

UNIVERSIDADE FEDERAL RURAL DO RIO DE JANEIRO
INSTITUTO MULTIDISCIPLINAR

KLEYTON PONTES COTTA

**Ferramenta de Detecção de Conflitos e
Gestão para o Timetabling Problem**

Prof. Filipe Braidão do Carmo, M.Sc.
Orientador

Rio de Janeiro, Dezembro de 2014

Ferramenta de Detecção de Conflitos e Gestão para o Timetabling Problem

Kleyton Pontes Cotta

Projeto Final de Curso submetido ao Departamento de Ciência da Computação do Instituto Multidisciplinar da Universidade Federal Rural do Rio de Janeiro como parte dos requisitos necessários para obtenção do grau de Bacharel em Informática.

Apresentado por:

Kleyton Pontes Cotta

Aprovado por:

Prof. Filipe Braida do Carmo, M.Sc.

Prof. Leandro Guimaraes Marques Alvim, D.Sc.

Prof. Carlos Eduardo Ribeiro de Mello, D.Sc.

Prof. Fellipe Ribeiro Duarte, M.Sc.

RIO DE JANEIRO, RJ - BRASIL

Dezembro de 2014

Agradecimentos

Quero agradecer aos meus pais, Sebastião Cotta e Maria do Rosário por toda educação que me ofereceram e pelo apoio em todos os momentos da minha vida. A minha irmã Kelly que me deu forças para continuar meus estudos, juntamente com minha tia Maria do Carmo, que sem seu apoio e sua bondade não conseguiria superar os obstáculos.

Agradecimento especial ao meu filho Nicolas, que é minha inspiração para seguir em frente e superar os desafios impostos pela vida, dedico todas as conquistas a esse garoto que eu amo.

Agradeço também a todos os professores que me acompanharam durante a graduação, em especial ao Prof. Filipe Braidá, por ter acreditado e foi responsável pela realização deste trabalho.

Aos meus colegas de faculdade e do trabalho, que não se negaram a ajudar quando solicitados, em especial, Hugo, Júlio e Raul, que contribuíram muito para esse momento.

E por fim, agradecer todos meus amigos e familiares que incentivaram e que souberam entender as minhas ausências.

RESUMO

Ferramenta de Detecção de Conflitos e Gestão para o Timetabling Problem

Kleyton Pontes Cotta

Dezembro/2014

Orientador: Filipe Braida do Carmo, M.Sc.

O Timetabling Problem devido a sua natureza combinatória é caracterizado como sendo de alta complexidade computacional. Consiste em alocar recursos sujeitos a restrições, e tais alocações devem acontecer para um período finito de tempo. Atualmente não existe uma ferramenta interativa boa para visualização das informações e principalmente, que possa detectar os conflitos dinamicamente. O trabalho tem como objetivo desenvolver uma ferramenta de detecção de conflitos para ser utilizada como apoio na alocação de recursos e na gestão de todos os processos que envolvem a criação de uma grade de horário. A solução possui interface web de modo interativo, no qual levam em conta os princípios de usabilidade, a visualização de dados com várias dimensões e a interação homem-máquina.

Palavras-chave: Timetabling, Detecção de Conflitos, Grade de horário, Visualização da Informação.

ABSTRACT

Ferramenta de Detecção de Conflitos e Gestão para o Timetabling Problem

Kleyton Pontes Cotta

Dezembro/2014

Advisor: Filipe Braida do Carmo, M.Sc.

The Timetabling Problem due to its combinatorial nature is characterized as high computational complexity. Is to allocate resources subject to restrictions, and such allocations must happen for a finite period of time. Currently there is no good interactive tool for information display and mainly that can detect conflicts dynamically. The study aims to develop a conflict detection tool to be used as support in the allocation of resources and the management of all processes involving the creation of a time grid. The solution has interactive mode of web interface, which take into account the principles of usability, data visualization with various dimensions and the man-machine interaction.

Keywords: *Timetabling, Conflict Detection, Schedule, Information Visualization.*

Lista de Figuras

3.1	Diagrama de Atores.	14
3.2	Diagrama de Caso de Uso.	24
3.3	Caso de uso inserir turma.	25
3.4	Caso de uso alterar turma.	26
3.5	Caso de uso criar grade.	27
3.6	Caso de uso carregar grade.	27
3.7	Diagrama de Classe.	30
4.1	Tela de apresentação do sistema.	36
4.2	Menu da parte interna do sistema.	36
4.3	Janela com as opções de cursos.	36
4.4	Selecionador de grade de horário.	37
4.5	Modos de visualização separado por abas.	37
4.6	Grade de horário para alocação de tempos.	38
4.7	Modos de visualizações.	38
4.8	Campo para especificar o período vigente da grade.	39
4.9	Box de ofertas de disciplinas.	39
4.10	Campo para adicionar ofertas de disciplinas.	39

4.11	Legenda das ofertas de disciplinas.	40
4.12	Grade com horários corretos.	41
4.13	Ofertas com horário fixo ativado.	41
4.14	Ofertas incompletas na grade.	41
4.15	Representação dos conflitos nas ofertas.	42
4.16	Grade da sala com conflitos.	42
4.17	Representação das ofertas no box.	43
4.18	Editar oferta de disciplina.	44

Lista de Tabelas

3.1	Relação dos requisitos de usuário.	14
3.2	Requisito criar grade.	16
3.3	Requisito carregar grade.	17
3.4	Requisito salvar grade.	17
3.5	Requisito inserir turma.	18
3.6	Requisito adicionar turma grade.	18
3.7	Requisito retirar turma grade.	19
3.8	Requisito alterar turma.	19
3.9	Requisito excluir turma.	20
3.10	Requisito associar sala.	20
3.11	Requisito associar período.	21
3.12	Requisito associar professor.	21
3.13	Relação das regras do negócio.	22
3.14	Relação dos requisitos de interface.	23

Lista de Códigos

A.1 JSON	51
----------------	----

Lista de Abreviaturas e Siglas

API	Application Programming Interface
CSS	Cascading Style Sheets
DEPTO	Departamento
GNU	GNU's Not Unix
GPL	GNU General Public License
HTML	HyperText Markup Language
ID	Identificador
JSON	JavaScript Object Notation
MIT	Massachusetts Institute of Technolog
RFC	Request for Comments
SAGeH	Sistema Acadêmico de Gestão de Horário
STP	School Timetabling Problem
UML	Unified Modeling Language
W3C	World Wide Web Consortium
WEB	World Wide Web
WHATWG	Web Hypertext Application Technology Working Group
WWW	World Wide Web
XML	Extensible Markup Language

Sumário

Agradecimentos	i
Resumo	ii
Abstract	iii
Lista de Figuras	iv
Lista de Tabelas	vi
Lista de Códigos	vii
Lista de Abreviaturas e Siglas	viii
1 Introdução	1
1.1 Cenário Geral	1
1.2 Objetivo do Trabalho	3
1.3 Organização do Trabalho	3
2 Fundamentação	4
2.1 Timetabling Problem	4

2.1.1	Complexidades	5
2.1.2	Variações de Timetabling	6
2.2	Visualização da Informação	7
2.2.1	Desafios	8
2.3	Trabalhos Relacionados	9
3	Sistema Acadêmico de Gestão de Horário (SAGeH)	12
3.1	Motivação	12
3.2	Descrição Sistema	13
3.2.1	Descrição de Requisito	13
3.2.1.1	Atores	13
3.2.1.2	Requisitos de Usuário	14
3.2.1.3	Requisitos de Sistema	16
3.2.2	Regras de Negócios	22
3.2.3	Requisitos de Interface	23
3.2.4	Diagrama de Casos de Usos	24
3.2.4.1	Descrição de Caso de Uso	25
3.2.5	Modelagem de Classes	28
4	Implementação	31
4.1	Tecnologias Utilizadas	31
4.1.1	Arquitetura	31
4.1.2	Linguagem JavaScript	32
4.1.3	HTML e CSS	33

4.1.4	Biblioteca jQuery	33
4.1.5	FullCalendar	34
4.1.6	JSON	35
4.2	Resultado	35
4.3	Problemas Encontrados	44
5	Conclusão	45
5.1	Considerações Finais	45
5.2	Trabalho Futuro	46
	Referências	47
A	Criação do Arquivo de Configuração	50

Capítulo 1

Introdução

1.1 Cenário Geral

Atualmente a tecnologia se torna essencial em varias situações e lugares, com esse auxilio é capaz desenvolver novas ferramentas e procedimentos para solucionar incontáveis problemas no cotidiano de empresas e instituições, podemos citar um problema comum que acontece em um ambiente educacional, que é a confecção de uma grade de horária de uma instituição de ensino.

A criação de grade de horários, nada mais é que um problema de alocação de recursos, também conhecido como *Timetabling Problem*, muito estudado devido à facilidade de modificação das variáveis e das restrições envolvidas, podendo então, ser aplicado em diversos contextos. Porém mesmo com o grande interesse nessa área de pesquisa, percebe-se que as instituições de ensino enfrentam normalmente em todo início de período letivo uma dificuldade para gerir essa criação de horários, dado que é uma tarefa complexa e árdua, que pode exigir vários dias de trabalho de muitos profissionais e não há uma ferramenta adequada para dar o apoio e auxiliar para execução dessa tarefa.

Chefes de departamento e coordenadores de curso dedicam horas (ou dias) de trabalho em busca de uma solução que, quase invariavelmente, não é ótima. De fato, trata-se de um problema combinatório complexo, pois o conjunto de interesses

particulares da instituição, professores e alunos geram diversas condições conflitivas a serem atendidas. Esta tarefa torna-se tanto mais difícil quanto maiores forem os conflitos de alocação da demanda (alunos, turmas, horários) com os recursos existentes (professores, salas de aula, laboratórios).

O problema da alocação didática em escolas já foi alvo de muitos trabalhos. Devido a sua natureza combinatória, ele é caracterizado como sendo de alta complexidade computacional. Existem algumas ferramentas comerciais que prometem a geração automatizada de grades de horários, entretanto, sua utilização é pouco frequente uma vez que este tipo de problema incorre em necessidades específicas de um determinado curso, em detrimento das soluções genéricas existentes.

Outro fator que abrange o problema de alocação de horário é a escassez de software capaz de mostrar em tempo de execução, de modo interativo, os conflitos ocorridos. A maioria dos sistemas estudados está preocupada somente com os modelos matemáticos, visando encontrar uma solução, no qual nem sempre a solução encontrada será ótima. Se levarmos em conta a forma de visualização dos dados gerados dos sistemas atuais, encontraremos outro empecilho, que é o descuido com a interação homem-máquina e a falta da utilização dos princípios de usabilidade para esse propósito.

Por causa das necessidades percebidas anteriormente, acaba possibilitando um estudo para atender os pontos mais esquecidos, por exemplo, a visualização mais adequada dos dados e a inclusão de um modo interativo para detecção de conflitos. Deste modo, conseguiria unir o que já existe sobre os estudos de otimização, com os conceitos de design de interface e visualização de dados com várias dimensões

Dado esse cenário, esse trabalho tem por prioridade atender a carência de software especializado em detecção de conflitos, que dê apoio através de uma interface agradável, que tenha preocupação com a integração com o usuário e que utilize uma arquitetura baseada em cliente/servidor.

1.2 Objetivo do Trabalho

Este projeto tem como propósito desenvolver e implementar uma ferramenta de detecção de conflitos para ser utilizada como apoio na alocação de recursos, no qual permitirá também gerenciar toda parte logística para criação de uma grade de horário. Essa ferramenta contará com a inserção, remoção, consulta e a exibição para cada um dos elementos a seguir:

- Disciplinas, professores, salas, ofertas de turmas, cursos e departamentos.

E por fim, permitirá a alocação das turmas em seus determinados horários informando-o seus possíveis conflitos.

1.3 Organização do Trabalho

Esta monografia foi organizada da seguinte maneira:

O capítulo 2 discute os conceitos referentes à fundamentação teórica do trabalho, abordando os conceitos de *Timetabling Problem* e visualização de dados, mostrando os trabalhos relacionados nessas áreas.

O capítulo 3 pretende apresentar a proposta do trabalho, descrevendo seus principais requisitos e casos de usos.

No capítulo 4 serão discutidos os aspectos da implementação da solução proposta e as dificuldades encontradas.

No Capítulo 5 apresenta as considerações finais e as propostas de melhoria para trabalhos futuro. Por fim, são apresentadas as referências bibliográficas e o apêndice deste trabalho.

Capítulo 2

Fundamentação

Para que seja possível o entendimento das técnicas e tecnologias utilizadas, este capítulo apresenta os conceitos teóricos de maior relevância. As seções tratam os seguintes assuntos: *Timetabling Problem*, onde será apresentado o problema de alocação de recursos, sua complexidade e suas variações; Visualização da Informação, nesta seção será detalhado a importância da apresentação dos dados e as suas características; Por fim, serão mencionados os trabalhos na literatura que utilizam esse conjunto de técnicas dentro do contexto da otimização e da visualização dos dados.

2.1 Timetabling Problem

Timetabling é o nome utilizado pela literatura para referenciar problemas de alocação de recursos em seus respectivos horários. De acordo com CORNE *et al.* (1994), é em essência um problema de escalonamento sujeito a restrições onde os eventos são considerados recursos do problema que devem acontecer em um período finito de tempo. Sendo assim, tais recursos não podem ser solicitados por mais de dois eventos ao mesmo tempo, portanto deve existir uma quantidade suficiente de recursos para atender os eventos durante todo tempo de escalonamento [8].

Para BURKE *et al.* (2006), um problema genérico de timetabling incluir certo número de eventos, por exemplo, cursos, exames, salas de aula entre outros, em um

limitado número de períodos de tempo enquanto satisfaz, tanto quanto possível, o maior número de restrições requeridas [4]. Para ROSS *et al.* (2003), um problema genérico de timetabling pode ser definido por três conjuntos básicos. São eles:

- $E = \{e1, e2, \dots, ev\}$, ou seja, um conjunto finito de eventos que inclui atividades diversas como exames, seminários, projetos, ou aulas;
- $T = \{t1, t2, \dots, ts\}$, que é um conjunto finito de horários, para realização dos eventos;
- $A = \{a1, a2, \dots, am\}$, que implica em um conjunto finito de agentes (instrutores, monitores ou professores), que têm o papel de monitorar eventos particulares.

Da definição anterior é válido representar o conjunto de elementos envolvidos no contexto de timetabling pelo conjunto $\{e, t, a\}$ onde $e \in E$ (conjunto de eventos ou atividades), $t \in T$ (conjunto de horários) e $a \in A$ (conjunto de recursos ou agentes), podendo ser interpretado como: o evento e inicia em um determinado tempo t e tem como agente a [19].

Conclui-se que uma timetabling nada mais é do que uma coleção de triplas como o descrito, uma por evento, sendo que as escalas a serem produzidas não devem violar um conjunto de restrições pré-definido pelo contexto.

2.1.1 Complexidades

Em 1936 TURING mostrou através do problema da parada que existem problemas que não podem ser resolvidos por nenhum computador. Dado isto, algoritmos que tenha entradas de tamanho n e possuem eficiência em relação ao tempo de execução pertence a uma função do tipo $O(n^k)$, são chamados de algoritmos de tempo polinomial. Problemas que não podem ser resolvidos em tempo polinomial são nomeados de algoritmos de tempo não polinomial [21].

De acordo com CORMEN *et al.* (2009), problema de decisão solúvel em tempo polinomial pertence à classe tipo P, enquanto os problemas de classe NP são veri-

ficáveis em tempo polinomial e resolvidos em tempo não polinomial. Os problemas NP-Completo que são uma subclasse dos NP-Difícil, são aqueles que não se conhece uma solução determinística de tempo polinomial. Isso significa que um problema NP-Completo pode ser reduzido a outro NP-Completo qualquer. Portanto, um problema NP-Difícil não pertence necessariamente a uma classe NP [7].

De acordo com a literatura científica referente à classificação de complexidade do problema de timetabling, encontra-se em sua atribuição como no mínimo NP-Difícil e no máximo NP-Completo.

O problema de alocação de recurso é um problema de difícil solução, principalmente quando o número de variáveis é elevado. Foi descrito por GAREY e JOHNSON (1979), como um problema de complexidade NP-Difícil, não determinístico em tempo polinomial, ou seja, o trabalho necessário para sua resolução cresce exponencialmente com o tamanho do problema, demanda um esforço de muitas pessoas e tempo para encontrar uma solução de forma manual, e mesmo assim, pode não gerar uma boa solução [13].

Segundo CERDEIRA-PENA *et al.* (2008), o problema de timetabling consiste em incluir todas as restrições, no qual admite uma representação matemática para as condições, de modo que uma solução exata pode ser obtida pela aplicação técnicas bem conhecidas neste campo. No entanto, é conhecido por ser um problema NP-Completo. Na prática, a elevada dimensionalidade do problema torna impossível para encontrar uma solução exata, sendo necessário utilizar métodos de aproximação para encontrá-la. Na maioria dos casos, levam a boa qualidade soluções em um período de tempo razoável, mesmo em detrimento de não garantir que a melhor solução seja alcançada [5].

2.1.2 Variações de Timetabling

Existem vários tipos de *Timetabling Problem* encontrados na literatura, no qual pode ser aplicado a diversos contextos, e para isso basta modificar as variáveis e as restrições envolvidas no problema. Algumas dessas variações podem ser descritas como: Escalonamento de funcionários em turnos, remanejamento de máquinas em

fábrica, tabelas de horários ou recursos (exames, salas de aula, entre outros) para escolas ou para universidades. Assim, o problema é caracterizado por uma natureza fixa e um conjunto de restrições variáveis.

Seguindo o contexto das variações da timetabling, vamos concentrar no problema de criação de grade de horários decorrentes de instituições de ensino, conhecido também como *school timetabling problem* (STP), que estão relacionados com a tarefa de distribuição de professores, períodos de tempo, as lições e os recursos disponíveis (salas de aula, laboratórios, entre outros), de tal maneira que algumas exigências particulares deveram ser satisfeitas.

2.2 Visualização da Informação

A visualização da informação obteve destaque após o surgimento das interfaces gráficas presentes nos computadores dos dias atuais, no qual pode oferecer informações mais precisas e ricas em relação a sua qualidade de apresentação utilizando recursos gráficos semelhantes aos do mundo real. Entretanto, nem sempre é necessário utilizar recursos computacionais para construção de estruturas de apresentação de informações.

Grandes volumes de informação são produzidos e acessados pelos usuários todos os dias, dentre elas, muitas se tornam irrelevantes e desnecessárias ao seu contexto de interesse. Dessa forma, a sobrecarga de informações é uma das principais preocupações na representação dos resultados obtidos, temos que informar também, que existe uma necessidade de saber da qualidade dos dados a serem trabalhados, para que a representação em estruturas de visualização da informação seja realizada da melhor forma possível.

Segundo FREITAS (2001), a visualização da informação pode ser definida como uma área da Ciência que tem por objetivo o estudo das principais formas de representações gráficas para apresentação de informações, de modo que essa representação visual gerada possa contribuir para interpretação e compreensão das informações apresentadas, produzindo então novos conhecimentos baseados no que está sendo

apresentado. É uma ciência que combina aspectos de computação gráfica, interação homem-computador, cartografia e mineração de dados [12].

Uma representação visual corresponde às figuras empregadas para representar o conjunto de dados que estão em análise. Além dos gráficos tradicionais para apresentação de dados que permitem observar as relações entre atributos, pode adicionar elementos visuais como, cores ou símbolos, para codificar os relacionamentos entre entidades ou elementos de dados.

2.2.1 Desafios

Há vários desafios a serem superados em relação à visualização de informação, seja na consolidação de suas técnicas e conceitos, seja na utilização de seus resultados por outras áreas. Nesse sentido, CHEN (2005) aponta alguns dos problemas na área, listados a seguir [6]:

- Lentidão nos estudos de usabilidade e avaliações empíricas na área.
- Entender tarefas perceptivo-cognitivas e identificar agrupamentos e tendências de pontos em uma representação visual.
- Definir sistemas que se adaptem ao nível de conhecimento prévio que o usuário possui para entender a informação visualizada.
- Investir em pesquisas para o aprendizado de conhecimentos de Semiótica e de Comunicação Visual, tomar consciência de problemas de outras disciplinas que podem ser resolvidos com o auxílio de visualização de informação, mostrando assim o potencial da área.
- Definir medidas intrínsecas de qualidade.
- Tratar dos diferentes níveis de escalabilidade nos sistemas, tanto nos softwares desenvolvidos quanto no hardware que os suporta.
- Estudar a estética de uma representação visual, e seu impacto no processo de compreensão dos dados representados.

- Falta de mecanismos de detecção de tendências, sendo necessário utilizar a interdisciplinaridade para ter as colaborações das áreas de mineração de dados e de inteligência artificial, para amenizar esse problema.
- Estudar mecanismos que possibilitem a observação de causalidade, inferências visuais e avaliação de evidências existentes em um conjunto de dados, desenvolvendo algoritmos que resolvam evidências conflitantes e removam ruídos de fundo existentes nos dados.
- Conseguir meios de visualizar todo um domínio de conhecimento.

O conjunto de problemas citados indicam os desafios que a área de visualização de informação atua e onde poderá evoluir, mostra o crescente desafio que é a análises dos dados e suas formas de representação.

2.3 Trabalhos Relacionados

Como vimos na Seção 2.1, a alocação de professores e disciplinas é um problema complexo, de modo que sua solução pode aumentar muito o tempo computacional, classificando-o como de ordem NP-Difícil. Este tipo de problema pode ser aplicado para qualquer tipo de instituição de ensino. Porém, a elaboração da grade de horário em universidades torna essa tarefa mais árdua e complexa, simplesmente pelo fato de possuir um número de disciplina e professores bastante elevado.

Nesta seção são apresentados alguns dos trabalhos relacionados ao tema de alocação de recursos, tais problemas tem sido objeto de pesquisa usando-se inúmeros métodos de solução, deste o final da década de 50. Percebe-se atualmente que este problema continua sendo de enorme interesse, principalmente de especialista em pesquisa operacional, devido à natureza combinatória, pois, para encontrar uma solução ótima ou conveniente, é necessário analisar um grande número de combinações.

Outro fator interessante para analisarmos é a eficiência da apresentação dos resultados gerados pelos algoritmos de otimização propostos, verificando se as formas de visualização utilizadas ajudam a compreender melhor os dados expostos, como

foi enfatizada na Seção 2.2, sobre a importância da visualização da informação para o entendimento de um problema e conseqüentemente extrair novos conhecimentos dos resultados apresentados.

Sabemos que soluções para o *Timetabling Problem* podem ser aplicadas em diversas áreas como na construção civil, projetos de tecnologia da informação e até mesmo na alocação de recursos financeiros. Principalmente pelo fato de tratar-se da busca pela otimização dos recursos disponíveis, tornando-o assim uma das características fundamentais para o sucesso de um projeto. Por outro lado, as soluções apresentadas não são ótimas, criando a necessidade de alguma ferramenta que possa ajudar no refino dessa solução, ou até mesmo, se adequar as necessidades específicas do cliente que não foram analisadas anteriormente pelo algoritmo.

A primeira formulação para o STP foi apresentado por GOTLIEB (1962), afirmando que o problema consistia em fixar um conjunto de aulas num determinado período de tempo e atender as exigências para o cumprimento da grade curricular. Em cada aula era necessário atender um grupo de alunos, com obrigatoriedade de um único professor [14]. Nos anos seguintes LAWRIE (1969) e DE WERRA (1970), propuseram para o mesmo problema, um modelo baseado em programação linear inteira [17] e um modelo baseado em algoritmo de fluxo de rede [11], respectivamente. Já na década de 80 GANS (1981), afirma ser impossível garantir que todas as restrições sejam atendidas, propondo então um modelo heurístico de resolução para o problema STP [10].

A partir da evolução tecnológica, que trouxe velocidade de processamento, armazenamento e de desenvolvimento de software e com a popularização dos computadores pessoais, nos meados de 90, obteve um grande número de trabalhos relacionados na área de otimização.

Podemos destacar pelo fato da abordagem diferente das tradicionais, o estudo de SMITH, ABRAMSON e DUKE (2003), comparava os resultados obtidos utilizando redes neurais artificiais, aplicando a técnica de Hopfield, que são ferramentas matemáticas biologicamente inspirada que pode ser usada para resolver problemas de difícil otimização, com as melhores abordagens meta-heurísticas. Caso os resultados

fossem similares justificaria o investimento na implementação de um hardware de rede neural [20].

O estudo de CORRÊA *et al.* (2010) conseguiu unir duas situações, foi utilizado o algoritmo genético para resolução do STP, além disso, o sistema foi modelado utilizando técnicas de engenharia de software, com interface para web [9], de acordo com os princípios de usabilidade de NIELSEN (1993) [18]. Porém o sistema desenvolvido pelos autores tem limitações em relação a mudanças pontuais nos dados, sendo sempre necessário criar uma grade para cada atualização, não permitindo que o usuário faça ajustes ao final da cada geração da grade de horários.

A tese realizado por BORNIA POULSEN (2012) apresentou um modelo baseado em meta-heurística simulated annealing para tratar o STP [3], diferente dos demais, o autor apresentou mais claramente as diferentes visualizações dos resultados, entretanto, ficou a desejar nas questões de interatividade dos dados após as análises, não permitindo modificar os elementos pontualmente caso necessário. Outro empecilho foi à escolha da arquitetura voltada somente para desktop, não se preocupando com a escalabilidade que poderia atingir se criasse um software voltado para web.

O que foi percebido através dos trabalhos relacionados é a preocupação com a otimização da grade de horário, visando os modelos matemáticos, porém as formas de apresentação destes dados ao usuário ficaram para segundo plano, tendo na maioria das vezes somente uma tabela para visualização, no qual é estático, sem a possibilidade de mudanças após as execuções das funções para geração da grade de horário. Deste modo, fica em aberto o caminho para realizar os estudos para atender essas situações para visualização mais adequada e de modo interativo dos dados.

Capítulo 3

Sistema Acadêmico de Gestão de Horário (SAGeH)

3.1 Motivação

Através do cenário mostrado no Capítulo 2, no qual foi apresentado o *Timetabling Problem* com seu alto custo computacional e sua importância para ser estudado, juntamente com a visualização da informação, uma área aparentemente nova, que procura sanar as complicações que são geradas quando é usado um tratamento inadequado na apresentação dos dados, dificultando assim a análise dos mesmos, foi percebido a necessidade de obter um sistema onde consiga mostrar de forma interativa os conflitos e as informações das diversas dimensões que ocorrem no problema de criação da grade de horários.

Desta forma, foi utilizado como base de estudo o Departamento de Ciência da Computação do Instituto Multidisciplinar de Nova Iguaçu da Universidade Federal Rural do Rio de Janeiro, para fornecer os requisitos para o desenvolvimento do sistema, onde a criação da grade de horária é feita de forma manual utilizando planilhas eletrônicas onde não é possível visualizar os dados com clareza e nem verificar os conflitos instantaneamente, desta forma percebe-se a dificuldade de verificar os problemas relacionados a essa tarefa.

Portanto, o sistema SAGeH vem para facilitar essa criação da grade de horários e alocação de professores, disciplinas e salas, agilizando o tempo de trabalho dos envolvidos, atendendo os pontos mais esquecidos, por exemplo, a visualização mais adequada dos dados e a inclusão de um modo interativo para detecção de conflitos. Deste modo, conseguiria unir o que já existe sobre os estudos de otimização, com os conceitos de design de interface e visualização de dados com várias dimensões.

3.2 Descrição Sistema

Nesta seção serão mostrados e descritos as relações dos atores com o sistema, os requisitos do sistema através dos casos de uso e diagramas, que serviram de base para o funcionamento do projeto.

3.2.1 Descrição de Requisito

Ha diferentes níveis de detalhamento dos requisitos que na Engenharia de Requisitos apresenta-se ser um constante problema. Para resolver isso foi realizada uma distinção usando os seguintes termos.

3.2.1.1 Atores

O ator interage com o sistema sempre solicitando uma ação e recebendo uma reação do sistema. Não pertencem ao sistema estando fora dele, mas são responsáveis por iniciar os casos de usos. Um ator pode fazer parte de vários casos de uso, podendo ser pessoas que irão interagir com o sistema, ou outra parte do sistema ou ainda outro sistema [2].

Compreendendo a definição de ator e percebendo através das análises dos requisitos do usuário, foram definidos os atores do sistema, no qual seria o chefe de departamento, o coordenador e o professor, segue a descrição de cada elemento.

O chefe de departamento tem uma relação de herança com o professor, será o cliente final do sistema, atualmente tem todas as ações que um ator pode ter no

sistema. O coordenador também tem uma relação de herança com o professor, porém terá suas ações limitadas no sistema em comparação ao chefe de departamento. Já o professor participa do sistema com poucas ações.

Entretanto o sistema foi desenvolvido para atender todas as requisições para qualquer usuário que tenha o acesso. Porém com essa definição será possível incrementar novos módulos posteriormente e já deixa definido quais são os atores que interagem com o sistema.

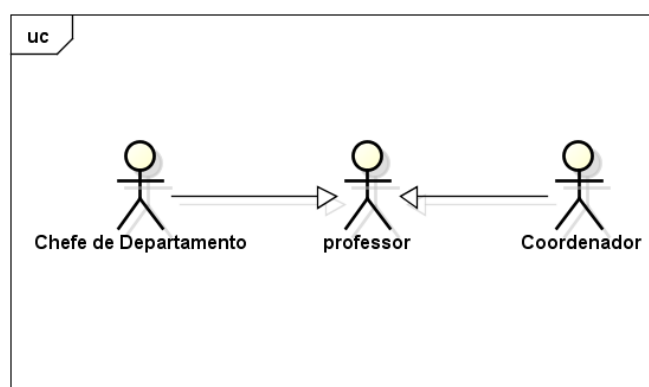


Figura 3.1: Diagrama de Atores.

3.2.1.2 Requisitos de Usuário

São usados para designar um detalhamento dos requisitos em alto nível sobre as funções que o sistema deve fornecer e as restrições que deve possuir. Estes requisitos não devem possuir detalhamentos técnicos, porém deve ser compreensível ao cliente/usuário, especificando somente o comportamento externo do sistema.

Tabela 3.1: Relação dos requisitos de usuário.

ID	Data	Descrição	Autor
RU01	12/04/2014	O sistema terá que possuir inclusão e exclusão de ofertas de turma.	Coordenador
RU02	12/04/2014	O sistema terá vários modos de exibição da grade de horário.	Chefe Depto

Continua na próxima página

Tabela 3.1 – Continuação da página anterior

ID	Data	Descrição	Autor
RU03	12/04/2014	O sistema terá que permitir associar as salas nas ofertas de turmas.	Chefe Depto
RU04	12/04/2014	O sistema terá que permitir associar os professores nas ofertas de turmas.	Chefe Depto
RU05	12/04/2014	O sistema terá que permitir associar os períodos nas ofertas de turmas.	Coordenador
RU06	12/04/2014	O sistema imprimirá a grade de horário quando solicitado.	Professor
RU07	12/04/2014	O sistema terá que salvar a grade de horário.	Chefe Depto
RU08	12/04/2014	O sistema deverá verificar conflito de horário entre os professores.	Chefe Depto
RU09	12/04/2014	O sistema deverá verificar conflito de horário entre as salas.	Chefe Depto
RU10	12/04/2014	O sistema deverá verificar conflito de horário entre as disciplinas.	Chefe Depto
RU11	09/07/2014	Algumas ofertas de turma poderá ter horário fixo na grade no qual não poderá ser alterado.	Chefe Depto
RU12	09/07/2014	Os tempos de aula das ofertas de turmas poderão ser associados para salas diferentes.	Chefe Depto
RU13	09/07/2014	As ofertas de turma poderá possui quantidade de tempo diferenciado.	Chefe Depto
RU14	18/09/2014	O sistema terá sua grade montada em cima de um arquivo modelo.	Chefe Depto
RU15	18/09/2014	O sistema terá sua grade montada em cima de um arquivo externo.	Chefe Depto
RU16	03/11/2014	O sistema deverá ter uma estatística dos professores.	Chefe Depto

3.2.1.3 *Requisitos de Sistema*

São usados para designar um detalhamento dos requisitos em alto nível sobre as funções que o sistema deve fornecer e as restrições que deve possuir. Estes requisitos não devem possuir detalhamentos técnicos, porém deve ser compreensível ao cliente/usuário, especificando somente o comportamento externo do sistema.

A seguir serão mostrados os principais requisitos de sistema, aqueles que foram essenciais para o processo de desenvolvimento e assim elucidar melhor o trabalho.

Função	Criar Grade
<i>Descrição</i>	O sistema terá uma área para escolher um curso para dar início a uma nova grade.
<i>Entradas</i>	Não se aplica.
<i>Saídas</i>	Geração da grade para o curso escolhido.
<i>Ação</i>	O usuário solicita a lista de cursos disponíveis e realiza sua escolha.
<i>Necessidades</i>	Criar uma nova grade utilizando um arquivo base do curso escolhido.
<i>Pré-condições</i>	Não se aplica.
<i>Pós-condições</i>	Grade pronta para inserção e visualização das ofertas.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS01

Tabela 3.2: Requisito criar grade.

Função	Carregar Grade
<i>Descrição</i>	O sistema terá uma área para buscar um arquivo externo, onde poderá abrir ou iniciar, de acordo com a configuração do mesmo, uma nova grade.
<i>Entradas</i>	Arquivo externo.
<i>Saídas</i>	Geração da grade para o curso escolhido.
<i>Ação</i>	O usuário escolhe um arquivo e envia para dar inicio a abertura da grade.
<i>Necessidades</i>	Criar ou abrir uma grade com base no arquivo externo escolhido.
<i>Pré-condições</i>	Arquivo externo deverá seguir as normas estabelecidas.
<i>Pós-condições</i>	Grade pronta para inserção e visualização das ofertas.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS02

Tabela 3.3: Requisito carregar grade.

Função	Salvar Grade
<i>Descrição</i>	Terá um acesso visível para salvar a grade a qualquer momento.
<i>Entradas</i>	Não se aplica.
<i>Saídas</i>	Arquivo salvo com todas as modificações realizadas no formato JSON.
<i>Ação</i>	O usuário clica no botão para salvar a grade.
<i>Necessidades</i>	Salvar as modificações realizadas na grade de horários.
<i>Pré-condições</i>	Deverá existir uma grade.
<i>Pós-condições</i>	Não se aplica
<i>Funcional</i>	Sim
<i>Identificador</i>	RS03

Tabela 3.4: Requisito salvar grade.

Função	Inserir Turma
<i>Descrição</i>	Inserir uma ou mais oferta de turma no período escolhido.
<i>Entradas</i>	Número do período selecionado.
<i>Saídas</i>	Oferta de turma criada.
<i>Ação</i>	O usuário clica no botão e escolhe uma ou mais disciplina.
<i>Necessidades</i>	Criar ofertas de turmas.
<i>Pré-condições</i>	Deverá existir uma grade e no mínimo uma disciplina.
<i>Pós-condições</i>	Oferta de turma disponível para alterações.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS04

Tabela 3.5: Requisito inserir turma.

Função	Adicionar Turma Grade
<i>Descrição</i>	Inserir um tempo de aula da oferta de turma na grade de horário.
<i>Entradas</i>	Número do período ou da sala ou nome professor pré-estabelecido.
<i>Saídas</i>	Tempo de aula da oferta de turma inserida na grade.
<i>Ação</i>	O usuário arrasta o tempo da oferta e solta no lugar desejado.
<i>Necessidades</i>	Alocar os tempos das ofertas na grade de horário.
<i>Pré-condições</i>	Deverá existir uma oferta de turma criada.
<i>Pós-condições</i>	Tempo da oferta de turma alocado na grade.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS05

Tabela 3.6: Requisito adicionar turma grade.

Função	Retirar Turma Grade
<i>Descrição</i>	Retirar a oferta de turma da grade de horário.
<i>Entradas</i>	Tempo da Oferta de Turma.
<i>Saídas</i>	Oferta de turma retirada da grade.
<i>Ação</i>	O usuário seleciona a opção, retirar da grade, localizado dentro do alterar oferta de turma ou através do ícone de excluir.
<i>Necessidades</i>	Retirar as ofertas da grade de horário.
<i>Pré-condições</i>	Deverá existir uma oferta de turma criada.
<i>Pós-condições</i>	Oferta de turma sairá da grade e aparecerá no box lateral.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS06

Tabela 3.7: Requisito retirar turma grade.

Função	Alterar Turma
<i>Descrição</i>	Realizar alterações no tempo da oferta de turma.
<i>Entradas</i>	Tempo da Oferta de Turma.
<i>Saídas</i>	Oferta de turma atualizada.
<i>Ação</i>	O usuário seleciona a oferta de turma e realiza a ação desejada.
<i>Necessidades</i>	Modificar os atributos da oferta de turma.
<i>Pré-condições</i>	Deverá existir uma oferta de turma criada.
<i>Pós-condições</i>	Oferta de turma atualizada.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS07

Tabela 3.8: Requisito alterar turma.

Função	Excluir Turma
<i>Descrição</i>	Remover a oferta de turma do sistema.
<i>Entradas</i>	Oferta de Turma.
<i>Saídas</i>	Oferta de turma removida.
<i>Ação</i>	O usuário seleciona a oferta de turma e realiza a ação de remover.
<i>Necessidades</i>	Remover a oferta de turma do sistema.
<i>Pré-condições</i>	Deverá existir uma oferta de turma criada.
<i>Pós-condições</i>	Não se aplica.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS08

Tabela 3.9: Requisito excluir turma.

Função	Associar Sala
<i>Descrição</i>	Adicionar uma sala para a oferta de turma.
<i>Entradas</i>	Oferta de Turma
<i>Saídas</i>	Oferta de turma atualizada.
<i>Ação</i>	O usuário seleciona a oferta de turma e realiza a ação de escolher uma sala.
<i>Necessidades</i>	Adicionar uma sala para a oferta de turma possibilitando a visualização da grade no modo sala.
<i>Pré-condições</i>	Deverá existir uma oferta de turma criada.
<i>Pós-condições</i>	Atualizar a oferta na grade de horário da visualização no modo sala.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS09

Tabela 3.10: Requisito associar sala.

Função	Associar Período
<i>Descrição</i>	Adicionar um ou mais período para a oferta de turma.
<i>Entradas</i>	Oferta de Turma.
<i>Saídas</i>	Oferta de turma atualizada.
<i>Ação</i>	O usuário seleciona a oferta de turma e escolhe os períodos para associar.
<i>Necessidades</i>	Associar a oferta de turma para vários períodos.
<i>Pré-condições</i>	Deverá existir uma oferta de turma criada.
<i>Pós-condições</i>	Atualizar a oferta na grade de horário dos períodos selecionados.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS10

Tabela 3.11: Requisito associar período.

Função	Associar Professor
<i>Descrição</i>	Adicionar um professor para a oferta de turma.
<i>Entradas</i>	Oferta de Turma.
<i>Saídas</i>	Oferta de turma atualizada.
<i>Ação</i>	O usuário seleciona a oferta de turma e escolhe um professor para associar.
<i>Necessidades</i>	Associar a oferta de turma para um professor.
<i>Pré-condições</i>	Deverá existir uma oferta de turma criada.
<i>Pós-condições</i>	Atualizar a oferta na grade de horário do professor selecionado.
<i>Funcional</i>	Sim
<i>Identificador</i>	RS11

Tabela 3.12: Requisito associar professor.

3.2.2 Regras de Negócios

As regras do negócio são políticas, condições ou restrições que devem ser consideradas na execução dos processos existentes em uma organização (GOTTESDIENER, 1999), descrevem a maneira como a organização funciona, normalmente têm influência sobre a lógica de execução de um ou mais casos de uso, por esses motivos é considerada uma parte importante dos processos organizacionais [15].

A seguir mostraremos as regras do negócio especificadas pelos autores para a criação do sistema proposto.

Tabela 3.13: Relação das regras do negócio.

ID	Descrição	Relacionamento
RN01	Uma turma ofertada não pode ser associada para mais de um professor.	RS11
RN02	Uma turma ofertada pode ter o período de tempo associado a salas diferentes.	RS09
RN03	Uma oferta pode ter seus períodos de tempo diferentes em relação ao horário fixo.	RS05, RS07, RI01
RN04	Uma nova turma é criada caso já exista uma turma com a mesma disciplina no período.	RS04
RN05	Ao criar uma oferta de turma de disciplina que já exista em outro período, essa oferta será associada a ela.	RS04
RN06	Uma turma deve ter no mínimo 1 período associado, podendo ser possível ser associada a vários outros períodos.	RS04, RS07
RN07	Um professor não pode lecionar mais de uma turma no mesmo horário.	RS05, RS07, RS11
RN08	Uma sala não pode alocar mais de uma oferta de turma no mesmo horário.	RS05, RS07, RS09
Continua na próxima página		

Tabela 3.13 – Continuação da página anterior

ID	Descrição	Relacionamento
RN09	Um período não pode ter mais de uma turma no mesmo horário.	RS05, RS07, RS10

3.2.3 Requisitos de Interface

Os requisitos de interface são definições específicas a respeito da interface do sistema: cor, estilo, interatividade entre outros, e podem estar relacionados a um ou mais casos de uso do sistema.

Tabela 3.14: Relação dos requisitos de interface.

ID	Descrição	Relacionamento
RI01	Um tempo com horário fixo na grade terá a cor cinza para diferenciar dos demais.	CSU03
RI02	Será mostrado conflito de horários entre professores, salas e turmas modificando a cor dos elementos em questão para vermelho.	CSU03
RI03	Quando todos os períodos de tempo de uma oferta estiverem alocados na grade será representado pela cor verde, se somente um período de tempo estiver na grade terá a cor laranja e se nenhum período de tempo estiver na grade terá a cor azul.	CSU02, CSU03
RI04	Quando passar o mouse por cima de um período de tempo de uma oferta será identificado os outros períodos de tempo relacionados, tanto na grade quanto no box, através de uma cor diferentes das demais (azul claro).	CSU03
Continua na próxima página		

Tabela 3.14 – Continuação da página anterior

ID	Descrição	Relacionamento
RI05	A grade deverá ser criada através de um JSON pré-definido, no qual deverá conter: dados gerais, horário, professor, disciplina, sala e departamento.	CSU01, CSU04

3.2.4 Diagrama de Casos de Usos

O Diagrama de Caso de Uso é uma técnica utilizada para representar graficamente o que o novo sistema irá realizar. Foram constituídos a partir da interação com o cliente/usuário, a fim de realizar uma especificação de comum acordo. Os principais objetivos desses diagramas são decidir e descrever o requisito funcional do sistema e também descrever de forma clara e consistente o que o sistema deverá fazer. É uma descrição dos eventos realizados por um ator no sistema e nomeado por uma frase que indica uma ação iniciada por um ator. Os elementos que compõem os casos de uso são: os atores, caso de uso e o próprio sistema [2].

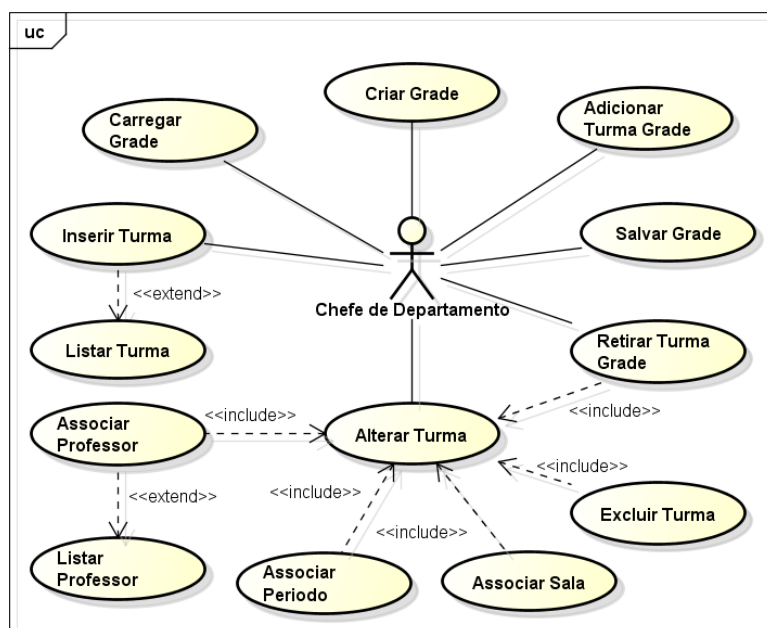


Figura 3.2: Diagrama de Caso de Uso.

3.2.4.1 Descrição de Caso de Uso

Para um melhor entendimento do diagrama de casos de uso, a seguir serão mostrados somente os principais casos de uso, pois os mais simples não tiveram a necessidade da descrição. Assim é apresentado a rotina e fluxo no sistema, de que forma ocorre a interferência do ator, as regras que o caso de uso obedece e as precondições, sempre focando na funcionalidade do sistema.

Esta descrição composta pelos seguintes itens: identificação do caso de uso, o sumário com uma pequena descrição, as regras para mostrar os requisitos associados, os atores do caso de uso, a precondição para o funcionamento, o fluxo principal, alternativo e exceção dos eventos do caso de uso.

Inserir Turma (CSU02)
<p>Sumário: Usuário solicita a inserção de uma turma</p> <p>Regras: RS01, RN04 e RN05</p> <p>Ator Primário: Chefe do Departamento</p> <p>Precondições: Existe uma grade criada</p> <p>Fluxo Principal</p> <ol style="list-style-type: none">1- O usuário solicitará a criação de uma turma clicando em um período;2- Sistema apresenta uma lista de disciplinas;3- Usuário escolhe as disciplinas desejadas;4- Sistema cria as turmas com as disciplinas escolhidas no período selecionado. <p>Fluxo Alternativo (3): Usuário cancela o pedido</p> <ol style="list-style-type: none">a. Sistema não mostra as opções das disciplinas e o caso de uso termina. <p>Fluxo Alternativo (4): Usuário cancela o pedido</p> <ol style="list-style-type: none">a. Sistema criará uma nova turma para aquela disciplina. <p>Fluxo Alternativo (4): Aderindo a RN05</p> <ol style="list-style-type: none">a. Sistema associará a essa turma a referência do período selecionado.

Figura 3.3: Caso de uso inserir turma.

Alterar Turma (CSU03)
<p>Sumário: Usuário solicita uma alteração para uma oferta de turma</p> <p>Regras: RS06, RS07, RS08, RS09, RS10, RS11, RN01, RN02, RN03 e RN06</p> <p>Ator Primário: Chefe do Departamento</p> <p>Precondições: Existe uma grade e uma oferta de turma criada</p> <p>Fluxo Principal</p> <ol style="list-style-type: none">1- O usuário seleciona uma oferta;2- Sistema apresenta os dados dessa oferta;3- Usuário realiza as modificações e confirma;4- Sistema verifica e salva as modificações. <p>Fluxo Alternativo (2): Opção da visualização</p> <ol style="list-style-type: none">a. Sistema apresenta a opção “retirar da grade” caso vier da grade de horário, caso contrário ele apresenta a opção “excluir turma”. <p>Fluxo Alternativo (3): Usuário cancela o pedido</p> <ol style="list-style-type: none">a. Sistema não atualiza os dados e o caso de uso termina. <p>Fluxo Alternativo (3): Usuário retira a turma da grade</p> <ol style="list-style-type: none">a. Sistema não atualiza os dados e vai para o caso de uso Retirar Turma Grade. <p>Fluxo Alternativo (3): Usuário exclui turma</p> <ol style="list-style-type: none">a. Sistema não atualiza os dados e vai para o caso de uso Excluir Turma. <p>Fluxo Alternativo (3): Violação de RN06</p> <ol style="list-style-type: none">a. Sistema emite uma mensagem de aviso e não permite salvar as modificações.

Figura 3.4: Caso de uso alterar turma.

Criar Grade (CSU01)
<p>Sumário: Usuário solicita a criação da grade</p> <p>Regras: RS04 e RI05</p> <p>Ator Primário: Chefe do Departamento</p> <p>Precondições: Não se aplica</p> <p>Fluxo Principal</p> <ol style="list-style-type: none">1- O usuário solicitará a criação da grade;2- Sistema apresenta as opções dos cursos disponíveis;3- Usuário escolhe um curso;4- Sistema gera a grade escolhida. <p>Fluxo Alternativo (2): Usuário cancela o pedido</p> <ol style="list-style-type: none">a. Sistema não mostra as opções dos curso disponíveis e o caso de uso termina.

Figura 3.5: Caso de uso criar grade.

Carregar Grade (CSU04)
<p>Sumário: Usuário solicita o carregamento da grade</p> <p>Regras: RS02 e RI05</p> <p>Ator Primário: Chefe do Departamento</p> <p>Precondições: Não se aplica</p> <p>Fluxo Principal</p> <ol style="list-style-type: none">1- O usuário solicitará o carregamento da grade;2- Sistema apresenta a opção para buscar o arquivo na máquina do cliente;3- Usuário escolhe o arquivo para abrir;4- Sistema gera a grade escolhida. <p>Fluxo Alternativo (2): Usuário cancela o pedido</p> <ol style="list-style-type: none">a. Sistema não mostra a opção para escolher o arquivo e o caso de uso termina. <p>Fluxo Alternativo (3): Violação do RI05</p> <ol style="list-style-type: none">a. Sistema não gera a grade e segue para o item 2.

Figura 3.6: Caso de uso carregar grade.

3.2.5 Modelagem de Classes

Representa uma estrutura estática do sistema mostrando os objetos pertencentes, as relações entre estes, os atributos e as operações que caracterizam estes objetos, isto é, uma representação gráfica formal dos objetos e seus relacionamentos. O diagrama da UML utilizado para representar esse aspecto é o diagrama de classes.

E importante mencionar que o modelo de classes evolui durante as iterações do desenvolvimento do sistema. A medida que o sistema é desenvolvido, o modelo de classes é incrementado com novos detalhes. Existem três níveis de de abstração que o modelo passa. Esses níveis são: domínio, especificação e implementação.

O modelo de classe de domínio representa termos do domínio do negócio. Seu objetivo é esclarecer o problema representado pelo sistema a ser desenvolvido, não considera características da solução a ser utilizada e é construído na fase de análise. Já os modelos de especificação e implementação, consideram detalhe de software a ser utilizada. No entanto, ao contrario do modelo implementação, o de especificação descreve em um nível mais alto de abstração [2].

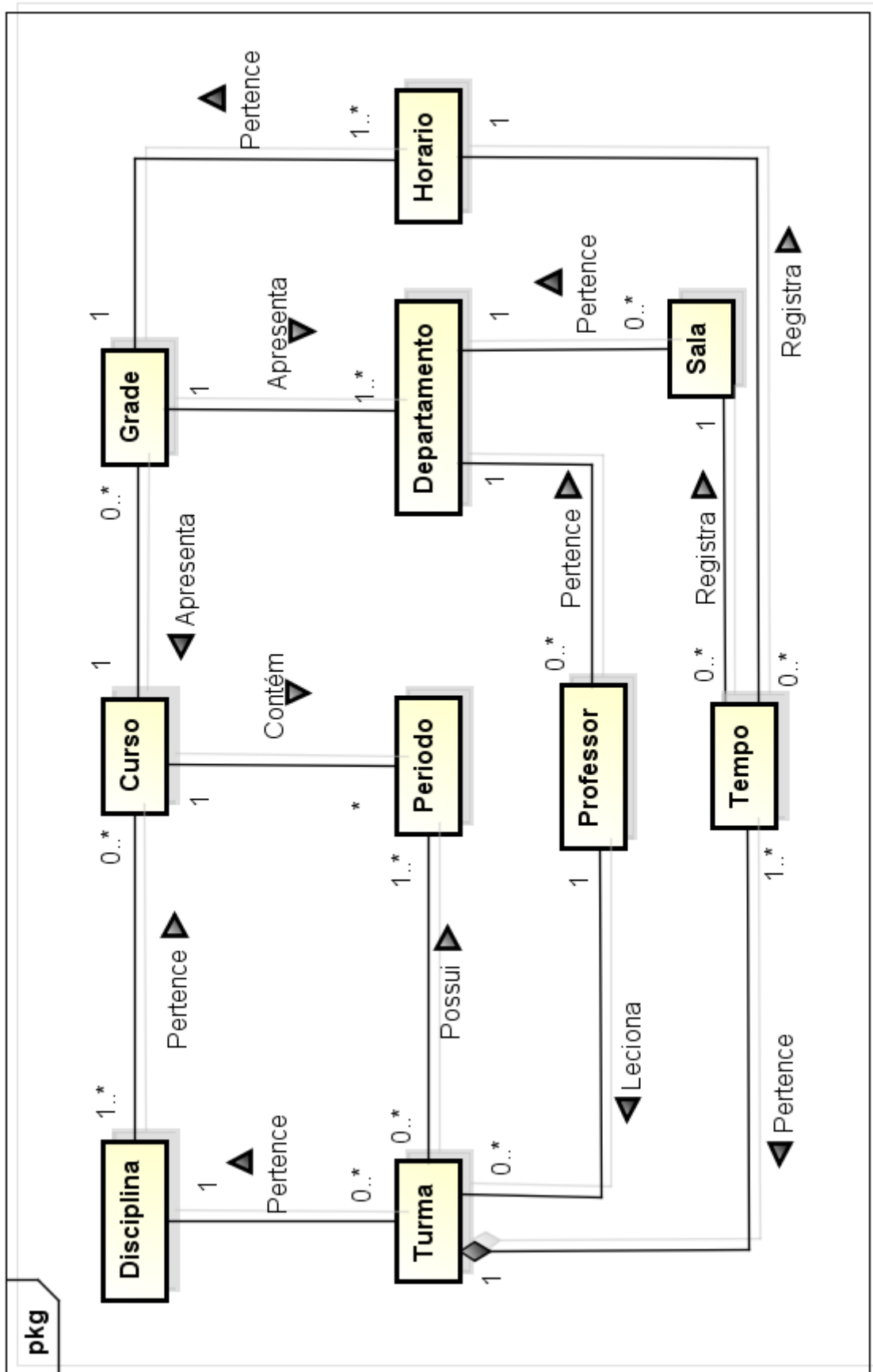
A Figura 3.7 demonstra o diagrama de classes modelado para o sistema, mostrando os tipos de relacionamento utilizados e suas multiplicidades. Para um melhor entendimento do diagrama de classes criado, a seguir apresenta-se a descrição das classes de forma textual.

- **Grade:** é uma classe que apresentada os atributos do período vigente, tendo relação com os métodos para criação de grades. Responsável em armazenar as relações de horário, departamento e curso para ser utilizados pelos métodos responsáveis pela iniciação do sistema.
- **Curso:** esta classe apresenta os atributos referentes ao curso e tem relacionamento com disciplina e período, dado essa relação, os métodos desta classe apresenta a classe grade suas opções para visualização.
- **Disciplina:** classe responsável para apresentar a quantidade de tempo de aula que uma turma terá, além de conter os atributos referentes à disciplina, por

exemplo, nome e código da disciplina. Possui relação com curso e turma.

- Período: classe que armazena a ligação das turmas associadas por período, mostra quais os períodos que vão ser disponível para associação com as ofertas de turma, sendo eles determinados pela relação que tem com o curso.
- Turma: essa classe é responsável para armazenar as ofertas de turmas criadas, tem relação com professor, período, disciplina e tempo, tem acesso aos métodos referentes à criação e edição de turmas e associações.
- Tempo: classe criada para identificar os períodos de tempo que uma oferta de turma possui, os objetos são criados no momento da criação da turma, portando só existe quando uma turma também existir. Deste modo, foi possível dividir a oferta em bloco de tempo, dos quais poderá ser alocados a horários e salas diferentes, além de conter informações da interface do sistema e associações entre os tempos da oferta de turma.
- Horário: essa classe faz a ligação do tempo com a grade, ficando armazenado o tempo da oferta de turma com a grade do período vigente em um determinado horário específico.
- Professor: classe responsável em apresentar os atributos dos professores, ficando disponível para ser utilizada nos métodos de associações nas ofertas de turmas, tem uma relacionamento com o departamento.
- Departamento: classe responsável em mostrar as salas pertencentes ao departamento, ficando assim disponíveis para ser associadas através dos métodos para alguma oferta de turma já criada.
- Sala: é uma classe que apresenta os atributos da sala, por exemplo, capacidade, tipo de sala e localização entre outros, após sua criação fica disponível para ser associada a um tempo da oferta de turma através da classe tempo.

Figura 3.7: Diagrama de Classe.



Capítulo 4

Implementação

Nesse capítulo é dedicado a mostrar as tecnologias utilizadas do decorrer do desenvolvimento do sistema, juntamente com a apresentação da interface utilizada no sistema tendo como objetivo o usuário final.

4.1 Tecnologias Utilizadas

Nessa seção vamos apresentar um pouco da história e funcionalidades básicas das tecnologias que foram utilizadas, da arquitetura escolhida, em fim, todos as ferramentas e conceitos que contribuíram para desenvolvimento do sistema SAGeH.

4.1.1 Arquiteura

A forma mais comum de desenvolvimento de aplicação para web é e a utilização da arquitetura cliente/servidor, segundo BATTISTI (2005), esse modelo é definida como uma arquitetura onde o processamento da informação é dividido em módulos ou processos distintos. Um processo é responsável pela manutenção da informação (Servidor), enquanto que outro é responsável pela obtenção dos dados (Cliente) [1]. Para VASKEVITCH (1995), essa arquitetura é uma abordagem da computação que separa os processos em plataformas independentes que interagem, permitindo que os recursos sejam compartilhados enquanto se obtém o máximo de benefício de cada

dispositivo diferente, ou seja, Cliente/Servidor é um modelo lógico [22].

Outro fator essencial é a escolha do modelo, uns dos mais usados é o modelo de três camadas, na qual as camadas são: camada de apresentação ou interface do usuário, esta camada interage diretamente com o usuário, é através dela que são feitas as requisições como consultas; camada de negócio ou processamento é nela que fica as funções e regras de todo o negócio. Não existe uma interface para o usuário e seus dados são voláteis, ou seja, para que algum dado seja mantido deve ser utilizada a camada de dados; e por fim, a camada citada anteriormente, a de dados ou repositório, esta camada recebe as requisições da camada de negócios e seus métodos executam essas requisições em um banco de dados. Alterando o banco de dados alteraria apenas as classes da camada de dados, e o restante das camadas não seriam afetados por essa alteração.

De acordo com as informações anteriores, podemos definir que a arquitetura na qual foi criado o SAGeH tem algumas das características citadas com algumas ressalvas, sendo ela especificamente um arquitetura voltada somente para o cliente. Onde todo o sistema foi construído com a inclusão da camada das regras de negócio e a parte de camada de dados para armazenamento voltado para o lado do cliente, isso que dizer, junto com interface. Essa solução foi adotada, para a ferramenta ser utilizada de forma independente da arquitetura do servidor, podendo ser acoplada em qualquer linguagem ou modelo adotado por um desenvolvedor.

4.1.2 Linguagem JavaScript

A linguagem JavaScript foi criada pela Netscape Communications Corporation e foi lançada em 1995 integrando a versão 2.0B3 do navegador Netscape e visava implementar uma tecnologia de processamento modo cliente, permite criar pequenos programas embutidos no próprio código de uma página HTML e capazes de gerar números, processar alguns dados, verificar formulários, alterar valor de elementos HTML e criar elementos HTML. Tudo isso diretamente no computador cliente, evitando a troca de informações com o servidor e o tempo passa a depender somente do processamento local do cliente [16].

JavaScript é uma linguagem completa e poderosa que possui muitas das qualidades de diversas outras linguagens como: listas associativas, tipagem dinâmica e expressões regulares de Perl e a sintaxe similar a C/C++, linguagens de grande reconhecimento tanto no mundo acadêmico quanto comercialmente. Além disso, JavaScript é multiparadigma e entre eles destacam-se a programação estrutural e orientada a objeto; possui funções de ordem superior; entre outros.

4.1.3 HTML e CSS

HTML deve início na década de 90 essencialmente como uma linguagem para descrever semanticamente documentos científicos, com adaptações ao longo dos anos permitiu seu uso para descrever uma série de outros tipos de documentos. Atualmente é uma linguagem para estruturação e apresentação de conteúdo para o World Wide Web.

Com a criação do W3C e mais tardar a parceria com WHATWG, foi possível discutir e elaborar padrões para atender melhor as necessidades de evolução da linguagem de marcação para web, tendo como resultado dessa união a apresentação do HTML5 em 2008, que continham novas API's, entre elas uma para desenvolvimento de gráficos bidimensionais, aprimoramento do uso off-line, melhoria na depuração de erros, controle embutido de conteúdo multimídia que eliminaria a necessidade de plug-ins para aplicações multimídia em navegadores.

CSS é a linguagem para descrever a apresentação de páginas da Web, incluindo cores, layout e fontes. Permite modificar a apresentação para diferentes tipos de dispositivos, tais como telas grandes, telas pequenas, ou impressão. É independente do HTML e pode ser usado com qualquer linguagem de marcação baseada em XML.

4.1.4 Biblioteca jQuery

A biblioteca jQuery¹ foi criada por John Resig e disponibilizada como software livre e aberto, ou seja, de emprego e uso regido segundo licença conforme as regras

¹<http://jquery.com/>

estabelecidas pelo MIT – Massachusetts Institute of Technology ou pelo GPL – GNU General Public License. Isso, resumidamente, significa que você pode usar a biblioteca gratuitamente tanto em desenvolvimento de projetos pessoais como comerciais.

jQuery é uma biblioteca JavaScript que possui entre suas características o uso de seletores CSS para localizar elementos componentes da estrutura de marcação HTML da página, ter arquitetura compatível com instalação de plug-ins e extensões em geral, possuir indiferença às inconsistências de renderização entre navegadores, ser capaz de interação implícita isto é, não há necessidade de construção de loops para localização de elementos no documento, admite programação encadeada, ou seja, cada método retorna um objeto, e ser extensível, pois admite criação e inserção de novas funcionalidades na biblioteca existente. Destina a adicionar interatividade e dinamismo às páginas web e assim enriquecer a experiência do usuário.

Sua instalação é trivial, a biblioteca jQuery nada mais é do que um arquivo JavaScript, portanto basta chama-lo na a página web onde serão aplicados os efeitos.

4.1.5 FullCalendar

FullCalendar² é disponibilizado como software livre e aberto, liberado sob a licença MIT. Em resumo, significa que você pode usar a biblioteca gratuitamente tanto em desenvolvimento de projetos pessoais como comerciais.

FullCalendar é um plugin jQuery criado por Adam Shaw, que fornece um calendário completo de eventos, no qual pode ser configurado para ser utilizado no seu próprio formato. Ótimo para exibição de eventos, mas não é uma solução completa para o gerenciamento de conteúdo desses eventos, devendo então ser implementado a parte.

²<http://fullcalendar.io/>

4.1.6 JSON

O formato JSON foi originalmente especificado por Douglas Crockford em 2001, e é descrito no RFC 4627 como: leve, baseado em texto, formato de intercâmbio de dados independente de linguagem. Foi derivada a partir de Programação ECMAScript Idioma Padrão. Define um pequeno conjunto de regras de formatação para a representação do portátil estruturado dados.

Sendo assim, JSON é um padrão aberto baseado em texto que permite a troca de dados legíveis, de fácil leitura para tanto para o programador tanto para a máquina. É derivado da linguagem de JavaScript para representar estruturas simples e matrizes de associação de dados, chamados de objetos, podendo então representar quatro tipos primitivos (strings, números, booleanos e nulos) e dois tipos estruturados (objetos e matrizes).

Apesar de ter uma relação com JavaScript, é completamente independente de linguagem, pois usa convenções que são familiares às linguagens C e familiares, incluindo C++, C#, Java, JavaScript, Perl, Python e muitas outras. No site JSON³ oferece uma lista completa de bibliotecas existente, organizados pela linguagem.

4.2 Resultado

Através das análises dos capítulos anteriores, dos quais obtivemos as restrições e conteúdos capazes para o desenvolvimento do sistema, levando em conta os objetivos do cliente, vamos agora apresentar nessa seção, as telas criadas para o sistema SAGeH. Desta forma, ficará mais claro o entendimento de como ficou a interação entre o usuário e o sistema.

O SAGeH foi implementado para funcionar em um ambiente web, permitindo que seu acesso possa ser realizado de qualquer dispositivo com acesso a internet.

Primeiramente o sistema conta com uma parte de apresentação, onde é mostrado de uma forma rápida as características do sistema e seus objetivos, uma espécie de

³<http://www.json.org/>

vitrine, na qual pode ser visto através da Figura 4.1.



Figura 4.1: Tela de apresentação do sistema.

Após acessar a parte interna do sistema, o menu apresentará as funcionalidades de inserir, carregar e salvar grade, como mostra a Figura 4.2.



Figura 4.2: Menu da parte interna do sistema.

Após a solicitar a opção criar no menu, será mostrado uma lista de cursos para criar uma nova grade para o curso selecionado, como é visto na Figura 4.3.

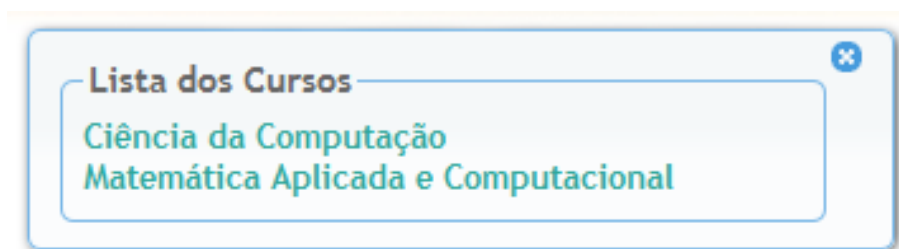


Figura 4.3: Janela com as opções de cursos.

Já na Figura 4.4, mostra a janela de carregamento de uma grade que esteja salva no cliente. Isso quer dizer que, dado qualquer arquivo que esteja configurado de acordo com as regras estabelecidas, das quais podem ser consultadas no Apêndice A,

pode ser aberto pelo sistema, portanto, cria a possibilidade de obter diversas grades curriculares.

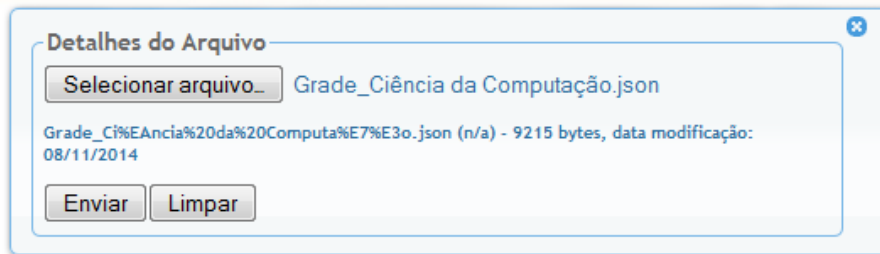


Figura 4.4: Selecionador de grade de horário.

O sistema contém três modos de visualização da grade de horário, sendo elas: a grade de disciplina por período, a grade de professor por disciplina e a grade sala por disciplina. Os modos de visualização foram separados por abas, ficando disponível a qualquer momento para uma possível consulta, como pode ser visto na Figura 4.5.



Figura 4.5: Modos de visualização separado por abas.

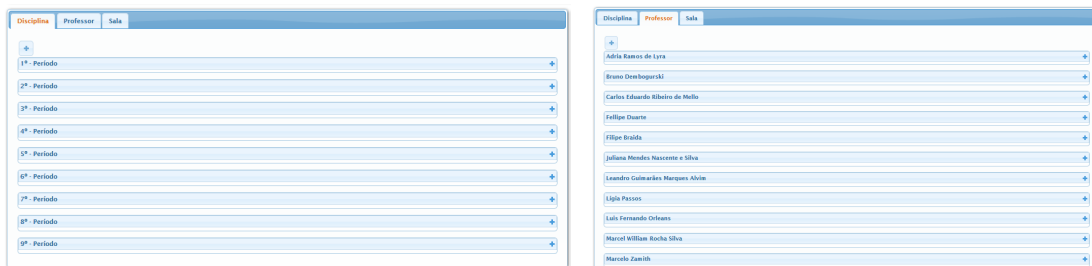
Para darmos continuidade a apresentação do sistema, temos que definir e mostrar através de imagem os conceitos presentes na formação da interface do sistema. De acordo com a Figura 4.6, uma grade de horário é representada por uma matriz 7x6, nela é adicionada os tempos das ofertas de disciplinas. Vamos definir que para cada

período, professor e sala contém uma grade, das quais serão analisadas para mostrar os conflitos.

	Segunda	Terça	Quarta	Quinta	Sexta	Sábado
08:00						
10:00						
12:00						
14:00						
16:00						
18:00						
20:00						

Figura 4.6: Grade de horário para alocação de tempos.

A Figura 4.7(a) e a Figura 4.7(b), apresentam os elementos presentes dentro das estruturas de visualizações da disciplina e do professor respectivamente, percebe-se que é possível com um simples comando, esconder e mostrar o elemento que deseja. Tal funcionalidade também está presente na visualização da sala.

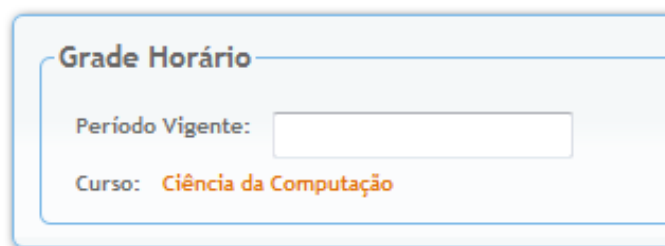


(a) Disciplina.

(b) Professor.

Figura 4.7: Modos de visualizações.

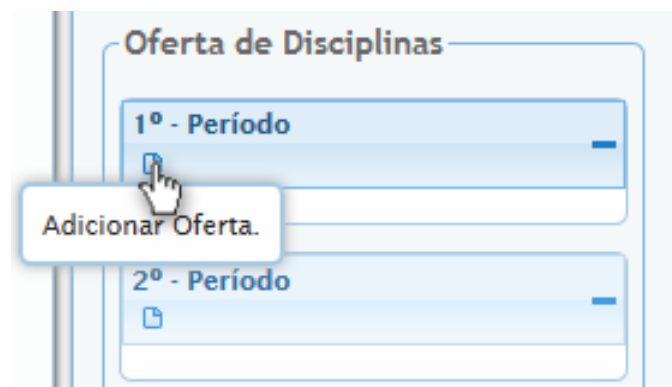
Ao selecionar um curso para criar uma grade de horário, é possível adicionar o período vigente para essa grade, o campo pode ser visto na Figura 4.8.



The image shows a form titled "Grade Horário". It contains a text input field labeled "Período Vigente:" and a label "Curso: Ciência da Computação".

Figura 4.8: Campo para especificar o período vigente da grade.

Outro método que é fundamental para o funcionamento do sistema é a inserção das ofertas de disciplinas, através da Figura 4.9 é possível localizar onde pode ser acessado esse comando.



The image shows a section titled "Oferta de Disciplinas". It contains two sections: "1º - Período" and "2º - Período". A mouse cursor is hovering over a small icon in the "1º - Período" section, and a tooltip box appears with the text "Adicionar Oferta.".

Figura 4.9: Box de ofertas de disciplinas.

Após a chamada do método para adicionar uma oferta, surgirá uma janela onde poderá escolher através de um campo por auto complementação varias disciplinas para ser ofertada no período escolhido, como mostra a Figura 4.10.



The image shows a dialog box titled "Adicionar Oferta". It contains a label "Disciplina:" followed by a text input field with a dropdown menu. The dropdown menu is open, showing three options: "x Geometria Analítica", "x Cálculo II", and "x Computação I". At the bottom of the dialog box, there are two buttons: "Adicionar" and "Cancelar".

Figura 4.10: Campo para adicionar ofertas de disciplinas.

As ofertas de disciplina criadas são alocadas no box do período escolhido obedecendo os requisitos de interface da Seção 3.2.3. Para facilitar o aprendizado do usuário perante as opções que podem ocorrer com uma oferta de disciplina durante a execução dentro do sistema, foi criada uma legenda com os seguintes esclarecimentos:

- Oferta sem horário – são as ofertas que não estão alocadas na grade;
- Oferta sem conflito – são as ofertas alocadas na grade;
- Oferta horário incompleto – são as ofertas que somente um período de tempo está alocado na grade;
- Oferta com conflito – são as ofertas que por algum motivo está em conflito com outro período de tempo;
- Oferta com horário fixo – são as ofertas que estão na grade que tenha a opção de horário fixo ativado, como pode ser vista na Figura 4.11.

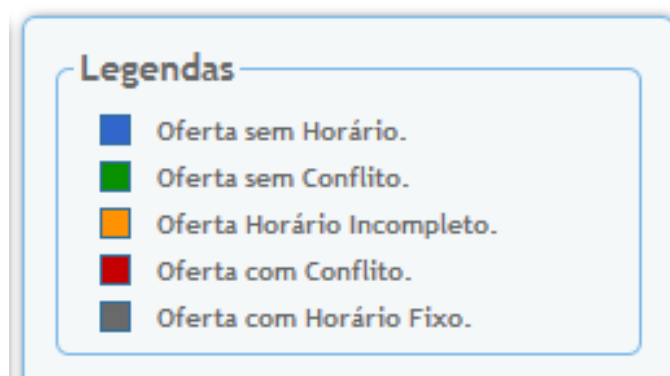


Figura 4.11: Legenda das ofertas de disciplinas.

Na Figura 4.12 vamos demonstrar como fica uma grade de horário onde as ofertas de turmas que estão alocadas estão corretas, ou seja sem conflito ou outros impedimentos, podendo então ser finalizada.

1º - Período						
	Segunda	Terça	Quarta	Quinta	Sexta	Sábado
08:00	T1- Fundamentos de Sistemas	T1- Geometria Analítica	T1- Fundamentos de Sistemas	T1- Álgebra Linear	T1- Computação I	
10:00	T1- Computação I	T1- Geometria Analítica	T1- Cálculo I	T1- Cálculo I	T1- Álgebra Linear	
12:00						
14:00						
16:00						
18:00						
20:00						

Figura 4.12: Grade com horários corretos.

Em seguida apresentaremos na Figura 4.13 um exemplo no qual um período de tempo de uma oferta tem horário fixo ativado.

	Segunda	Terça	Quarta	
08:00				
10:00		T1- Computação I	T1- Computação I	

Figura 4.13: Ofertas com horário fixo ativado.

Já na Figura 4.14 mostraremos uma oferta que esteja com somente um período de tempo alocado na grade.

	Terça	Quarta
	T1- Geometria Analítica	T1- Cálculo I
	T1- Geometria Analítica	

Figura 4.14: Ofertas incompletas na grade.

A representação de conflito entre as ofertas de disciplinas pode ser descritas de dois modos: o conflito local, onde o período de tempo da oferta esta na mesma grade de visualização, como pode ser visto na Figura 4.15(a); e o conflito associado, são conflitos gerado em uma outra visualização na qual esta oferta tem relação ou através da associação dos períodos de tempo desta oferta, sendo representado pela Figura 4.15(b).



(a) Conflito local.

(b) Conflito associado.

Figura 4.15: Representação dos conflitos nas ofertas.

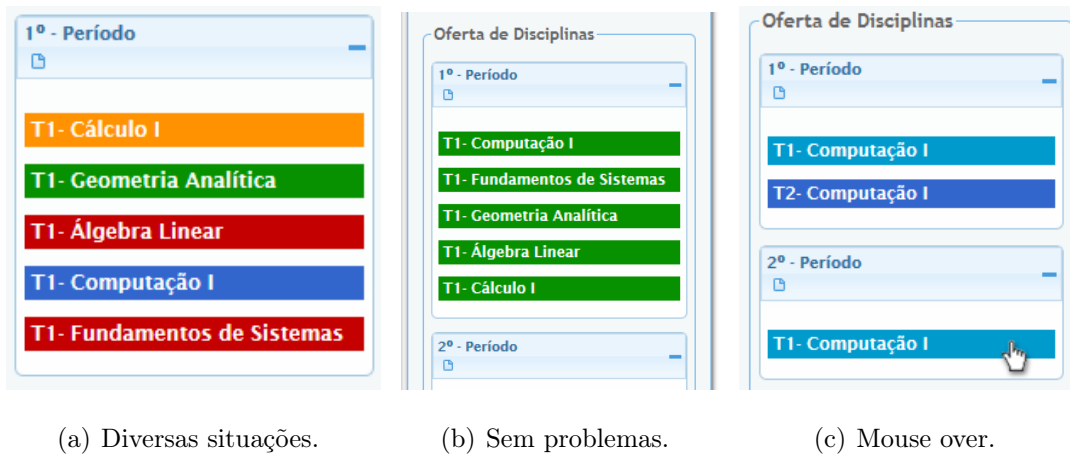
Um exemplo dos conflitos citados anteriormente pode ser visto pela Figura 4.16, no qual mostra através da visualização da grade da sala o conflito local entre a disciplina Estrutura de dados I e Circuitos Digitais, note que, o período de tempo associado a disciplina de Circuitos Digitais sofre o conflito associado.

Sala 301						
Dados da Sala						
Localização: Bloco Multimídia - 3º andar						
Tipo de Sala: Convencional						
Capacidade: 50						
Características: Projektor, quadro negro.						
	Segunda	Terça	Quarta	Quinta	Sexta	Sábado
08:00						
10:00						
12:00						
14:00				T1- Circuitos Digitais - Sala: 301	T1- Estrutura de Dados I - Sala: 301	
16:00						
18:00				T1- Circuitos Digitais - Sala: 301		

Figura 4.16: Grade da sala com conflitos.

Outra representação de conflitos por cores esta presente no box de ofertas de disciplinas por períodos. Assim facilita o usuário identificar quais ofertas está com problemas e também mostrar suas associações.

Na Figura 4.17(a), demosramos as diversas situações que pode ocorrer com as ofertas, na Figura 4.17(b), foi mostrado que todas as ofertas estão corretas e alocadas na grade e já na Figura 4.17(c), apresenta a capacidade de localização das ofertas associadas através do Mouse Over, esse método localiza também ofertas que estejam em outros períodos.



(a) Diversas situações.

(b) Sem problemas.

(c) Mouse over.

Figura 4.17: Representação das ofertas no box.

Na Figura 4.18, vamos apresentar a tela responsável em editar uma oferta de disciplina, não podemos deixar de mencionar que é através deste método que é associado o professor, a sala e os períodos em qual a oferta em questão terá. Lembrando que essas associações obedecem aos requisitos descritos no Capítulo 3.

Outro fator importante dessa interface é a presença dos botões retirar da grade (1) e o excluir oferta (2), que dependendo de onde é chamado o método para editar, pode aparecer um ou outro, sendo assim, toda vez que o método for acionado vindo da grade aparecerá o (1), caso o método vier do box aparecerá o (2), as funcionalidades deste botões são diferentes, o (1) é utilizado para retirar a oferta alocada da grade já o (2) é para excluir a oferta de turma do box e do sistema.



Figura 4.18: Editar oferta de disciplina.

4.3 Problemas Encontrados

Ao longo do desenvolvimento do SAGeH foram encontrados alguns obstáculos que precisaram ser contornados. A primeira delas se refere a uma limitação da biblioteca FullCalendar. Esta limitação foi identificada ao se tentar adicionar uma formatação diferenciada para os conteúdos das ofertas de disciplinas quando está alocado na grade de horário, isso quer dizer que, não foi possível utiliza algumas das formatações mais adequadas para o design dos quadros. A solução foi manter a formatação padrão da biblioteca.

Outra dificuldade encontrada se refere ao método de adicionar as ofertas na grade de horário, o fato de não permitir a inserção de uma oferta de disciplina na grade sem o conhecimento prévio do ID do Calendário, vez com que criar-se uma regra na qual seria necessário escolher de antemão um professor e/ou uma sala para inserir nas suas respectivas grades de visualização.

Por fim, apesar do esforço para desenvolver um sistema web que seja compatível com todos os navegadores, é recomendável utilizar para o SAGeH o navegador Mozilla Firefox, para evitar possíveis instabilidades na execução em outros browser.

Capítulo 5

Conclusão

5.1 Considerações Finais

Sabemos que existem diversos sistemas orientados a aperfeiçoar a alocação de recursos, mais conhecido como *Timetabling Problem*, vale ressaltar que tais problemas pertencem à classe de problemas NP-Completo, que são amplamente conhecidos no âmbito acadêmico e computacional, pelo seu nível de complexidade, como pode ser visto no Capítulo 2.

O trabalho buscou mostrar como o SAGeH foi concebido, explorando cada etapa do seu desenvolvimento, como a coleta e a análise de requisitos, a construção de diagramas para facilitar a implementação do sistema, as tecnologias utilizada entre outros detalhes. Além disso, demonstrou também a importância que um sistema precisa ter para proporcionar um ambiente mais agradável e interativo, no qual o usuário possa fazer as modificações na grade de horário deixando-a personalizada de acordo com as suas preferências e não menos importante, salientou a necessidade de preocupar-se com as formas de visualização das informações.

Desta forma, o trabalho foi voltado para atender a carência de software especializado em detecção de conflitos, que dê apoio através de uma interface agradável, que tenha preocupação com a integração com o usuário, além de propiciar formas mais adequadas de visualização dos dados e que utilize uma arquitetura baseada em cli-

ente/servidor, todas essas características abrangem as necessidades percebidas para uma melhor solução ao problema estudado.

Entretanto, o trabalho não teve como foco a otimização automática para a alocação de recursos, porém o sistema SAGeH tem em uma das suas funcionalidades, dar suporte a receber arquivos externos para configuração de uma grade de horário, no qual poderá conter uma grade já otimizada, e assim, o usuário poderá utilizar esta ferramenta para fazer as modificações necessárias, tendo como limitação somente a forma de criação deste arquivo, que deverá obedecer de acordo com o Apêndice A.

Uma das principais contribuições deste trabalho foi atender as necessidades do cliente estudado, atingindo de uma forma direta e modificando para melhor os processos de trabalho do Departamento Ciência da Computação. E assim, contribuir de uma forma prática, para o crescimento da Instituição, na qual adquirir os conhecimentos para execução deste trabalho.

5.2 Trabalho Futuro

Muitas são as possibilidades de expansão deste trabalho. Indo do simples melhoramento da forma de como os dados é apresentados, ou acréscimo de outras dimensões de visualização, ou até mesmo a inserção de uma estatística, ou a melhoria do algoritmo para os arquivos de importação, ou criação de um algoritmo para gerar uma grade automática.

Além das modificações mais conservadoras, podemos trabalhar a ideia de integrar todo o sistema com a utilização de NodeJS junto com o banco de dados MongoDB, já que os procedimentos estão no formato JSON e assim conseguir um sistema mais robusto.

Deste modo, ao dar continuidade ao trabalho aproveitaria os estudos sobre o *Timetabling Problem*, conseqüentemente ficaria mais próximo para gerar uma ferramenta mais robusta e completa para atender as necessidades deste problema.

Referências

- [1] BATTISTI, J. *SQL Server 2005: administração & desenvolvimento: curso completo*. Axcel Books, 2005.
- [2] BEZERRA, E. *Princípios de Análise e Projeto de Sistemas com UML*. Elsevier, 2006.
- [3] BORNIA POULSEN, C. J. Desenvolvimento de um modelo para o school timetabling problem baseado na meta-heurística simulated annealing. Tese de Mestrado, Universidade Federal do Rio Grande do Sul, 2012.
- [4] BURKE, E. K., PETROVIC, S., E QU, R. Case-based heuristic selection for timetabling problems. *Journal of Scheduling* 9, 2 (2006), 115–132.
- [5] CERDEIRA-PENA, A., CARPENTE, L., FARINA, A., E SECO, D. New approaches for the school timetabling problem. In *Artificial Intelligence, 2008. MICAI'08. Seventh Mexican International Conference on* (2008), IEEE, pp. 261–267.
- [6] CHEN, C. Top 10 unsolved information visualization problems. *Computer Graphics and Applications, IEEE* 25, 4 (2005), 12–16.
- [7] CORMEN, T. H., LEISERSON, C. E., RIVEST, R. L., STEIN, C., E OTHERS. *Introduction to algorithms*, vol. 3. MIT press Cambridge, 2009.
- [8] CORNE, D., ROSS, P., E FANG, H.-L. *Fast practical evolutionary timetabling*. Springer, 1994.

- [9] CORRÊA M, L. H. Implementação de um sistema de alocação de professores e disciplinas em grades horárias utilizando algoritmos genéticos. Universidade Anhembi Morumbi, 2010.
- [10] DE GANS, O. B. A computer timetabling system for secondary schools in the netherlands. *European Journal of Operational Research* 7, 2 (1981), 175–182.
- [11] DE WERRA, D. Construction of school timetables by flow methods.
- [12] FREITAS, C. M. D. S., CHUBACHI, O. M., LUZZARDI, P. R. G., E CAVA, R. A. Introdução à visualização de informações. *Revista de informática teórica e aplicada. Porto Alegre. Vol. 8, n. 2 (out. 2001), p. 143-158 (2001).*
- [13] GAREY, M. R., E JOHNSON, D. S. *Computers and Intractability: An Introduction to the Theory of NP-completeness*, vol. 29. WH Freeman, San Francisco, 1979.
- [14] GOTLIEB, C. The construction of class-teacher time-tables. In *COMMUNICATIONS OF THE ACM* (1962), vol. 5, ASSOC COMPUTING MACHINERY 1515 BROADWAY, NEW YORK, NY 10036, pp. 312–313.
- [15] GOTTESDIENER, E. Capturing business rules. *Software Development Magazine* 7, 12 (1999).
- [16] GRILLO, F. D. N., E FORTES, R. P. D. M. Aprendendo javascript.
- [17] LAWRIE, N. L. An integer linear programming model of a school timetabling problem. *The Computer Journal* 12, 4 (1969), 307–316.
- [18] NIELSEN, J. *Usability engineering*. Elsevier, 1994.
- [19] ROSS, P., HART, E., E CORNE, D. Genetic algorithms and timetabling. In *Advances in evolutionary computing*. Springer, 2003, pp. 755–771.
- [20] SMITH, K. A., ABRAMSON, D., E DUKE, D. Hopfield neural networks for timetabling: formulations, methods, and comparative results. *Computers & industrial engineering* 44, 2 (2003), 283–305.

-
- [21] TURING, A. M. On computable numbers, with an application to the entscheidungsproblem. *J. of Math* 58 (1936), 345–363.
- [22] VASKEVITCH, D. *Estratégias cliente/servidor*. Berkeley Brasil Editora, 1995.

Apêndice A

Criação do Arquivo de Configuração

Vamos mostrar as regras de como criar seu arquivo de configuração para gerar uma grade personalizada. O arquivo JSON de configuração da grade deverá seguir os seguintes critérios:

- Z (período vigente) = pode ser vazio.
- X (quantidade de período) = deverá ter um valor maior que 0.
- Y (período da disciplina) = deverá ter um valor menor ou igual X.
- N (quantidade de tempo da oferta) = deverá ter no mínimo o valor 1.
- Salvar o arquivo no fomato .JSON.

Código A.1: JSON

```
{ "grade": [
  { "dadosGerais": [
    { "periodoVigente": "Z", "curso": "Qualquer", "qtdPeriodo": "X" } ] },

  { "professor": [
    { "idProf": "1", "nome": "Professor 1" } ] },

  { "disciplina": [
    { "idDisc": "1", "periodo": "Y", "numero": "Qualquer", "qtdTempo": "N",
      "nome": "Disciplina 1" } ] },

  { "sala": [
    { "idSala": "1", "numero": "Qualquer", "capacidade": "Qualquer", "tipo":
      "Qualquer", "local": "Qualquer", "idDepto": "1", "
      caracteristica": "Qualquer" } ] },

  { "horario": [] },

  { "departamento": [
    { "idDepto": "1", "nome": "Qualquer", "sigla": "Qualquer" } ] }
]}
```