada a ferramenta JavaFx Scene Builder, um recurso de layout visual que permite aos desenvolvedores projetar rapidamente interfaces de usuário sem codificação. A ferramenta proporciona à seus usuários o recurso de arrastar e soltar componentes de interface em uma área de trabalho, modificar suas propriedades ou aplicar folhas de estilo com CSS, tudo isso, sem a necessidade de escrever código (Oracle, 2018).

3.2.1 Linguagem de Programação

Para a implementação da aplicação proposta neste trabalho, foram analisadas algumas linguagens de programação utilizadas no desenvolvimento de sistemas *desktop*. Nesse mercado, existem diversas linguagens de programação, que seguem diferentes paradigmas; um exemplo disto é a Orientação a Objetos (OO) que traz uma ideia muito interessante: a representação de cada elemento em termos de um objeto, ou classe. Esse tipo de ferramenta procura aproximar o sistema que está sendo criado ao que é observado no mundo real, e um objeto contém características e ações, assim como vemos na realidade. As linguagens mais utilizadas no desenvolvimento desse tipo de aplicação são Java, C#, C++ e Delphi, de acordo com pesquisas realizadas em diversos livros e sites.

Para o desenvolvimento do sistema proposto neste trabalho optou-se por utilizar Java, por ser uma linguagem de programação orientada a objetos que tem como um de seus princípios a portabilidade, que é a capacidade de ser interpretada em diversos sistemas operacionais e dispositivos sem a necessidade de reescrever o código, além de ser uma linguagem gratuita e com boa produtividade (Java, 2018).

3.3 Processo de Desenvolvimento

Inicialmente partiu-se do princípio de reutilizar o código de um sistema de controle acadêmico já existente, que foi desenvolvido para a Faculdade de Minas (FAMINAS), no entanto não seria viável a reutilização, pois o problema de programação de horários é de difícil generalização, como dito por Souza (2000). Cada sistema é desenvolvido para atender características específicas da instituição de ensino e do regime educacional da região onde ele será aplicado. Outro fator que dificultou a reutilização do *software* foi a tecnologia utilizada na implementação: a linguagem Delphi que, aplicada sem conceitos de orientação a objetos, prejudicaria o reuso e manutenibilidade do código.

O processo aplicado no desenvolvimento do sistema pode ser dividido em três etapas principais: desenvolvimento, teste e validação, representadas no diagrama de fluxo de atividades, Figura 2.

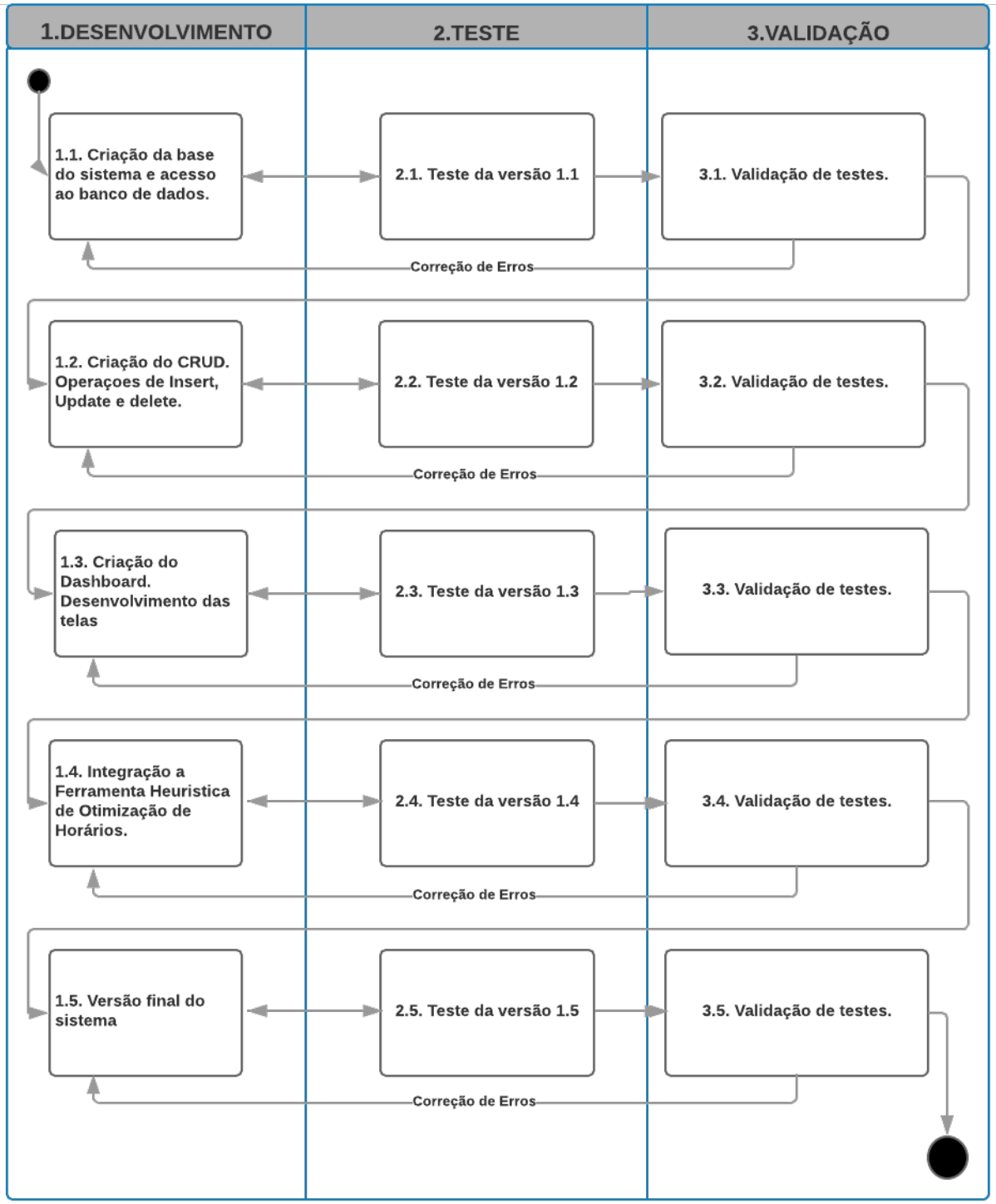

Figura 2 - Diagrama de Fluxo de Atividades do Projeto. Fonte: O autor.

Na etapa de desenvolvimento do projeto houve preocupação com a arquitetura da aplicação. Nessa fase foi dado valor à tecnologia diferente da etapa de análise onde apenas foi esboçado o problema.

De acordo com DOODLEY (2011), a arquitetura de um sistema tem diversos elementos como: elementos utilitários, elementos de interação, elementos que fazem parte do domínio do problema, elementos de conexão, elementos de persistência entre outros. Dessa forma, na arquitetura sempre é definida os seus elementos que precisarão ser utilizados no *software* e como eles se conectam. Alguns padrões arquiteturais já foram pensados para resolver problemas corriqueiros.

Decidiu-se adotar o padrão MVC (*Model-View-Controller*) para a arquitetura desse sistema, uma vez que possibilita a divisão em camadas muito bem definidas. A utilização do padrão MVC traz como benefício isolar as regras de negócios da lógica de apresentação, a interface com o usuário segundo Dooley (2011). Isto possibilita a existência de várias interfaces com o usuário que podem ser modificadas sem que haja necessidade da alteração das regras de negócios, proporcionando assim muito mais flexibilidade e oportunidades de reuso do código.

Na visão do padrão MVC tem-se primeiramente o controlador (*Controller*) que interpreta as entradas e ações do usuário em comandos que são enviados para o modelo (*Model*) e/ou para a janela de visualização (*View*) para efetuar a alteração apropriada. Por sua vez o modelo (*Model*) gerencia um ou mais elementos de dados e responde a instruções para mudar de estado. O modelo sabe o que o aplicativo quer fazer e é a principal estrutura computacional da arquitetura, pois é ele quem modela o problema que está se tentando resolver. Por fim, a visão (*View*) é responsável por apresentar as informações para o usuário através de uma combinação de gráficos e textos. A figura 3 demonstra de forma básica o fluxo de eventos e informações em uma arquitetura MVC.

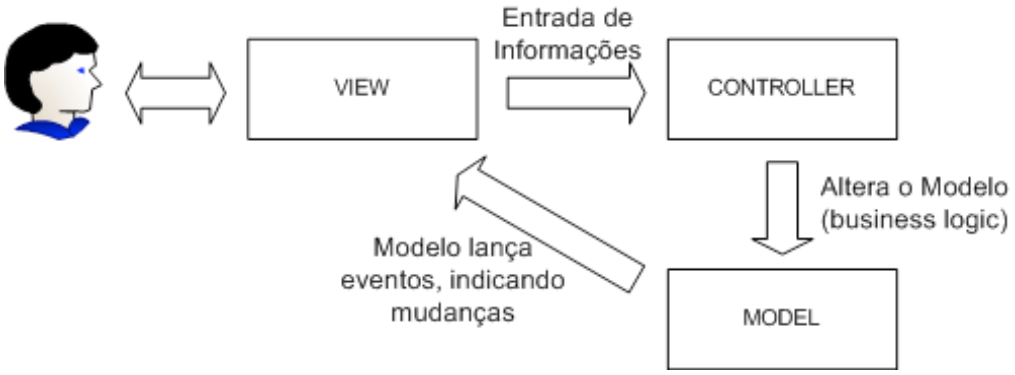

Figura 3 - Fluxo de eventos e informações em uma arquitetura MVC. Fonte: Sauvé (2018).

Como pode ser observado na Figura 2, o processo de desenvolvimento de *software* utilizado na construção do sistema foi o iterativo incremental, uma vez que, a cada etapa

concluída, foi agregada uma nova funcionalidade à aplicação. Testes menores e validações desses testes foram realizados junto ao processo de desenvolvimento de cada módulo com a intenção de testar pequenas partes isoladas.

Ao final da implementação das funcionalidades básicas de cada parte do sistema testes mais completos foram realizados. Esses testes tinham por objetivo testar a funcionalidade completa de cada parte desenvolvida. Ao término de todas as correções para os problemas encontrados, era considerada uma nova versão e uma nova etapa de desenvolvimento era iniciada.

4 RESULTADOS

A apresentação dos resultados foi organizada em três seções: apresentação dos requisitos que foram coletados e analisados, apontamento dos resultados da modelagem do sistema compreendendo o diagrama e as descrições dos casos de uso e o diagrama entidade relacionamento e, por fim, apreciação das telas de interface do sistema.

4.1 Análise de Requisitos

Como resultado do levantamento de requisitos realizado na Seção 3.1 obteve-se os requisitos funcionais e os não funcionais, descritos a seguir.

4.1.1 Requisitos Funcionais

Os requisitos funcionais descrevem as funcionalidades que o usuário deve ser capaz de acessar no sistema (SOMMERVILLE, 2011). Esses requisitos estão relacionados às atividades que o sistema realiza, sendo estes: cadastro de cursos, cadastro de turmas, cadastro de professores, cadastro de salas, cadastro de horário das aulas, cadastro de períodos letivos, gerar horário automático, montar horário manualmente, visualizar horário de professores e visualizar relação de professores por disciplina.

[REQ-F01] Cadastro de Cursos

O sistema deve permitir o cadastro dos cursos com a possibilidade de inclusão, consulta, alteração e exclusão dos atributos: Código, Nome, Tipo, Departamento no qual o curso pertence.

[REQ-F02] Cadastro de Turmas

O sistema deve permitir o cadastro de turmas com a possibilidade de inclusão, alteração e exclusão dos atributos: Nome da Turma, Ano Ingresso na Grade, Período, Semestre e Turno. Para inserir uma nova turma é necessário escolher um curso e seu projeto pedagógico.

[REQ-F03] Cadastro de Professores

O sistema deve permitir o cadastro dos dados pessoais dos professores com a possibilidade de inclusão, consulta, alteração e exclusão dos atributos nome (dados gerais) e a disponibilidade (Horários em que o professor se encontra disponível, nos turnos Matutino, Vespertino e Noturno).

[REQ-F04] Cadastro de salas

O sistema deve permitir o cadastro de salas onde as disciplinas serão ministradas com a possibilidade de inclusão, alteração e exclusão dos atributos: Numero ou nome da sala, localização, prédio a qual pertence.

[REQ-F05] Cadastro de Horário das aulas

O sistema deve permitir o cadastro de turmas com a possibilidade de inclusão, alteração e exclusão dos atributos: Disciplina que será ofertada horário de início, horário de término, sala na qual a disciplina será ministrada.

[REQ-F06] Cadastro de Períodos Letivos

O sistema deve permitir o cadastro dos dados relativos aos períodos letivos com a possibilidade de inclusão, consulta, alteração e exclusão dos atributos: código, descrição, início e término.

[REQ-F07] Gerar Horário Automático

O sistema deve permitir a geração de uma possibilidade de grade de horários de forma automática utilizando o otimizador heurístico integrado à solução.

[REQ-F08] Montar horário manualmente

O sistema dede permitir ao usuário montar o horário de forma Manuel, permitindo a verificação das ofertas de cada curso, o local onde ocorrerá as aulas e o professor responsável pela disciplina. Caso ocorra um conflito de horário o sistema deve informar o mesmo ao usuário, seja de professor ministrando duas disciplinas ao mesmo tempo ou uma sala com duas disciplinas ao mesmo tempo. O alerta do conflito de horários é muito importante, é a partir dele que se realizam as alterações e ajustes até se encontrar uma solução de grade de horário favorável aos interessados.

[REQ-F09] Visualizar Horário De Professores

O Sistema deve permitir a visualização dos Horários dos Professores. A visualização pode ser individual (por professor) ou de todos os professores simultaneamente de acordo com o período, e tuno desejado. Os turnos são matutino, vespertino e noturno.

[REQ-F10] Visualizar Relação de Professores por Disciplina

O Sistema deve permitir a visualização da relação entre professores e disciplinas, ou seja quais professores estão lecionando determinada disciplina.

4.1.2 Requisitos não-funcionais

Os requisitos não-funcionais são aqueles que não se relacionam diretamente com as funções do sistema; no entanto, são restrições impostas aos serviços oferecidos pelo sistema (SOMMERVILLE, 2011). A Tabela 1 apresenta a relação destes requisitos.

Tabela 1 - Requisitos não funcionais

Nome	Restrição	Categoria	Desejável	Permanente
NF1 Controle de Acesso	A função só pode ser acessada por usuário com perfil de funcionário ou professor.	Segurança	**( )**	**(x)**
NF2 Consultas	A função poderá ser acessada por qualquer usuário cadastrado (funcionários, professores, coordenadores)	Segurança	**( )**	**(x)**
NF3 Controle de Cadastro	A função permitirá que somente usuário com perfil de professor ou coordenador possa realizar inclusão, alteração e exclusão de dados.	Segurança	**(x)**	**(x)**

4.2 Diagrama de Casos de Uso

O diagrama de caso de uso é um modelo para representação das funcionalidades observáveis do sistema do ponto de vista do usuário, sem ser necessário especificar como estas funcionalidades serão implementadas.

A ferramenta Star UML foi utilizada para modelagem do diagrama de caso de uso apresentado no Diagrama 1.

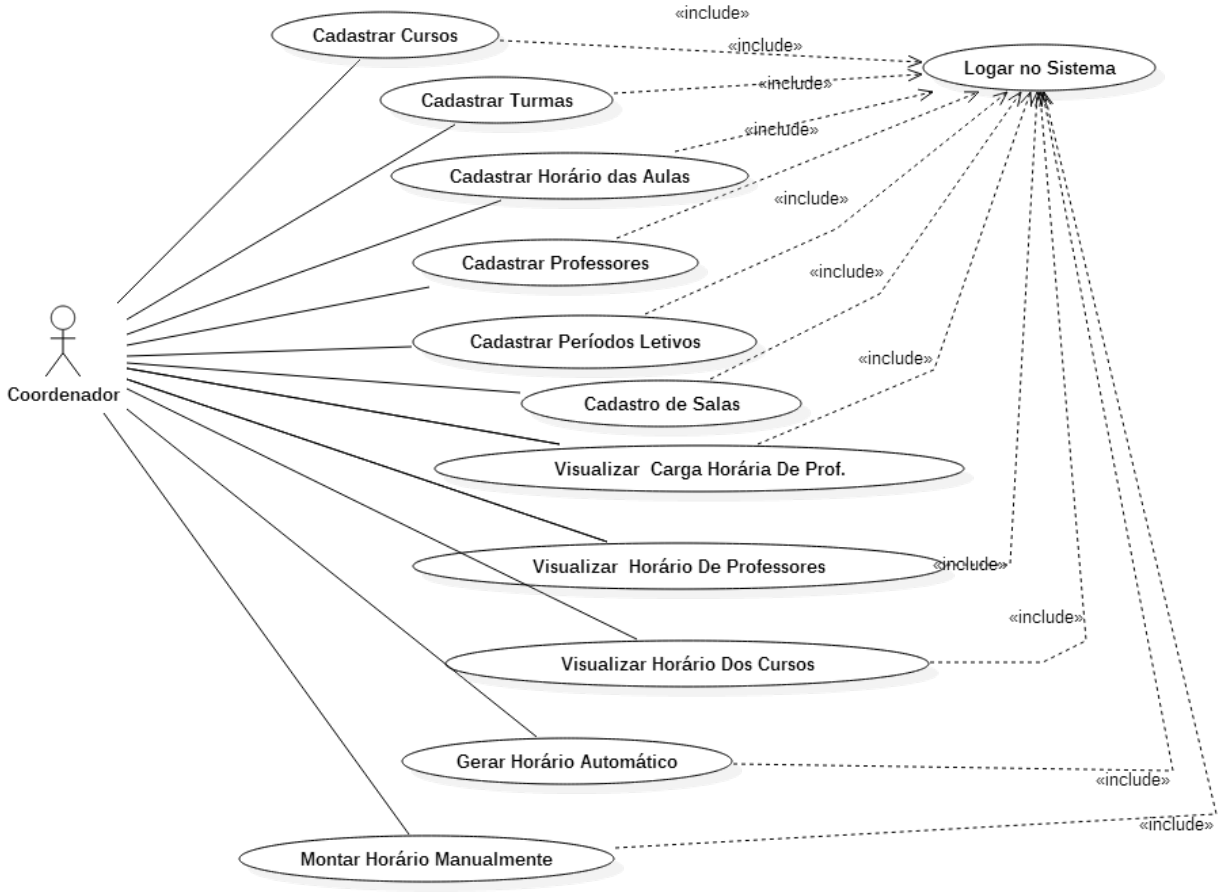

Diagrama 1 - Diagrama de Caso de Uso. Fonte: O autor.

UC01 – Logar no Sistema

Por meio da funcionalidade “fazer login”, o usuário tem acesso ao sistema a partir da identificação com informações relativas ao cadastro e senha realizados anteriormente, a fim de que mesmo tenha acesso às funcionalidades do sistema.

UC02 – Cadastrar Cursos

O cadastro de Cursos permite fazer o armazenamento de informações de todos os cursos pertencentes à uma instituição. Além do cadastro o usuário poderá recuperar e editar estas informações.

UC02 – Cadastrar Turmas

Por meio do cadastro de turmas o usuário poderá realizar o armazenamento de informações das turmas. Além do cadastro o usuário poderá recuperar e editar estas informações.

UC03 – Cadastrar Horário das Aulas

O cadastro de professores permite realizar o armazenamento de informações dos horários em que cada aula será ministrada. Além do cadastro o usuário poderá recuperar e editar estas informações.

UC04 – Cadastrar Professores

O cadastro de professores permite realizar o armazenamento de informações dos professores. Além do cadastro o usuário poderá recuperar e editar estas informações.

UC05 – Cadastrar Períodos Letivos

O cadastro de períodos letivos permite fazer o armazenamento de informações dos períodos letivos. Além do cadastro o usuário poderá recuperar e editar estas informações.

UC06 – Cadastro de Salas

Por meio do cadastro de salas o usuário poderá fazer o armazenamento dos locais onde serão ministradas as aulas. Além do cadastro o usuário poderá recuperar e editar estas informações.

UC07 – Visualizar Carga Horária de Professor

Por meio dessa funcionalidade o usuário pode selecionar um professor com a intenção de visualizar sua carga horária total por semestre.

UC08 – Visualizar Horário de Professores

Esta funcionalidade permite ao usuário visualizar a grade de horários dos professores separados por turnos e por dias da semana. Essa funcionalidade permite a visualização de forma individual para cada professor.

UC09 – Visualizar Horário dos Cursos

Esta funcionalidade permite ao usuário visualizar a grade de horários para cada período de cada curso.

UC10 – Gerar Horário Automático

Esta funcionalidade permite ao usuário gerar uma possibilidade de grade de horários, por meio de uma heurística que foi integrada ao projeto.

UC11 – Montar Horário Manualmente

Esse caso de uso especifica a ação de cadastro e confecção manual da grade de horários. Sua principal funcionalidade é a possibilidade de atribuição de uma disciplina a

um professor em determinado horário. Este caso de uso comtempla e recuperação e edição desses dados.

4.3 Diagrama Entidade Relacionamento (MER)

Todas as informações utilizadas pelo sistema são lidas de um banco de dados relacional. O Sistema Gerenciador de Banco de Dados (SGBD) utilizado para implementação dessa solução é o MySQL.

O modelo entidade relacionamento (MER) é a representação logica de como o banco de dados está disposto para um sistema. Para a geração do Diagrama 2, foi utilizado o MySQL Workbench, uma ferramenta para criação e administração de banco de dados MySQL.

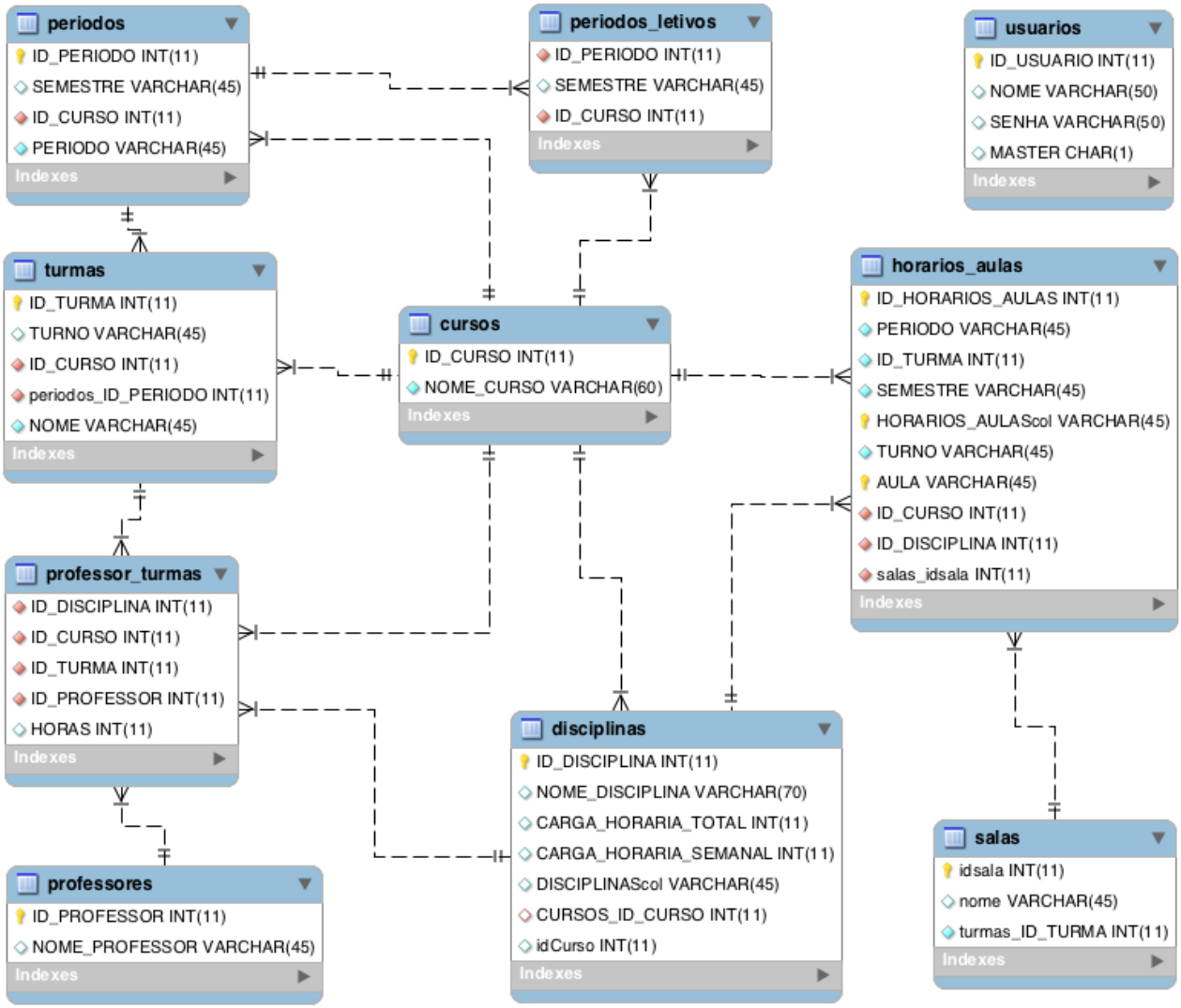

Diagrama 2 - Modelo Entidade Relacionamento. Fonte: O autor.

O Diagrama 2 fornece uma representação visual do banco de dados, onde as principais tabelas do banco são denominadas períodos, cursos, professores, disciplinas, turmas, salas e horários_aulas.

A tabela Períodos representa a entidade Período, que é composta por quatro atributos: um identificador único (id_periodo), período, semestre e é relacionada com a tabela cursos pela chave id_curso. A tabela Cursos representa a entidade Curso e é composta por dois atributos: um identificador único e nome.

A tabela professores representa a entidade Professor, que é composta por dois atributos: um identificador único (id_professor) e o nome (nome_professor).

A tabela disciplinas representa a entidade Disciplina, que tem como principais atributos: identificador único (id_disciplina), nome (nome_disciplina), a carga horária do semestre (carga horária total) e a quais cursos a disciplinas está relacionada (idCurso).

A tabela turmas representa a entidade Turma e caracteriza-se por cinco atributos: identificador único (id_turma), nome, turno, a quais cursos uma turma está relacionada (idCurso) e a qual período uma turma está relacionada (períodos_id_periodo).

Já a tabela horários_aulas é concebida por meio da combinação de diversos elementos e com os identificadores únicos relacionando as entidades disciplinas, cursos e salas. A tabela salas representa o espaço físico onde ocorrerão as aulas e possui dois atributos: um identificador único (idsala), nome e a relação com turmas (turmas_id_turma).

4.4 Telas

Nesta seção serão apresentadas as principais telas do sistema, a descrição de seu funcionamento e utilização.

4.4.1 Tela de Login

Antes do acesso as das funcionalidades do sistema, o usuário deverá se autenticar informando seu usuário e senha, para que seja validado o acesso. Essa autenticação é feita através da tela de login, como mostra a Figura 4.

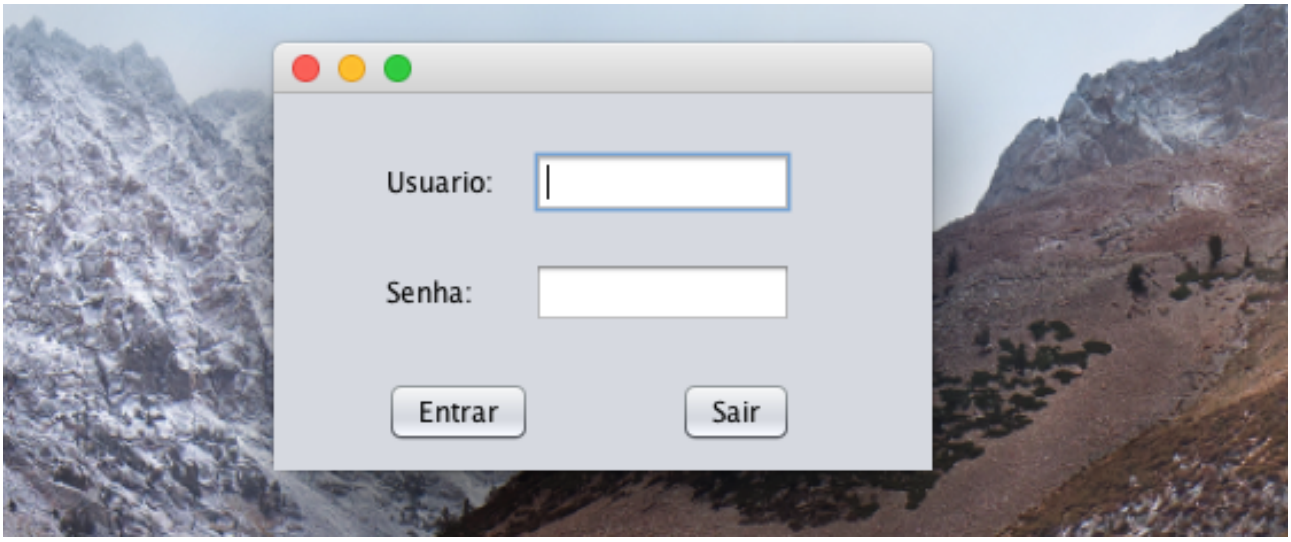

Figura 4 - Tela de Login. Fonte: O autor

4.4.2 Dashboard

O acesso ao sistema será liberado à usuários previamente cadastrados no banco de dados. Esse cadastro é realizado diretamente no banco e é feito pelo administrador ou responsável pela implantação do sistema.

Em caso de erro na autenticação é exibida uma mensagem de erro ao usuário informando que o acesso foi negado e o motivo, conforme mostrado na Figura 5.

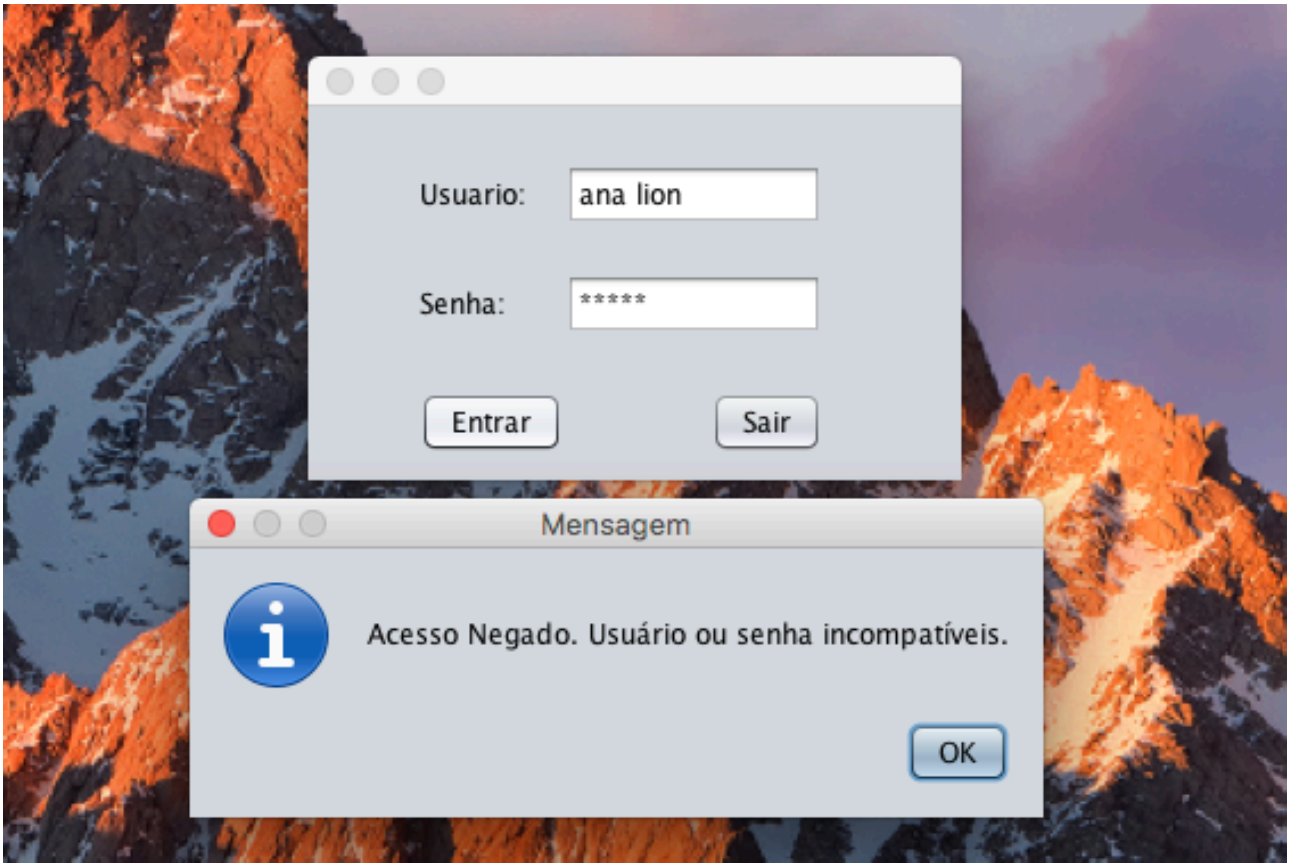

Figura 5 - Mensagem de erro de login. Fonte: O autor

Em caso de sucesso, o usuário é direcionado para a tela principal do sistema, que é um dashboard. Nos conceitos de interface com o usuário, um dashboard é a área da aplicação utilizada para reunir informações sobre diversas partes do sistema em um único lugar, facilitando sua visualização de acordo com Leon (2013).

Neste dashboard se encontra o menu que disponibiliza o acesso as demais telas do sistema, conforme destacado em vermelho na Figura 6.

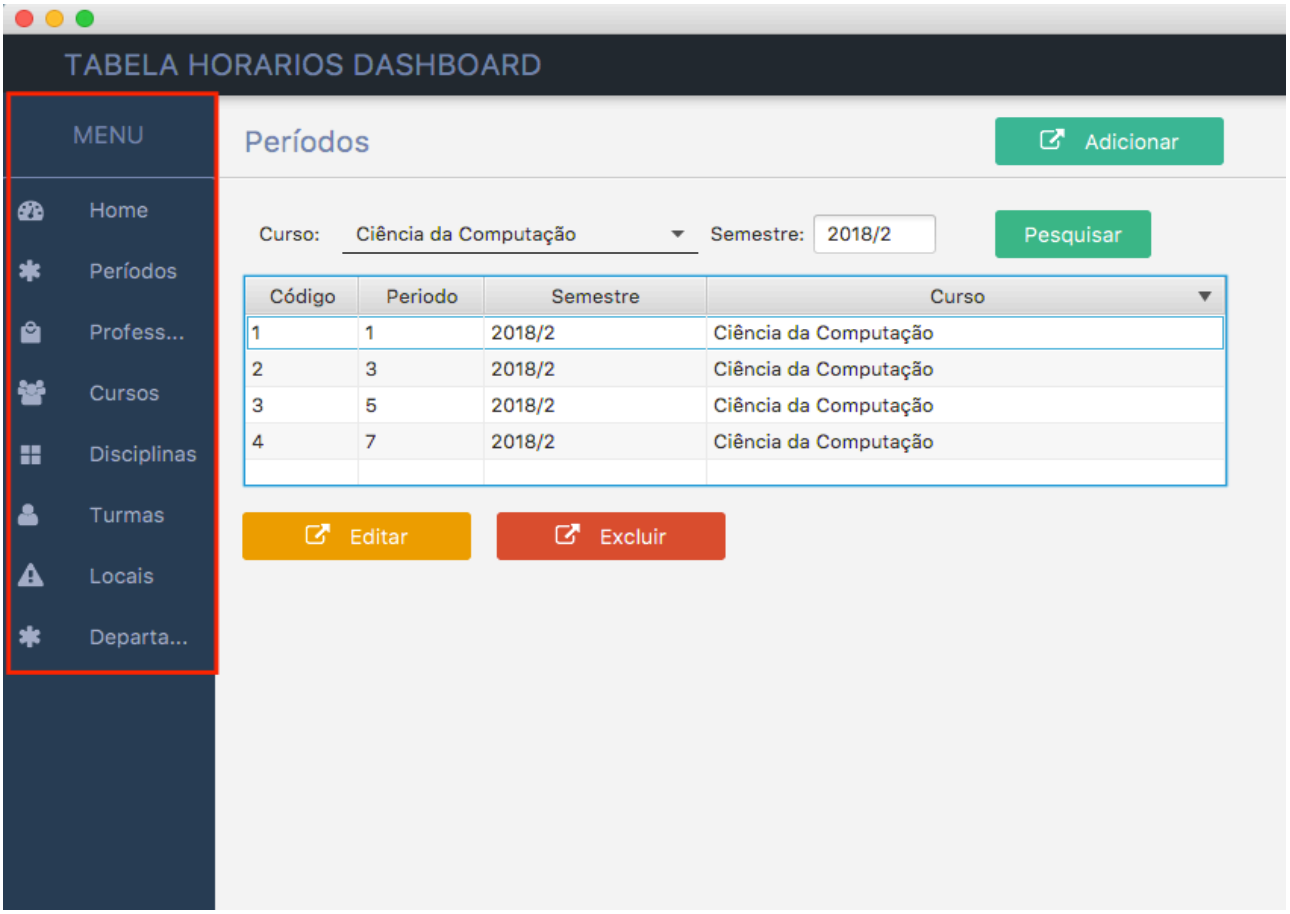

Figura 6 – Dashboard. Fonte: O autor.

4.4.3 Períodos

Ao selecionar o menu Períodos o usuário tem acesso a lista de períodos cadastrados com a possibilidade de adicionar um novo, editar ou excluir um já existente. Ao adicionar um novo é aberta a tela de cadastro com os campos Período, Semestre e Curso disponíveis para preenchimento conforme mostrado na Figura 7. Ao editar um período já cadastrado a mesma tela é aberta, porém com os campos preenchidos de acordo com o registro selecionado na listagem.

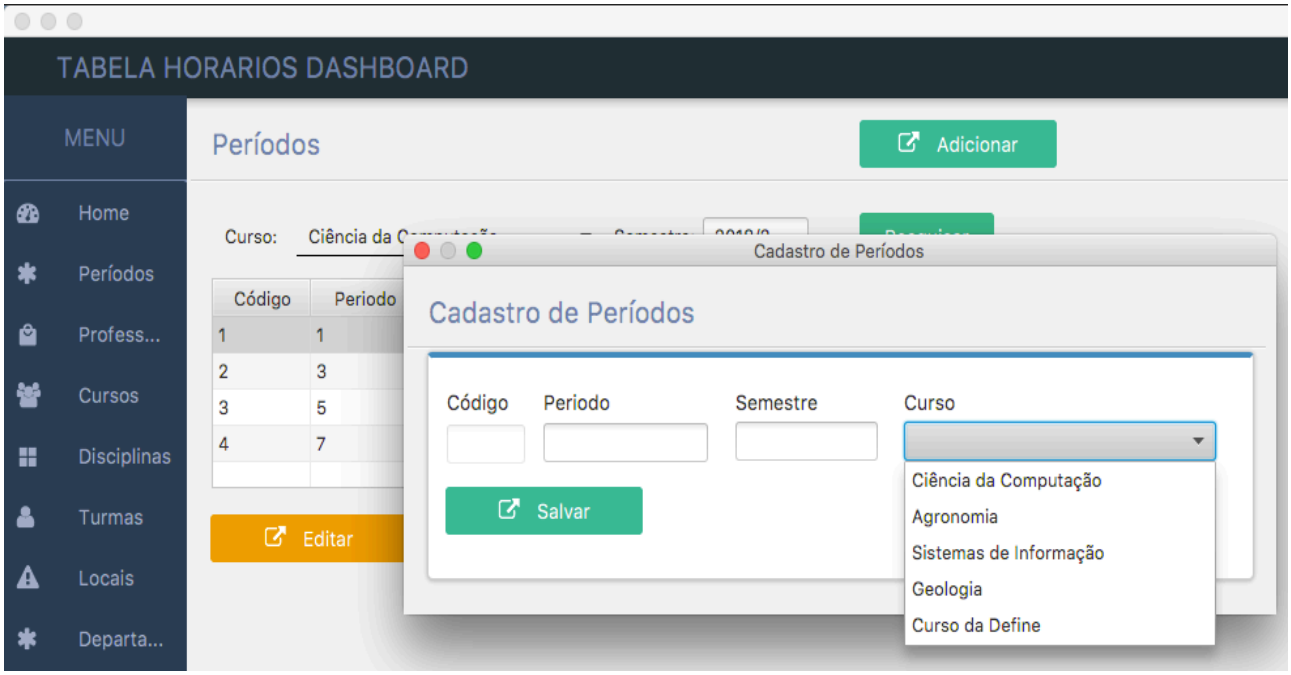

Figura 7 - Tela de Cadastro de Períodos. Fonte: O autor.

4.4.4 Cursos

O menu Cursos do dashboard disponibiliza a tela de listagem dos cursos cadastrados com a possibilidade de adicionar, editar e excluir, Figura 8.

Ao clicar no botão Adicionar, é aberta uma nova tela para realizar o cadastro de um registro, com os campos nome e departamento habilitados para preenchimento, conforme Figura 9. Para a opção de edição a mesma tela de cadastro é aberta, porém com os campos já preenchidos de acordo com o registro selecionado na listagem.

Figura 8 - Listagem de cursos. Fonte: O autor.

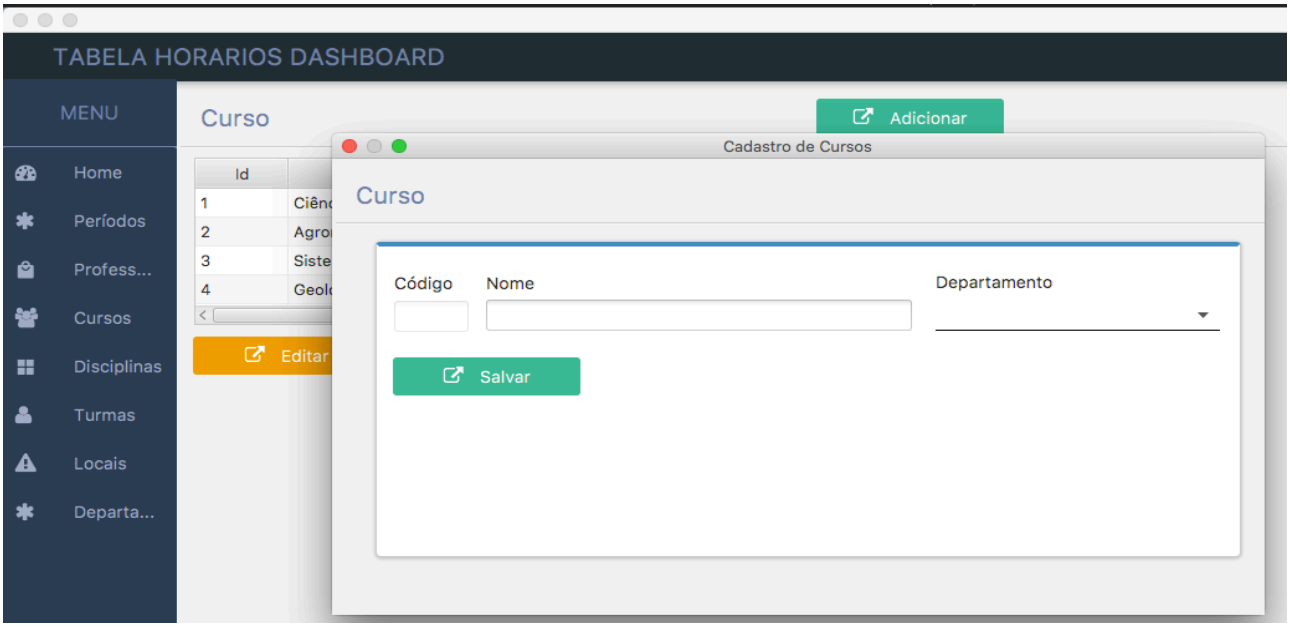

Figura 9 - Cadastro de curso. Fonte: O autor.

4.4.5 Disciplinas

Na opção Disciplinas do menu, o usuário tem acesso a tela de listagem de disciplinas cadastradas com a possibilidade de adicionar, editar e excluir os registros. Além disso, diferente das telas de listagem apresentadas anteriormente, esta oferece a possibilidade de realizar a pesquisa filtrando por um determinado curso, conforme apresentado na Figura 10. Caso não seja escolhida nenhuma opção no filtro todas as disciplinas cadastradas serão listadas.

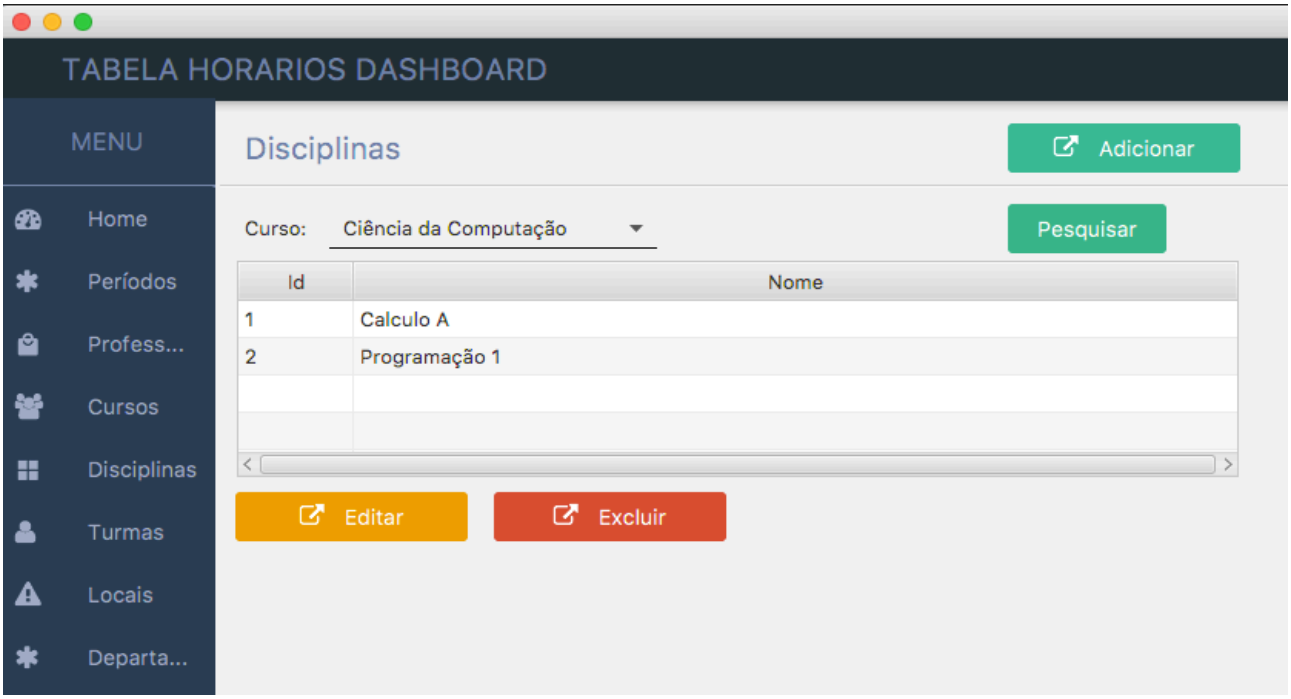

Figura 10 - Listagem de disciplinas. Fonte: O autor.

Ao clicar no botão Adicionar, de acordo com a Figura 11, é aberta uma tela para realizar o cadastro de uma nova disciplina, com os campos nome, carga horária semanal, carga horária total e o curso a qual pertence.

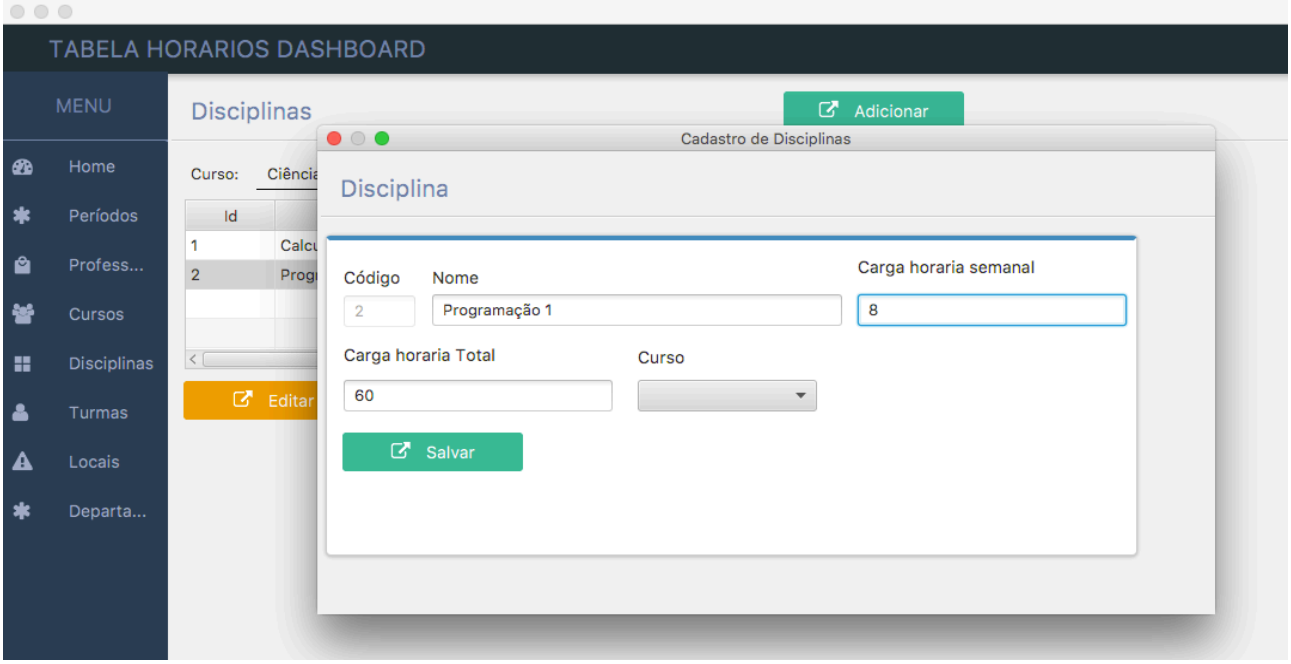

Figura 11 - Cadastro de disciplina. Fonte: O autor.

4.4.6 Locais

A opção Locais do menu possibilita o acesso a tela de listagem de locais/salas onde as aulas são ministradas com a possibilidade de adicionar, editar e excluir os registros listados, conforme Figura 12.

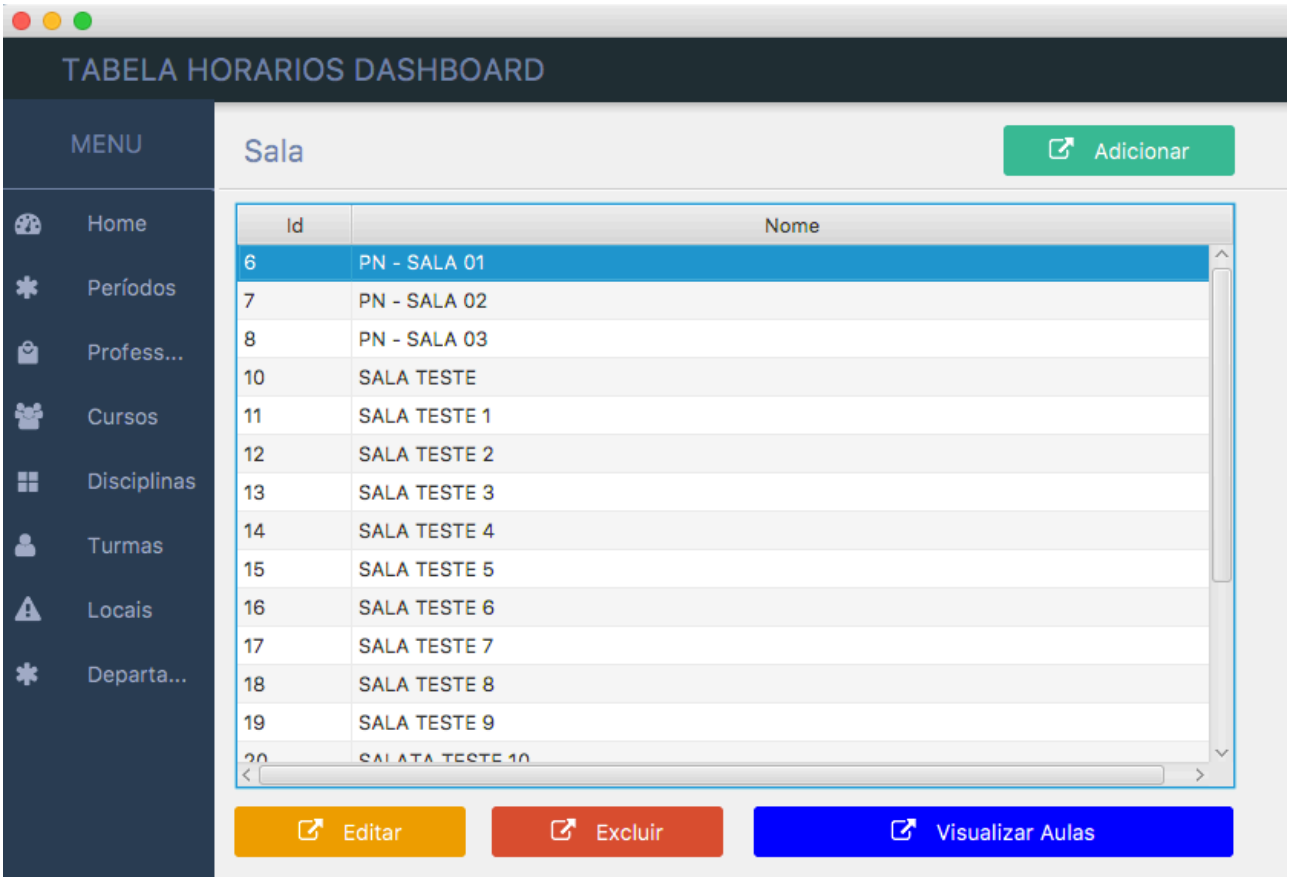

Figura 12 - Listagem de Locais. Fonte: O autor.

Ao clicar no botão adicionar, é aberta uma tela para realizar o cadastro de um novo local, com o campo nome disponível para preenchimento do usuário. Ao editar, a mesma tela de cadastro é aberta, porém com os campos código e nome preenchidos com os valores do registro selecionado da tela de listagem.

O botão visualizar aulas possibilita ao usuário visualizar a grade de horários de aulas que serão ministradas em determinado local. A Figura 13 exemplifica os horários de todas as aulas que serão ministradas no local PN – SALA 3 de acordo com os dias da semana. Essa tela é somente para visualização não permitindo edição.

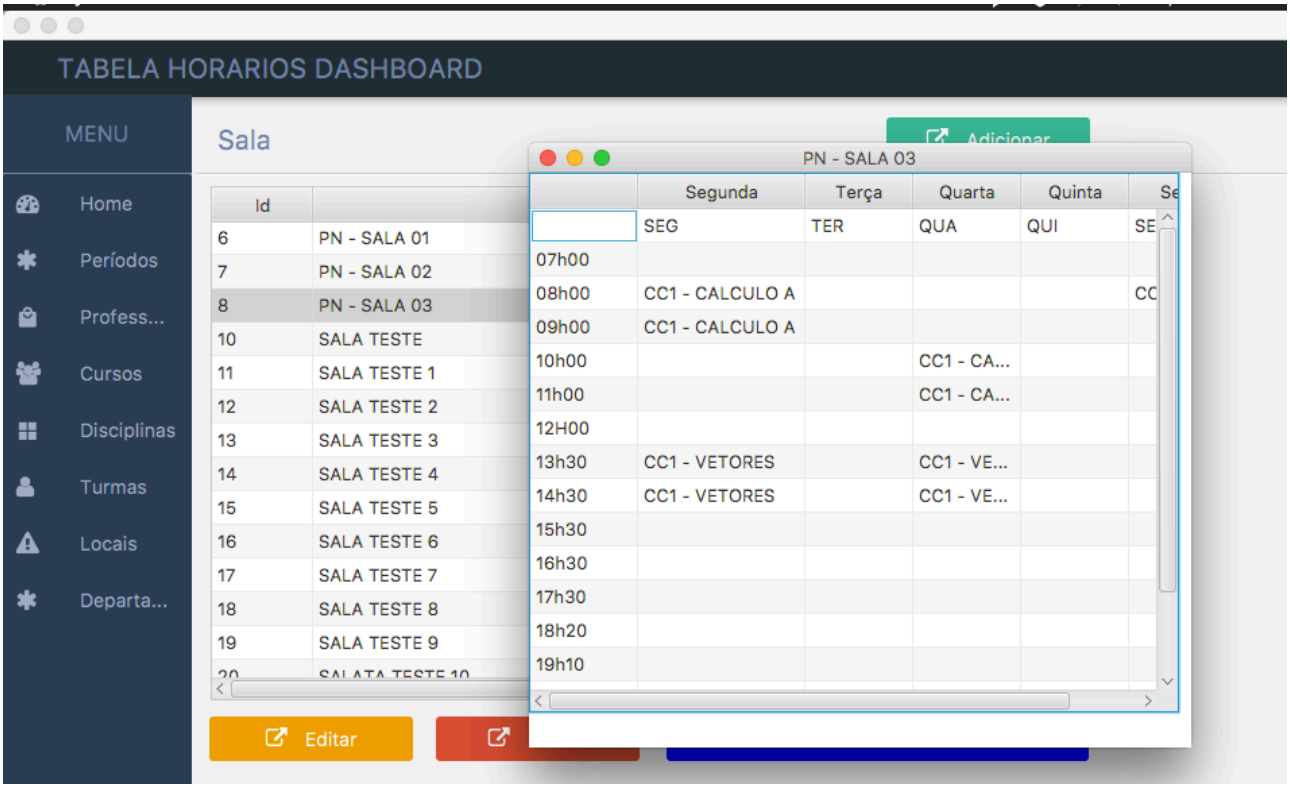

Figura 13 - Horários de aula por sala. Fonte: O autor.

4.4.7 Professores

A opção Professores do menu possibilita o acesso a tela de Listagem de professores com a possibilidade de adicionar, editar e excluir os registros listados, conforme Figura 14.

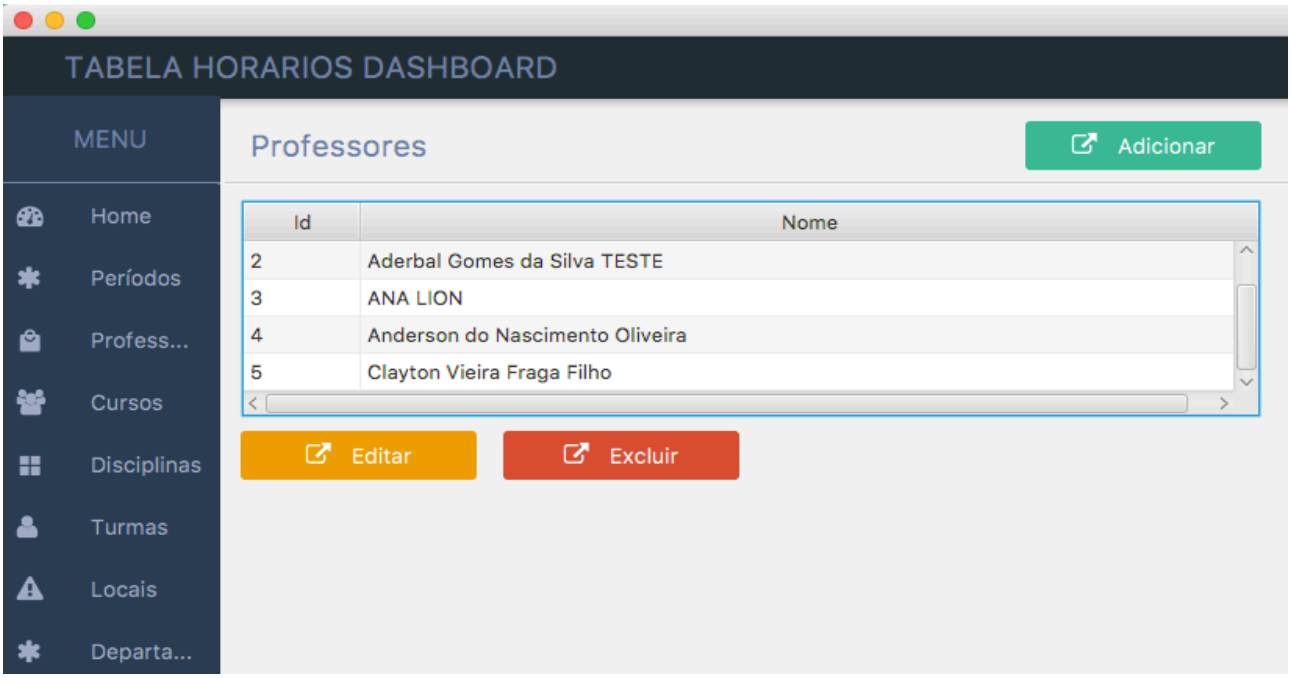

Figura 14 - Listagem de Professores. Fonte: O autor.

O botão adicionar abre a tela cadastro de professores com o campo nome disponível para preenchimento. Ao editar um professor previamente cadastrado é aberta a mesma tela de cadastro, porém agora são exibidos os quadros de horários do professor em questão, separados por turnos; matutino, vespertino e noturno como pode ser observado na Figura 15.

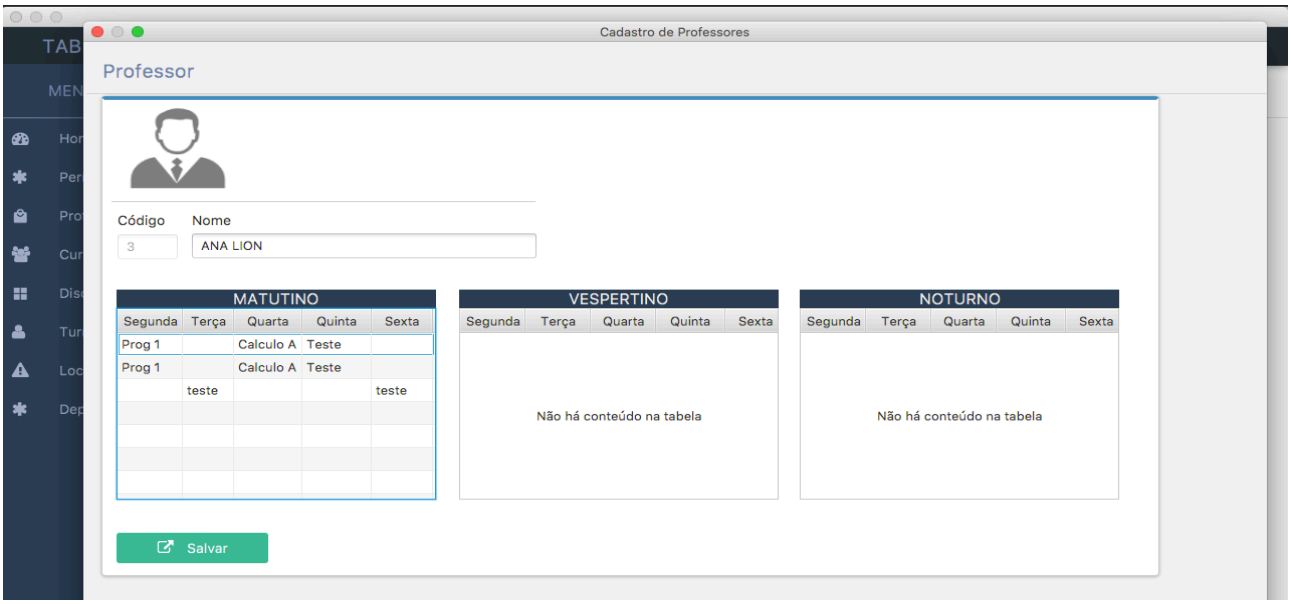

Figura 15 - Cadastro de Professor. Fonte: O autor.

Ao clicar em qualquer um dos quadros é aberta uma nova tela com a mesma informação dos horários, porém, agora, todos agrupados e com a informação adicional da hora conforme apresentado na Figura 16. Essa tela além de exibir, possibilita a edição e ajuste dos horários. Por exemplo, para reajustar o horário do professor Clayton cadastrado no dia de quinta-feira às 13:30 horas, basta clicar na célula, arrastar e soltar sobre a nova alocação de horário. Ao clicar no botão salvar desta tela as alterações são persistidas no banco de dados.

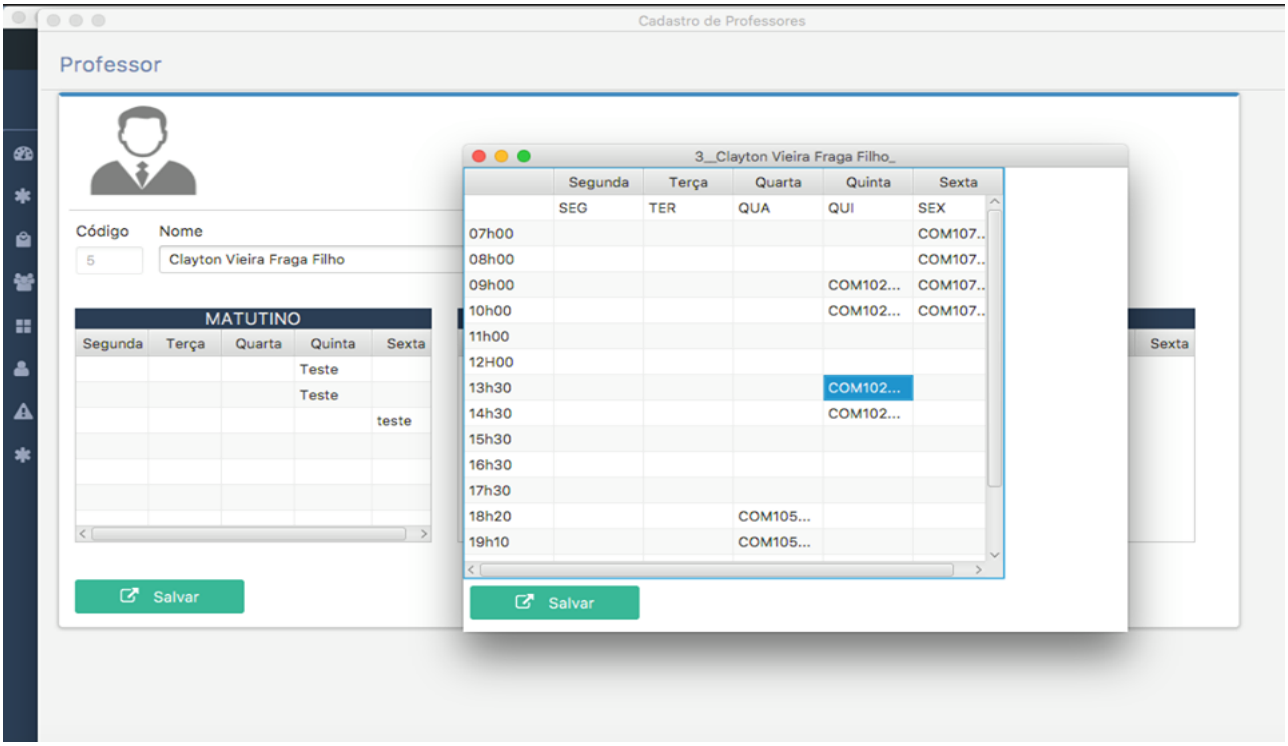

Figura 16 - Tela de ajuste de horário. Fonte: O autor.

4.4.8 Turmas

A opção turmas do menu possibilita o aceso a tela de listagem, cadastro, edição e exclusão de turmas, e se difere das demais telas do sistema por estar dividida em três abas: turmas, disciplinas/professores/salas e horário, conforme pode ser observado na Figura 17.

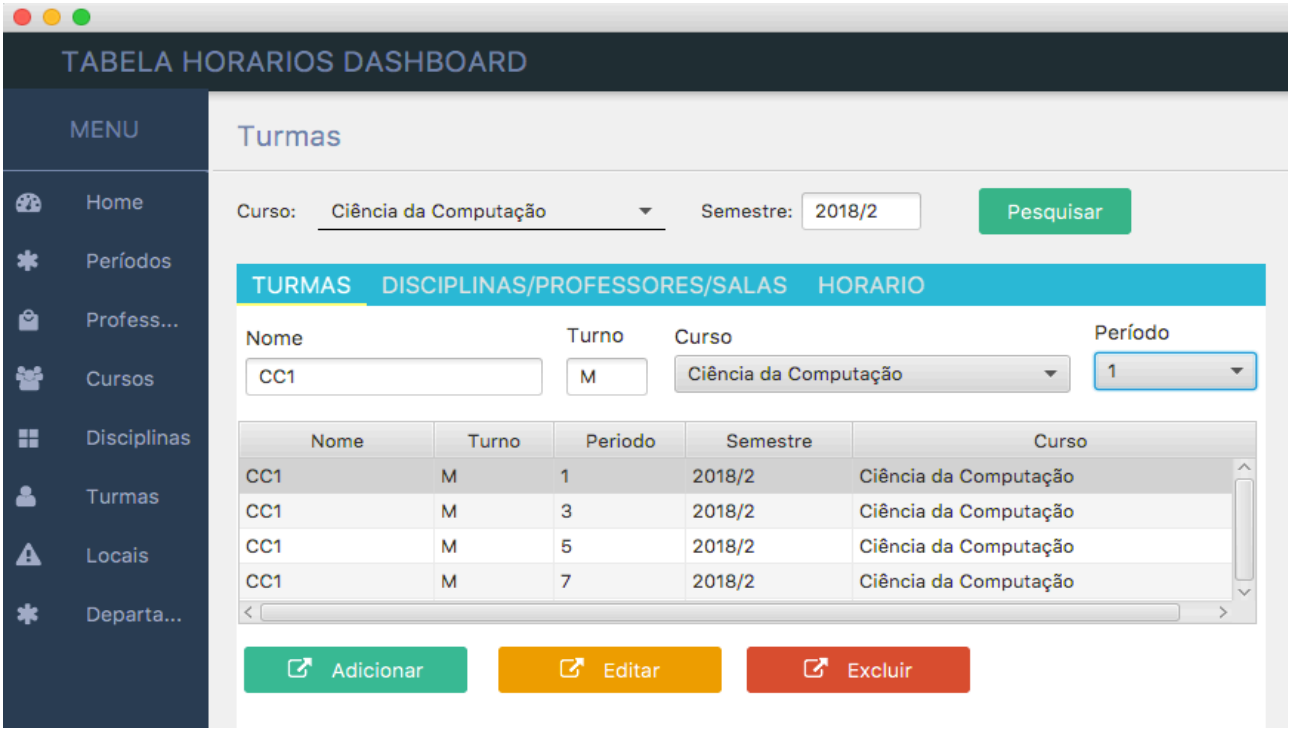

Figura 17 - Cadastro de turmas. Aba Turmas. Fonte: O Autor.

Nesta tela, o usuário pode realizar a pesquisa de turmas filtrando por curso e semestre, para facilitar a separação e visualização das turmas listadas. Na aba Turmas é possível cadastrar, editar e excluir uma turma. Diferente das demais telas exibidas até o momento, nesta, ao cadastrar ou editar um registro os campos localizados acima da tabela de listagem são utilizados e não é aberta uma nova tela para estes processos. Os campos curso e período trazem as informações das entidades já cadastradas no sistema, ou seja não são campos livres inseridos pelo usuário.

A aba Disciplinas/Professores/Salas mostrada na Figura 18, possibilita relacionar professores com disciplinas e salas de aula. As informações desta aba dependem da turma selecionada na tabela da aba anterior em um conceito conhecido como master-detail. Este conceito representa um relacionamento um para muitos entre duas relações de dados e seu objetivo é simplificar a visualização do relacionamento entre eles, de acordo com RODRIGUES (2012).

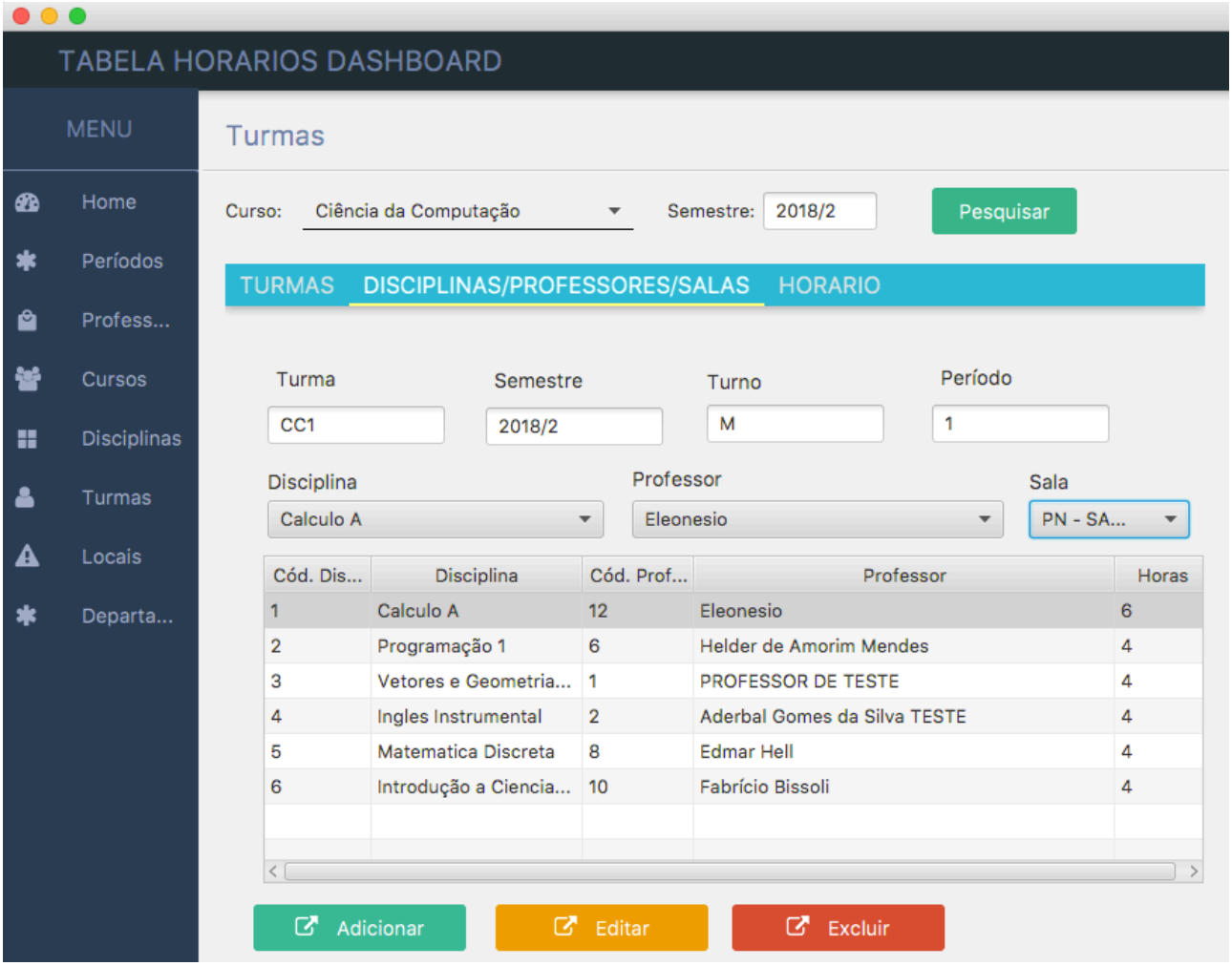

Figura 18 - Aba Disciplinas/Professores/Salas. Fonte: O autor.

Os campos turma, semestre, turno e período são apenas de visualização, funcionando como um cabeçalho e não podem ser editados. Já os campos disciplina, professor e sala são editáveis e o usuário pode relacionar as opções disponíveis já cadastradas no sistema.

Após o relacionamento de professores, disciplinas e salas o usuário pode visualizar a grade de horários na aba Horários exemplificado na Figura 19. Os horários são exibidos por dias de semana de segunda-feira a sexta-feira e são consideradas aulas de duas horas de duração. A cada célula correspondente a uma aula no quadro de horários são exibidas as informações de professor, código e nome de disciplina e qual a sala em que ocorrerá a aula.

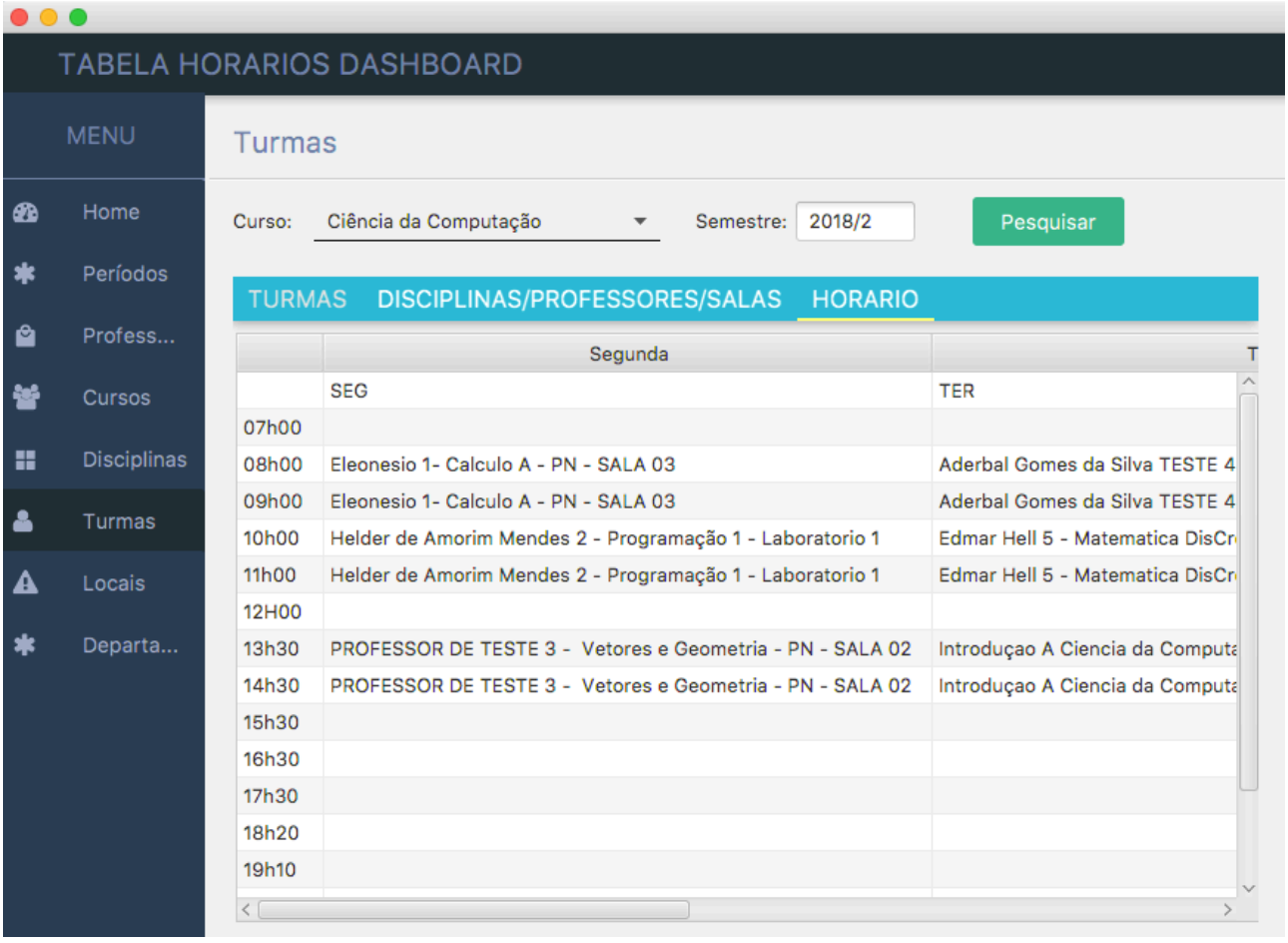

Figura 19 - Aba Horários. Fonte: O autor.

4.4.9 Home

Por fim é apresentada a tela Home. Esta tela apesar de ser bastante simples possui uma funcionalidade essencial para o funcionamento do sistema. Ao pressionar o botão “Gerar Grade de Horários”, conforme apresentado na Figura 20, o sistema reúne as

informações de períodos, disciplinas, cursos e professores cadastradas nas telas anteriores e executa chamada ao método heurístico desenvolvido por Soares (2015) fornecendo as entradas necessárias para a execução da heurística.

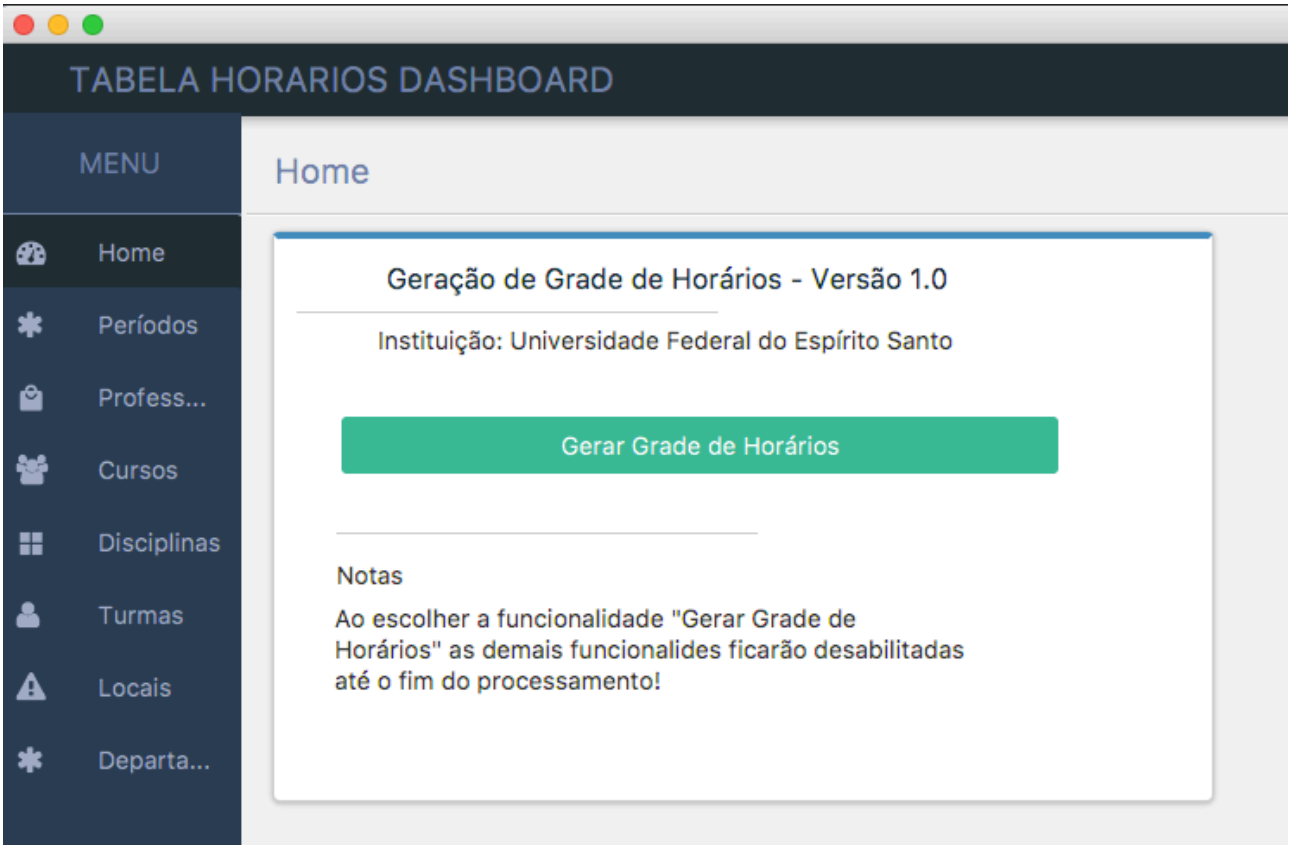

Figura 20 - Tela home. Fonte: O autor.

Ao final do processamento, o método gera saídas em formato de planilhas do excel (.xlsx) com os horários propostos separados por professores, salas e turmas. O sistema por sua vez percorre as planilhas geradas e persiste as informações no banco de dados o que alimenta as demais telas do sistema.

5 CONCLUSÃO

Atualmente a confecção da grade de horários em instituições de ensino, com grande número de cursos e disciplinas, é uma tarefa complexa e que pode demandar muito tempo para ser realizada quando feita de forma manual, como é o caso do Centro de Ciências Exatas Naturais e da Saúde da Universidade Federal do Espírito Santo (CCENS – UFES). É importante destacar que quando confeccionada de forma manual, falhas humanas podem ocorrer, como por exemplo, aulas alocadas em uma mesma sala ao mesmo tempo, causando confusão e inconsistências.

Mediante as dificuldades encontradas e relatadas pelos coordenadores de curso e com base em trabalhos anteriores, este trabalho apresentou o desenvolvimento de um sistema para automação da elaboração dos quadros de horários, utilizando ferramentas e *frameworks open-source*, tomando como base a realidade do CCENS – UFES, afim de otimizar o tempo e os recursos envolvidos neste processo.

Em sua maioria os sistemas encontrados no mercado, que tratam grades de horário, são voltados para a educação básica e não podem ser aplicados ao ensino superior, com exceção alguns que tratam o problema de forma genérica. Devido a isso existe pouco referencial teórico para este trabalho.

A utilização da solução proposta ameniza o esforço, reduz o tempo necessário para confecção da grade e diminui a suscetibilidade a erros, uma vez que o usuário não necessita de vários recursos e tecnologias para aplicar a solução.

Uma vez que os testes do sistema foram realizados apenas pelo autor e desenvolvedor deste trabalho, não havendo aplicação deste sistema na universidade, em um cenário real, não foi possível efetuar a comparação dos resultados atuais com as possibilidades que o sistema pode gerar, o que, por sua vez não desqualifica o projeto que é um protótipo em condições de testes em situações reais e com a possibilidade de abrangência à outras instituições.

6 REFERÊNCIAS

ABRAMSON, D. Constructing school timetables using simulated annealing: sequential and parallel algorithms. *Management Science*, v. 37, p. 98-113, 1991.

CARVALHO, A. S. Meta-heurísticas para resolução de um caso real de programação de horários. Trabalho de Conclusão de Curso (Sistemas de Informação) – Universidade Federal do Espírito Santo, Alegre, 2015.

BERARDINO, D. R. D. *Software* para Geração da Grade Horária de curso BSI da UNIRIO. Trabalho de Conclusão de Curso (Sistemas de Informação) – Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro, 2013.

CSIMA, J.; GOTLIEB, C. C. Tests on a computer method for construction of school timetables. *Communications of the ACM*, New York, v. 7, p.160-163, mar. 1964.

CHAVES, A. A. *Uma meta-heurística híbrida com busca por agrupamentos aplicada a problemas de otimização combinatória***.** 2009. 197 f. Tese (Doutorado) - Instituto Nacional de Pesquisas Espaciais, São José Dos Campos, 2009.

COLORNI, A.; DORIGO, M.; MANIEZZO, V. Metaheuristics for high school timetabling. *Computational Optimization and Applications*, v. 9, p. 275-298, 1998.

CRONOS. Solução em Horários Escolares. Disponível em <https://www.cronostimetable.com/> Acesso em 10 de outubo de 2018.

DIJKSTRA, Edsger Wybe. *Structured programming*. 1969.

DOOLEY, J. *Software Architecture. In: Software Development and Professional Practice. Apress, Berkeley*, CA. 2011

GRIDCLASS, Programa de Horário Escolar Online. Disponível em <http://gridclass.com.br/> Acesso em 10 de outubro de 2018.

JAVA, Obtenha Informações Sobre a Tecnologia Java. 2015. Disponível em <https://www.java.com/pt_BR/about/> Acesso em 22 de outubro de 2018.

LEON, M. Dashboard Democracy. Computerworld, junho, 16, 2003.

MARIANO, G. P. *Resolução do problema de programação de horários de disciplinas do CCA-UFES utilizando a meta-heurística ALNS*. Trabalho de Conclusão de Curso (Ciência da Computação) – Universidade Federal do Espírito Santo, Alegre, 2014.

ORACLE. Java FX. 2018. Disponível em: <https://www.oracle.com/technetwork/java/javase/overview/javafx-overview-2158620.html/> Acesso em 05 de outubro de 2018.

PRESSMAN, Roger S. Engenharia de *software*. 6ª ed. Porto Alegre: Bookman, 2006.

RAMOS, P. S. Sistema Automático de Geração de Horários para a UFLA utilizando Algoritmos Genéticos. Trabalho de Conclusão de Curso (Ciência da Computação) – Universidade Federal de Lavras.

RODRIGUES, J. GridView: Exibindo Registros no Formato Master-Detail. Disponível em <https://www.devmedia.com.br/asp-net-gridview-exibindo-registros-no-formato-master-detail/25454>. Acesso em 10 de outubro de 2018.

SANTOS, H. G.; Souza, M. J. F. Programação de horários em instituições educacionais: formulações e algoritmos. In: SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, 39., 2007, Fortaleza. *Anais...* Fortaleza: SBPO, 2007.

SAUVÉ, J. P. Model-view-controller (MVC). Notas de Aula. Disponível em <http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/arqu/mvc/mvc.htm/>. Acesso em 10 de outubro de 2018.

SOMMERVILLE, Ian. Engenharia de *software*. 8ª ed. São Paulo: Pearson Addison-Wesley, 2007.

SOUZA, M. J. F. *Programação de Horários em Escolas: Uma Aproximação por Metaheurísticas*. 2000. 160 f. Tese (Doutorado em Engenharia de Sistemas e Computação), Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2000.

YOURDON, E. Análise estruturada moderna. Rio de Janeiro : Campus, 1990.