UNIVERSIDADE ESTADUAL DE GOIÁS
UNIDADE UNIVERSITÁRIA DE ITABERAÍ
CURSO DE SISTEMAS DE INFORMAÇÃO

Luvânio Lopes Lima
Rodrigo Alves F. Rodrigues

Desenvolvimento de um aplicativo para o Problema da Programação de Horários (PPH) em Instituições de Ensino por meio de uma aplicação WEB utilizando Timetabling.

Itaberaí
2011

LUVÂNIO LOPES LIMA
RODRIGO ALVES F. RODRIGUES

Desenvolvimento de um aplicativo para o Problema da Programação de Horários (PPH) em Instituições de Ensino por meio de uma aplicação WEB utilizando Timetabling.

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Sistemas de Informação da Universidade Estadual de Goiás - Unidade Universitária de Itaberaí, como requisito parcial, para obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof° Ms. Rogério Sousa e Silva

Itaberaí

2011

LUVÂNIO LOPES LIMA
RODRIGO ALVES F. RODRIGUES

Desenvolvimento de um aplicativo para o Problema da Programação de Horários (PPH) em Instituições de Ensino por meio de uma aplicação WEB utilizando Timetabling.

Este trabalho de conclusão de curso foi considerado adequado para obtenção dos créditos na disciplina de trabalho de conclusão de curso, obrigatória para obtenção do título de Bacharel em Sistemas de Informação.

_____

Prof. Ms. Rogério de Sousa e Silva

_____

Profª Ms. Luciana Nishi

_____

Prof° Esp. Justino Porto Neto

“Não deixe o barulho da opinião dos outros abafar sua voz interior. E mais importante, tenha a coragem de seguir seu coração e sua intuição. Eles de alguma forma já sabem o que você realmente quer se tornar. Tudo o mais é secundário”.

(Steve Jobs).

DEDICATÓRIA

Dedicamos este trabalho:

*A **DEUS** por sempre me permitir caminhar ao seu lado por todos os momentos de minha vida e sempre me fortaleceu em minhas batalhas.*

Aos nossos pais que sempre foram e sempre serão o alicerce firme de nossas vidas, que sempre nos amparam em nossas decisões e souberam dizer sim ou não, e sendo um dos "sim" ser: bacharel em Sistemas de Informação.

Aos nossos avós que não cessaram nenhum minuto de interceder pela nossa vitória.

Aos nossos irmãos que com seu jeito de ser, sempre nos deram forças e nos amaram quando nós achávamos que tudo estava perdido e que deram júbilos de alegria com nossas vitórias.

Aos nossos Amigos que tanto nos ajudaram nessa jornada nos dando força, ouvidos e ombros quando necessário nesta grandiosa fase de nossas vidas.

Aos nossos colegas de sala e amigos de sala que foram verdadeiros amigos nas dificuldades e nas alegrias.

As nossas namoradas pela compreensão e força para continuarmos nessa meta.

Luvânio Lopes Lima

Rodrigo Alves F. Rodrigues

AGRADECIMENTO

A aquele que é supremo de todos os nossos agradecimentos por tudo que nos oferece em nossa vida: ***DEUS***

Aos nossos pais que tanto nos incentivaram, para que persistíssemos rumo à conquista da vitoria.

Ao Professor Rogério, que nos direcionou e auxiliou neste trabalho.

A todos os Professores da Unidade Universitária de Itaberaí que nos proporcionaram caminhos de aprendizagem e carga de um verdadeiro profissional.

Aos Colegas de sala pelas batalhas vencidas juntos, pelas partilhas de alegrias e sofrimentos, ou pelo simples fatos de sermos amigos quando necessário.

Por fim, a todos aqueles que contribuíram de forma direta ou indireta para a realização deste sonho.

RESUMO

Problemas de Programação de Horários (PPHs) tem sido amplamente estudados, dada a sua importância prática e teórica. A maioria das variações do problema pertence a uma classe NP-Completo. Em geral, trata-se da alocação de recursos materiais e humanos no espaço e no tempo, visando à otimização de um conjunto de objetivos definidos. Na Programação de Horários de Cursos Universitários, por exemplo, o objetivo é a satisfação do corpo administrativo da instituição de ensino.

Nos últimos anos, as formulações de PPHs propostas pela International Timetabling Competition (ITC) têm sido bastante utilizadas, sendo notável a predominância de métodos baseados em busca local e metaeurísticas entre as abordagens propostas recentemente. Este trabalho tem como objetivo propor algoritmos para o Problema de Programação de Horários Pós-Matrícula da ITC, focando principalmente em métodos heurísticos baseados em Programação Matemática. Entre os modelos de Programação Linear Inteira Mista que propomos para este problema, destaca-se o modelo baseado na Formulação de Representantes Assimétricos para o Problema de Coloração de Grafos. Abordamos a aplicação da heurística de Local Branching e propomos um esquema de resolução por Geração de Colunas, como forma de viabilizar o tratamento dos modelos propostos, uma vez que a complexidade de tais modelos representa um desafio para os programadores em Programação Linear Inteira Mista atualmente disponível.

Como abordado neste trabalho as técnicas aplicadas de Timetabling é de uma complexidade computacional NP-Completo, porém sendo ainda com ajustes, a melhor solução encontrada por nós para a geração de horários, e em especial no que será discorrido neste, o de horários de instituições de ensino.

PALAVRAS-CHAVE: Problemas de Programação de Horários (PPHs), NP-Completo, instituição de educação, Timetabling, Horários e Programação Linear Inteira Mista.

ABSTRACT

Scheduling problems (PPHs) have been widely studied because of their practical and theoretic importance. Most variations of the problem belong to a class known as NP-Complete. In general, it is the allocation of human and material resources in space and time takes aim at optimizing a set of defined objectives. An example would be the scheduling of University courses satisfactory to faculty as well as the academic performance of the students.

In recent years the formulations of timetabling PPHs proposed by the International Timetabling Competition (ITC) have been widely used, with a remarkable predominance in methods based on local searches along with other approaches that have been recently proposed. This work aims to propose algorithms to address the issue of post-acceptance scheduling created by ITC focusing maily on heuristics based on mathematical programming. Among the models of mixed intiger linear programming we use to address this problem, is the model based on Asymmetric Representation Formulation which addresses the issue of graph coloration. Addressing the application of Local Branching heuristic structure by proposing a solution of generating columns as a way to visualize the models, because the complexity of such models is a challenge for development of mixed integer linear programming that is currently available.

As discussed in this paper the techniques applied in timetabling is an NP-Complete computational complexity, but is even with adjustments, the best solution for us to generate schedules, and in particular this is discoursed, the schedule of institutions of education.

KEYWORDS: Scheduling problems (PPHs), NP-Complete, School, timetabling, Schedules and Mixed Integer Linear Programming.

Lista de Ilustrações

Lista de Abreviaturas e Siglas

ITC	*International Timetabling Competition*
PPHs	Problemas de Programação de horários
PHP	***H****ypertext* ***P****reprocessor"*, originalmente ***P****ersonal* ***H****ome* ***P****age*
MSG	Mensagem Enviada pelo Sistema
UML	*Unified Modeling Language* ou Linguagem de Modelagem Unificada
OMT	Técnica de Modelagem de Objeto
OO	Orientado a Objeto
OOSE	Engenharia de Software Orientada a Objeto
MVC	*Model-view-controller*, ou seja, Modelo – Visão – Controle
PL	Programação Linear
CSS	*Cascading Style Sheets*
HTML	*HyperText Markup Language* (Linguagem de Marcação de Hipertexto*)*
HTTP	**Hypertext Transfer Protocol** (Protocolo de Transferência de Hipertexto)
SGBD	Sistema de Gerenciamento de Banco de Dados
W3C	*World Wide Web Consortium*
PDF	*Portable Document Format*

Sumário

1. Introdução

Os problemas de Timetabling, também conhecidos como PPHs (Problemas de Programação de Horários), tem sido uma grande disseminação de pesquisas e estudos. Apesar do PPHs ser um problema cotidiano de inúmeras áreas, cada qual com seus aspectos próprios, acabam por possuírem rotinas de geração de horários únicos, seja qual for a peculiaridade da Solução do PPHs, podendo assim generalizar alguns problemas independente da área.

Seu estudo tem como principal objetivo distribuir de melhor maneira o tempo que um grupo de humanos é alocado em um determinado horário, visando à rapidez e facilidade, a aplicação de um conjunto de regras determinadas. Dessa maneira a identificação de elementos como: *restrições, eventos* e *recursos;* serão elementos básicos de um problema, utilizado estes para a resolução do próprio.

Para o sucesso na organização de um horário é necessário a determinação de um início e de um fim, e definir como serão aproveitados os recursos humanos e materiais da instituição, que na maioria dos casos são limitados.

Um dos principais elementos de uma organização de horário é a restrição, esta se destaca de diferentes maneiras; proibindo ou restringindo eventos de determinados horários. A organização de professores em uma escola, por exemplo, possui a função de melhor distribuir os professores na grade escolar de maneira que satisfaça tanto ao grupo de professores como a instituição de ensino.

Com o avanço dos estudos a forma de como esses horários são divididos, estão sendo aprimorados e passam a utilizar fórmulas que são trabalhadas pela ITC (International Timetabling Competition) que utiliza de métodos levantados em problemas de cada caso. Os seus métodos usam da Programação Matemática para desenvolver algoritmos que possam resolver a PPHs, aprimorando e evoluindo a metodologia de Timetabling ou PPHs.

O PPHs em muitos casos, na construção das tabelas de horários nem sempre geram horários com qualidade, podendo ser um problema desafiador, mesmo que feito por uma pessoa já experiente. Além disso, em alguns casos o problema se generaliza que é tão grande e complexo que um ser humano simplesmente não seria capaz de elaborar em tempo hábil o horário.

Este tem como foco principal pesquisar o Problema de Programação de Horários. Tendo como base de estudos o levantamento das abordagens utilizadas para que este problema e para outros relacionados, afim de ter uma visão panorâmica do problema, tenham assim a capacidade de abordar este problema de vários ângulos e possibilidades.

Buscamos inovar a maneira de associar o PPH há uma dificuldade prejudicial dentro de uma instituição de ensino. Levantando maneiras de otimizar o problema, com combinações de suas complexidades, com isso transformando o processo de Geração de Horários algo fácil e com agilidade de execução e de resultados mais precisos.

2. Fundamentos Teóricos

Para do desenvolvimento do projeto utilizamos a base de estudos passados durante esses anos.

2.1 O Desenvolvimento de Sistemas

Segundo (Dennis, Alan; 2005) “o desenvolvimento de sistemas tem um conjunto similar de quatro fases fundamentais: planejamento, análise, projeto e implementação”.

- A fase planejamento é o processo fundamental para compreender por que um sistema de informações deve ser construído e determinar como a equipe de projeto trabalhará para construí-lo.

- A fase de análise responde perguntas sobre quem usará o sistema o que o sistema fará e onde e quando ele será usado, durante essa fase a equipe de projeto investiga todos os sistemas atuais, identifica as oportunidades de aperfeiçoamento e desenvolve um conceito para o novo sistema.

- A fase de projeto decide como o sistema operará, em termos de infra-estrutura de *hardware, software* e rede a interface do usuário, os formulários e os relatórios que serão usados; e os programas, bancos de dados e arquivos específicos que serão necessários.

- A fase final do desenvolvimento de sistemas é a fase de implementação, quando o sistema é realmente construído. Nessa parte é gerado o código para a resolução do problema, depois o sistema construído é testado para garantir que funcione de acordo com o projeto. A fase de testes é uma das etapas mais críticas, porque os custos dos erros podem ser imensos, grande parte das empresas gasta mais atenção na parte de testes do que escrevendo programas. A instalação é o processo pelo qual o sistema antigo é desativado e o novo ativado, onde é desenvolvido um plano de treinamento para ensinar os usuários como usar o novo sistema e ajudar a gerenciar as alterações causadas por ele.

2.2 Projetos Orientado a Objetos

A fundamentação de um Projeto Orientado a Objeto se baseia em objetos que elaboram funções em seu próprio estado que fornece informações a outros objetos, sendo estas funções algo exclusivo e executado apenas pelo objeto não podendo ser acessado diretamente por outro. Como em todo processo de desenvolvimento este também se divide em análise, projeto e desenvolvimento, porém em uma metodologia Orientados a Objetos, com suas particularidades (HAFEMANN, 2000 apud COLEMAN, 1994).

A parte da Análise nesse processo foca na elaboração de um modelo orientado objetos do domínio das aplicações do futuro sistema. Os objetos da aplicação nesse modelo mostram as entidades e as operações em associação com os problemas a serem solucionados.

Na fase de Projeto da metodologia Orientado a Objeto elabora-se o foco do desenvolvimento, formulando assim o modelo para o software, para uma futura implementação dos requisitos levantados no método de Análise. É a etapa de documentação da estrutura do sistema. Onde também podem ser incrementados novos objetos para a solução total do problema a ser solucionado, modulando os métodos e funções para a implementação.

Na etapa de desenvolvimento, inicia a programação dos métodos e funções levantados e projetados, focalizando usar linguagens que tenha suporte a metodologia Orientados a Objetos, pois estas dão suporte a conceitos do desenvolvimento como classes, objetos, heranças, dentre outras. Seu principal foco é criar a solução requerida pelo Projeto e Analise de forma concreta.

Sendo esta metodologia de desenvolvimento implementada através de classes, com suas respectivas definições e que possuem hierarquias de funcionalidades, podendo assim as propriedades serem direcionadas de uma classe para uma subclasse através de heranças, oferecendo uma maior acessibilidade de inclusão de novas funcionalidades e solução de problemas, pois cada classe tem suas funcionalidades exclusivas. Os objetos também são componentes reusáveis, pois estes são encapsulados independentes dos demais objetos. Podendo assim também ser reaproveitado métodos já utilizados em outras implementações. A tendência indica que o software será construído

previamente e fornecido em componentes e pacotes já desenvolvidos por outros fabricantes, havendo assim um reaproveitamento de códigos (HAFEMANN, 2000 apud SCOLA, 1996).

E uma grande vantagem depois da finalização de um projeto orientado a objeto é a simplificação nas necessidades de mudanças no projeto. O porquê disso é que mudanças de detalhes internos de um objeto não influenciaram em outros objetos, pois este método não causa uma dependência de acoplação entre os objetos.

2.3 Engenharia de Software

Engenharia de Software se define em todos os métodos e aspectos da produção de um software, desde especificação do sistema, implementação e manutenção, após a entrada de operação do sistema (SOMMERVILLE, 1996). Porém a Engenharia não é apenas processos técnicos de desenvolvimento, mas torna-se responsável pelo gerenciamento do projeto, desenvolvimento de ferramentas, teorias e métodos que dêem suporte a construção do software. O conceito de Engenharia de Software traz os princípios de criação, construção, análise e manutenção. Esta veio criar métodos de tratamentos de engenharia para os desenvolvimentos de softwares. Seus métodos são a agilidade, exatidão e precisão no desenvolvimento, que gera uma grande complexidade exigindo mais do usuário.

As principais atividades do processo de engenharia de software são: Processo de software, Planejamento e Gerência de Projetos, Atividades de Desenvolvimento, Atividades de Gerência, Atividades de Garantia da Qualidade e Produto de Software; Gerência da Qualidade; Especificação e Análise de Requisitos; Projetos de Sistema; Implementação e Testes; Entrega e Manutenção (SOMMERVILLE, 1996).

Os padrões da engenharia de software são apenas uma resolução cabível de um problema gerado por um usuário e que pode ser resolvido através de aplicações de modelos que foram documentados e que pode enquadrar nas necessidades de uma solução (PRESSMAN, 2006).

Ao se tratar de métodos padrões da engenharia de software, especificamente

na arquitetura, vale frisar o método MVC (Model-View-Controller) que compreende em separar a parte lógica do software da lógica apresentada, dividindo em parte o desenvolvimento, o teste e a manutenção. O modelo tem caráter de definir e gerenciar as informações e notificar mudanças nos dados. A Visão apresenta o Modelo numa interface a se utilizar o sistema. O Controlador recebe entradas e inicia os processos solicitados. Funcionando em processos, sendo Model o gerenciador dos dados, o View a visão a ser exibida de interface e o Controller formulador de informações e execuções de métodos solicitados (SOMMERVILLE, 1996).

A Engenharia de Software veio moldar as fases de elaboração de um sistema, padronizando os métodos e facilitando sua criação.

2.4 UML

A UML (*Unified Modeling Language*) é uma linguagem de padronização da formulação de diagramas, UML não é um método de desenvolvimento por si só;, construídos na visualização dos softwares em desenvolvimento, porém não é um método de desenvolvimento por si só (HUNT, 2000). Proporcionando assim a modelagem dos conceitos dos sistemas e de todas as futuras interações e funções de cada método. Sendo usada principalmente no Planejamento Orientado a Objeto para esboçar a organização do problema em divisão de classes, proporciona-se assim a análise do sistema sobre diversas perspectivas.

A UML divide em duas categorias, (OMG, 2009) sendo sete tipos de diagramas representam informações de estruturas, e os outros sete representam a parte de geral de comportamento do sistema, sendo quatro deste que representam diferentes aspectos de interação.

Segue os 14 diagramas que compõem esta linguagem de modelagem, conforme os padrões da UML 2.2, que . (OMG, 2009):

- Estrutura

1 – **Diagrama de Classe:** Este modelo de diagrama é um dos mais utilizados, e possui características que exibem os tipos de objetos de um sistema, e apresenta propriedades, funcionalidades e os relacionamentos entre as classes.

2 – **Diagrama de Objetos:** Mostra os objetos e as relações entre eles, sendo

quase como o de classe, porém mais específico. Este mostra a real execução do sistema, como agiriam os objetos e suas relações com a funcionalidade do sistema.

3 – **Diagrama de Componentes:** Tem por objetivo mostrar a parte do sistema já implementado e ressaltar toda a organização dos componentes.

4 – **Diagrama de Implantação:** Apresenta a parte física, onde o sistema será instalado e executado.

5 – **Diagrama de estrutura composta**: descreve a estrutura interna de uma classe e as colaborações que esta estrutura torna possível.

6 – **Diagrama de pacote**: descreve como um sistema é dividido em agrupamentos lógicos mostrando as dependências entre esses agrupamentos.

7 – **Diagrama de Perfil:** opera no nível metamodelo para mostrar os estereótipos como classes, e perfis em pacotes. A relação de extensão (linha sólida com closed, repleto seta) indica qual elemento metamodelo um estereótipo dado está estendendo.

- Comportamento

1 – **Diagrama de Casos de Uso:** Especifica os comportamentos que o Sistema proporcionará no uso dos fatores externos. Neste se descreve toda a seqüencia dos eventos que poderão ser feitos por esses fatores.

2 – **Máquina de Estado:** Mostra os estados, as mudanças e eventos em um objeto ou em parte, do sistema.

3 – **Diagrama de Atividade:** Este é em si uma variação do diagrama de Estado, entretanto com um propósito diferente, representa diagramado todo o fluxo de eventos em um caso de uso.

- Interação

1 – **Diagrama de Sequência:** Este informa os objetos e uma sequência de requisições feitas para outros objetos. Segundo Barros (1998), um diagrama de sequência mostra a colaboração dinâmica entre os vários objetos de um sistema. Sendo através desse, apresentado a sequência de execução dos objetos dentro do mesmo.

2 – **Diagrama de comunicação:** mostra as interações entre os objetos ou partes em termos de mensagens seqüenciado. Eles representam uma combinação de informações extraídas de Seqüência de classe, e utilizar diagramas de caso descrevendo tanto a estrutura estática e comportamento dinâmico de um sistema.

3 – **Diagrama de interação Resumo:** fornece uma visão geral em que os nós representam diagramas de comunicação.

4 – **Diagramas de tempo:** um tipo específico de diagrama de interação, onde o foco é a restrições de tempo.

Estes são os modelos de diagramas oferecidos por este método de estruturação, focando-se na especificação, estruturação, documentação e elaboração de uma visão mais lógica no escopo do desenvolvimento do sistema. A UML é uma padronização nestes métodos de modelagem.

A Linguagem de Modelagem Unificada merece um foco especial, sendo esta desenvolvida para dar suporte na modelagem orientada a objetos. Indiferentemente do método desenvolvido, pois, sem perda de dados, pode substituir notações dos métodos Booch, OMT e OOSE, dentre outros inúmeros (Muller, 1997).

A grande importância dessa ferramenta é a precisão e a facilidade na análise, ao ser elaborado, não sendo necessário o conhecimento de todas as ferramentas de análise, devido o suporte oferecido pela linguagem.

A UML possui padrões de modelagem visual orientada a objetos, sendo de simples usabilidade, com funcionalidades e clareza ao especificar e descrever a grande maioria das funções, relacionamentos e técnicas de desenvolvimento orientado a objetos usados na atualidade (LARMAN, 2000).

2.5 Problemas de Programação de Horários (Timetabling).

A programação de horários é um problema enfrentado em diversas áreas, devido a complexidade de possibilidades que a geração de horários pode apresentar. Podem ser citadas como exemplos: a programação de algum evento esportivo, programação de horários no turno de serviço em empresas, programação no horário de chegada e saída de ônibus em uma rodoviária, etc.

2.5.1.1 Problema de formação de horários escolares

Das inúmeras áreas que projetam horários, enfatiza-se a programação de horários em uma instituição de ensino, devido à grande demanda de esforço para a solução deste, pois essa projeção irá influenciar várias pessoas, como alunos, professores e a administração da escola.

A solução manual deste trabalho requer vários dias, ainda assim, a solução nem sempre é satisfatória sob algum aspecto; por exemplo, um aluno pode estar impedido de cursar duas determinadas disciplinas de seu interesse porque estão marcadas no mesmo horário da semana, no caso deste poder montar sua grade curricular.

O Timetabling conhecido como Problema Programação de Horários, com foco na criação de horários escolares, tem a finalidade de programar encontros entre professores, em um período de tempo previamente fixado, usualmente semanais, de modo a satisfazer um conjunto de retenções que podem ser de vários tipos, levando assim a solução do PPHs de uma instituição de ensino (SCHAERF, 1999).

Destacando instituição de ensino, podemos citar três classes:

1. Programação de horários em escolas, que possui uma programação semanal, não permitindo que o professor não dê aulas seguidas, ou que um tipo de aula seja lecionada por mais de um professor.

2. Programação de horários de curso, que possui uma programação semanal de aulas em um conjunto de cursos universitários, onde diminui a repetição de alunos entre aulas de cursos em comum.

3. Programação de exames, o qual organiza horários de exames em um conjunto de cursos universitários, diminuindo o número desses em cursos semelhantes e aumentando ao máximo o tempo dos exames para os estudantes.

Esse meio de divisão, contudo, não pode ser tomado com definição permanente, pois possuem escolas que dão liberdade aos alunos escolherem as matérias que querem cursar, o que deixar-se-à o problema com característica tanto de programação de horário de uma escola como de programação de horários de cursos em uma universidade.

O problema de abordar uma solução única para todos esses casos é a dificuldade de comparação de seus requisitos, onde não há um modelo único que atenda a todas as situações existentes. Em escolas públicas as suas necessidades podem ser

diferentes de as escolas particulares. Nas universidades os critérios se diferenciam por unidades, onde algumas denominam alguns critérios importantes, outras podem ignorá-los (SCHAERF, 1999).

Devido essas circunstâncias, a criação de um algoritmo único que atenda a todos esses problemas é impossível, mas o desenvolvimento deve ser focado a uma única instituição. Diferentemente de outras áreas, onde um tipo de algoritmo possa atender a diversos casos, a constituição de um algoritmo único que atenda a todas essas áreas educacionais torna-se um desafio (MCCOLLUM, 2007).

No entanto o interessa por PPHs vem crescendo a cada ano, havendo várias competições de programação de horários, que são realizados em paralelo a eventos que falam desse assunto, permitido assim o maior fluxo de idéias para comparação de diferentes abordagens. Dentro dessas competições foi dado o problema de alocar os eventos de um curso, sem burlar as regras das restrições estabelecidas.

As instâncias mencionadas são:

P= conjunto de períodos disponíveis a uma alocação de eventos;

p= um determinado período de um conjunto de períodos;

D= conjunto de dias;

d= um determinado dia de um conjunto de dias;

E= conjunto de eventos a serem alocados aos períodos e salas;

R = conjunto de salas nas quais os eventos ocorrem, possuindo nelas a quantidade de alunos;

F = conjunto de recursos disponibilizados pelas salas e requeridos pelos eventos;

S= conjunto de estudantes que assistem varias combinações de eventos diferentes;

Utilizando dessas instâncias podemos fazer algumas referências, por exemplo, para representar o dia em que ocorre um determinado período, $p \in P$, usamos D (p), e para representar o conjunto de períodos que compõem um dia $d \in D$.

Para representar relacionamentos básicos entre os conjuntos utilizamos:

Capacity (r) = capacidade de uma sala $r \in R$;

Fr = conjunto de recursos disponibilizados por uma sala $r \in R$;

Fe =conjunto de recursos requeridos por um evento $e \in E$;

Es= conjunto de eventos assistidos por um aluno $s \in S$;

Se = conjunto de alunos que assistem um evento $e \in E$;

Para não ter dois sentidos, em cada conjunto Fi mencionado, por exemplo, será explicada a origem de i. Se i ϵ a R, por exemplo, fica fácil distinguir que i é um dos recursos disponibilizados pela sala R.

Sendo que nesse projeto foi escolhido o modelo de Programação Linear, onde sua resolução é representada através de geração de colunas.

2.5.1.2 Pré-processamentos de instâncias

Alguns processos isolados podem ser feitos sobre determinadas instâncias, para se chegar a um melhor resultado. Através de dados básicos, dois tipos de conjunto podem ser atribuídos pelas suas instâncias (LEWIS, 2008);

Re, conjunto de salas de salas apropriados para um evento e ϵ E;

Δ(e), conjunto de eventos que não podem ser alocados a um mesmo período que um evento e ϵ E, também chamado de conjunto adjacência de e.

Para garantir que uma sala é adequada a um evento, necessita-se analisar duas situações: se a capacidade da sala é suficiente para a quantidade de alunos que assistem o evento em questão e se todos os recursos que são requeridos estão disponíveis nesta sala. Construímos assim os conjuntos Re para cada evento e ϵ E, conforme apresentado na equação 1:

Re = {r ϵ R: Capacity(r) ≥ |Se |, Fe contido = Fr}, e ϵ E(1)

Alguns eventos não podem ser colocados no mesmo período que um determinado evento, isso quer dizer que ele faz parte do conjunto adjacência deste. Esta restrição é criada quando existem estudantes em comum, ou que possua uma ocorrência simultânea que o impeça de assistir.

Δ(u) = {v ϵ E: v ≠ u, Sv U Su ≠ 0}, u ϵ E (2)

[1] Equação de consulta de capacidade de sala para tal evento.

[2] Equação de impossibilidades de freqüências em uma mesma aula, repetição de aulas.

2.5.1.3 Restrições fortes

Para se resolver o problema de programação de horário e inserir um evento na tabela de horários, ou seja, atribuir a cada um dos eventos e ϵ E a um período p ϵ P em sala r ϵ R, observa-se um conjunto de restrições fortes (LEWIS, 2008);

NOSTUDENTCLASH: nenhum estudante deve ser obrigado a assistir mais de um evento ao mesmo tempo;

ROOMFITTING: A cada alocação de evento na sala, deve-se ter capacidade suficiente para comportar todos os estudantes que assistem o evento, e assim disponibilizar todos os recursos que são requeridos;

NOROOMCLASH: no máximo um evento alocado em uma sala, a cada período.

Uma tabela onde qualquer dessas restrições seja violada é considerada inviável.

2.5.1.4 Restrições fracas

Há também um conjunto de restrições fracas que devem ser evitadas para que a tabela de horários seja confortável para os seus usuários, os estudantes. No entanto essas restrições podem ser violadas e tornar a tabela inviável. São elas (LEWIS, 2008):

NOENDOFDAYEVENT: nenhum estudante teria que assistir a um evento no último período do dia;

NOEVENTSEQUENCE: nenhum estudante teria que assistir mais que um evento em **sequência;**

NOSIGLEEVENT: nenhum estudante teria que assistir a um único evento no dia.

A escolha das restrições fracas foi feita de forma a ser representada de três classes distintas de restrições. A violação de uma restrição **NOENDOFDAYEVENT** pode ser identificada apenas com uma verificação individual, sem reconhecimento do restante da tabela de horários. Agora uma violação da restrição **NOEVENTSEQUENCE** é verificada durante a construção de uma solução levando em consideração os períodos vizinhos. Por fim, a violação da restrição **NOSIGLEEVENT**

somente pode ser confirmada após a construção da tabela de horários terminada, com todos os eventos atribuídos a períodos.

2.5.2 Avaliação de soluções

Levando em consideração a qualidade das soluções, foi decidido que o critério de avaliação seria unicamente a soma das penalidades associado às violações de restrições fracas. Um algoritmo que não fosse capaz de gerar soluções sem violações restrições fortes dentro do tempo estabelecido não poderia participar da competição, uma vez que as soluções fornecidas seriam prontamente desqualificadas. Uma razão para isso estaria no problema de decidir em como comprar duas soluções com diferentes números de restrições, fracas e fortes (SCHAERF, 1999).

A medida que calcula a melhor solução de programação de horário é:

SOFTCOST: soma de todas as penalidades associadas às violações de restrições fracas.

Considerando que cada violação de restrição fraca, independentemente de sua natureza, é prejudicial à qualidade da tabela de horários, as penalidades são unitárias e o valor de uma solução é simplesmente a quantidade de violações fracas

2.5.3 Abordagem do problema

A maioria das técnicas abordadas eram baseadas na simulação do comportamento humano na solução manual do problema. Dessa forma, a solução parcial é especificada, até que todas as aulas, cursos e exames sejam sequenciados. A estratégia utilizada seria ordenar primeiro os elementos com mais restrições.

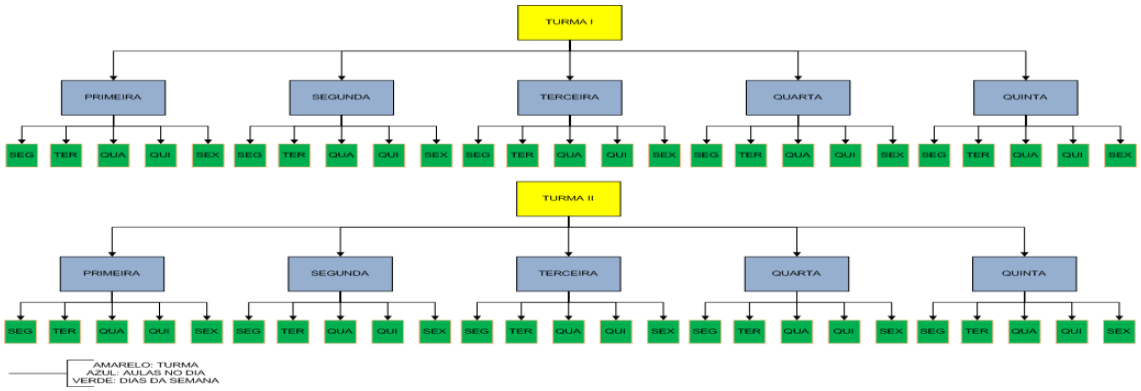

Figura1: Exemplo da Funcionalidade do PPH

A figura anteriormente apresenta o caminho por onde cada professor que se encaixará no horário escolar deve percorrer, sendo que ainda entram suas restrições, a disponibilidades, a quantidade de aulas por semana, as folgas e lembrando que o mesmo professor não pode estar na mesma aula em mais de uma turma no dia.

Mostrando assim as reais funcionalidades manuais geradas para criação do Horário Escolar de uma Instituição.

2.5.4 Particularidades e restrições

O objetivo do trabalho é produzir um método para tratar, especificamente, da classe **Escola** citado, porém com características que se apresentam em escolas públicas brasileiras de ensino básico e médio. Normalmente, as escolas de ensino básico ou médio atendem a um determinado número de turmas, limitado por sua capacidade física.

Geralmente há mais turmas que salas de aulas, mas as escolas trabalham em mais de um período por dia, normalmente nos períodos matutino, vespertino e noturno.

Cada turma possui uma relação de disciplina e certo número de aulas, dependendo do tipo do curso e a série que a turma pertence. A quantidade de aulas em cada turma preenche completamente a semana, ou seja, as turmas tem aulas em todos os horários no período, e todos os dias da semana.

O número de dias da semana e de horários, multiplicados pelo número de turmas da escola, nos mostra o total de aulas ministradas naquele período. Se for considerados os períodos do dia teremos todas as aulas ministradas na escola.

Um conjunto de professores ministram essas aulas, os quais trabalham na escola. Cada professor possui seu próprio número de aulas. Muitas das vezes um professor trabalha em mais de uma escola e em cada uma delas leciona uma quantidade de aula diferente. Uma escola pode ter professores que trabalham em um único período ou em mais de um período.

Vamos considerar neste trabalho, que cada período de aula em uma escola será uma instância independente. Algumas características próprias podem ser enumeradas neste contexto: todas as salas disponíveis na escola são utilizadas; os alunos possuem aulas em todos os horários disponíveis do período; os conflitos de simultaneidade podem ocorrer tanto com professores como com turmas.

Vamos considerar uma intensidade para as restrições. As restrições mais fortes são aquelas que garantem a viabilidade de uma solução, ou seja, referem-se à simultaneidade de aulas de professores e de turmas.

Também existem as restrições fracas, são aquelas onde, cuja não-satisfação, não causa, necessariamente, a inviabilidade da solução (LEWIS, 2008). Vão ser consideradas restrições desse tipo, situações que envolvam preferência de determinado horário do dia ou semana por alguns professores; e horários vagos para professores.

Enumerar a intensidade das restrições é uma tarefa que depende de cada instituição de ensino. Uma necessidade encontrada é a retirada de janelas, pela perca no desempenho educacional da turma que se encontra com essas. Algumas preferências de professores por determinados horários podem estar relacionados a outras atividades por eles exercidas, em situação que a dedicação exclusiva a uma escola não seja obrigatória. Pode se considerar ainda situações em que determinados horários são proibitivos, até por motivos religiosos.

As variantes que, determinam se uma restrição é forte ou fraca, são imensas, o que se aplica a não viabilidade de uma solução (LEWIS, 2008). Assim vamos tratar em nosso algoritmo restrições que podem ser ajustadas conforme o contexto.

2.5.4.1 Características da aplicação

Este estudo foi feito sobre uma variante do problema, onde considerar-se-a um professor associado em uma aula de determinada disciplina dada, para certa turma.

O problema será considerado como formação de agrupamentos. Cada aula possuirá uma sequência binária que represente uma dupla formação, pelo professor e pela turma, referentes àquela aula.

A sequência binária possui duas partes: uma que representa o professor e outra a turma. Na primeira parte, cada posição equivale a um professor e apenas aquele que forma a dupla tem na sua posição o valor 1, todos os demais tem o valor 0. A aula ministrada por esse professor é representada na segunda parte da seqüência.

O objetivo é, então, criar agrupamentos dessas duplas, um para cada horário de aula do período, evitando conflitos, ou seja, colocar as duplas em agrupamentos que não contenham outras duplas com o mesmo professor ou turma.

Como exemplo, podemos ter uma escola que no período matutino atende a

11 turmas.

Cada turma tem seis aulas por dia, cinco dias da semana. Se todas as turmas tiverem aulas em todos os horários, teremos um total de (6 × 5 × 11 = 330) trezentos e trinta aulas na semana para serem alocadas em (6 × 5 = 30) trinta agrupamentos de exatamente onze aulas cada um. Essas aulas poderiam ser ministradas por 15 professores, alguns deles com mais aulas do que os outros.

2.5.4.2 Representação

A representação usada foi:

Seja *m* o número de aulas ministradas numa escola durante a semana em um determinado período.

Seja ainda *k* o número de horários ou agrupamentos de aulas (duplas professor/turma) que se deseja formar.

Os esquemas e estruturas são, então, sequências de *m* símbolos pertencentes ao alfabeto *{0, 1,#}*. Cada esquema ou estrutura tem exatamente *k* posições com símbolo *1*, indicando duplas professor/turma que são sementes para formação de agrupamentos. As posições restantes têm os símbolos *0* ou *#*.

2.5.4.3 Associação

A formação dos agrupamentos foi feita com associações entre as duplas presentes em cada esquema. No entanto, ao invés de associar uma dupla a outra, nesta aplicação a associação foi feita entre a dupla e o agrupamento, representado por todas as duplas já associadas.

Cada agrupamento é também representado por uma sequência binária. Essa é obtida mesclando-se as sequências das duplas já pertencentes ao agrupamento. Trata-se basicamente de uma operação lógica OU (disjunção inclusiva) aplicada sobre todas as sequências das duplas associadas ao agrupamento. Inicialmente a única dupla de um agrupamento é a semente usada para sua formação.

As duplas do esquema ou da estrutura são associadas aos agrupamentos em uma determinada ordem, a qual considera uma precedência onde professores mais

antigos ou com mais aulas na escola podem ter privilégios sobre os demais na elaboração do horário e no atendimento de suas preferências. Se os professores têm a mesma precedência, são primeiro considerados aqueles com maior número de restrições de horários (LEWIS, 2008).

2.5.4.4 População inicial, Seleção e Recombinação

A população inicial foi gerada com 100 esquemas, cada um deles contendo exatamente *k* sementes em posições aleatórias e 20% das demais duplas associadas a algum agrupamento.

A população foi mantida ordenada de acordo com um critério que privilegia esquemas mais próximos de estruturas e com melhor adaptação,

Isto é, menos conflitos dentro dos agrupamentos. O método de seleção utilizado foi do tipo *base-guia* (MCCOLLUM, 2007).

Mantendo o número de sementes no esquema ou estrutura resultante.

2.5.4.5 Funções de avaliação

Para avaliar os esquemas ou estruturas *Si* foram usadas as funções *f* e *g*. No entanto, esse problema envolve restrições de intensidades diferentes, foram usadas três formulações para as funções *f* e *g*.

A primeira formulação envolve apenas as restrições fortes de viabilidade, isto é, considera apenas conflitos de simultaneidade gerados por professores que estariam ministrando mais de uma aula ao mesmo tempo e turmas que teriam mais de uma aula

Ao mesmo tempo. Nesse primeiro caso, as formulações utilizadas foram:

Onde *Cp (Si)* é o *p*-ésimo agrupamento de duplas.

Onde *Conf (.)* representa o número de conflitos no agrupamento.

A segunda formulação foi utilizada para as restrições referentes às preferências dos professores por determinados horários. As formulações utilizadas foram:

() = 2 *i g S número total de duplas*.

() () 2 *i* 2 *i f S = g S - número de duplas com conflito de preferência.*

A terceira formulação envolve conflitos referentes às restrições de janelas nas grades dos professores, isto é, situações em que os professores ficam sem aulas em horários intermediários entre sua primeira e última aula do período em um dia. As formulações utilizadas foram:

() = 3 *i g S número total de duplas.*

() () 3 *i* 3 *i f S = g S - número de janelas.*

Um único limitante superior *gmax* foi utilizado para as três formulações de g. O limitante possui a formulação:

g f k .

Onde: *m* é o número total de duplas professor/turma; *k* é o número de horários ou agrupamentos; e *f* é um fator para garantir o limitante superior.

Para cada uma das três formulações de *f* e *g* foi considerada uma medida da diferença *{g(Si)-f(Si)}* em relação à *g*, dada por:

Para unificar essa medida foi feita uma combinação envolvendo dois pesos, *wpref* e *wjan*, Ambos com valor entre 0 e 1. A formulação usada nessa combinação foi à seguinte:

A expressão foi utilizada tanto no cálculo da chave de ordenação dos indivíduos da população para o processo de seleção quanto para o cálculo do *rank* de cada indivíduo.

2.5.4.6 Mutação

O processo de mutação utilizado sobre estruturas geradas na recombinação ou geradas com a complementação dos esquemas base selecionados teve como principal objetivo a garantia da viabilidade da solução, além de procurar a melhoria da qualidade da solução. O processo de mutação foi feito em três etapas na seguinte ordem (SCHAERF, 1999):

Viabilidade de turmas – esta fase procura remover conflitos em que uma turma tenha mais de uma aula sendo ministrada ao mesmo tempo. O processo varre seqüencialmente todos os agrupamentos, procurando duplas com turmas repetidas. Para cada uma dessas duplas que for encontrada, nos agrupamentos restantes é procurada uma dupla cuja turma não esteja no presente agrupamento, e as duplas são trocadas.

Mutação: viabilidade de turmas.

Viabilidade de professores – esta fase procura remover conflitos em que um professor tem mais de uma aula sendo ministrada ao mesmo tempo. Nesse caso, novamente os agrupamentos são varridos seqüencialmente à procura de duplas com professores repetidos. Para cada uma dessas duplas que for encontrada, todos os outros agrupamentos são sondados à procura de outra dupla, de mesma turma, mas com outro professor. Se essa outra turma for encontrada e seu agrupamento não tiver ocorrência do professor repetido, as duplas são trocadas. O processo é visto no algoritmo Mutação: viabilidade de professores.

Melhoria do atendimento de preferências – esta fase procura remover conflitos referentes a restrições mais fracas envolvendo preferências de professores:

Para cada agrupamento

Para cada dupla com turma repetida

Procurar nos agrupamentos restantes uma dupla

Com turma não presente neste agrupamento

Trocar as duplas de agrupamento

Para cada agrupamento

Enquanto houver duplas com professor repetido e não for atingido um limite de iterações

Para cada dupla com professor repetido

Procurar em outro agrupamento uma dupla com mesma turma e Professor diferente

Se este agrupamento não tem o professor repetido

Trocar as duplas de agrupamento

Determinados horários da semana. Nesse caso as duplas são consideradas ordenadas de acordo com o grau de precedência dos professores sobre seus colegas para formação do horário, no caso de professores com mesma precedência, de acordo com o número de restrições dos professores.

Aqueles com maior precedência e mais restrições são considerados em primeiro lugar. Se a dupla estiver em um agrupamento referente a um horário em que o professor não pretende ministrar aulas, os outros agrupamentos sem restrições do professor são sondados à procura de uma dupla de mesma turma e outro professor. Se for encontrada essa dupla, elas são trocadas (SCHAERF, 1999).

3. Programação Linear

A programação Linear (PL), nos termos matemáticos se conceitua em solucionar o um problema através de restrições apresentadas, que através de possibilidades possa se chegar ao resultado esperado através de funções lineares (DE BELLIS, 2004).

A sua importância se dá devido a flexibilidade na programação linear em alcançar soluções dentro de um aspecto prático na área operacional através de probabilidades.

A programação linear e a base da formulação de conceitos centrais da otimização como:

- Decomposição;
- Dualidade;
- Importância da convexidade e suas generalizações.

Na área de definições geométricas, pode se encontrar na PL as restrições lineares que são a definições de um poliedro convexo, que dá origem ao conjunto dos pontos viáveis. Sendo que tendo o ponto a se alcançar, que seria a função objetivo, tudo se tem possibilidade de alcançar, gerando o ótimo global.

Sendo o objetivo uma função linear a solução pode acontecer em apenas um ponto viável dentre inúmeras possíveis soluções.

Para se alcançar o ótimo global se define três pontos na programação linear:

- Definição do objetivo do problema;
- Definição das variáveis de decisão envolvidas;
- Conhecimento das restrições a que está sujeito o problema.

Sendo que só há duas possibilidades de não se encontrar o ponto ótimo:

- Se as restrições forem contrarias uma da outra (por exemplo, $x \geq 2$ e $x \leq 1$) logo, a região factível é vazia e não pode haver solução ótima, já que não pode haver solução nenhuma.

- Quando o poliedro for ilimitado na direção da função objetivo (por exemplo: maximizar $x_1 + 3\ x_2$ sujeito a $x_1 \geq 0$, $x_2 \geq 0$, $x_1 + x_2 \geq 10$), neste exemplo não existe solução ótima uma vez que soluções arbitrariamente grandes da função objetivo

podem ser construídas, e o problema é dito ilimitado.

Fora estas duas condições o ótimo é sempre encontrado, ou seja, ser encontrado uma solução em cima de um dado problema.

O primeiro algoritmo de programação linear em tempo polinomial no pior caso foi proposto por Leonid Khachiyan em 1979. Foi baseado no método do elipsóide da *nonlinear optimization de Naum Shor*, que é uma generalização do método da elipsóide da otimização convexa de Arkadi Nemirovski, um dos ganhadores do John Von Neumann Theory Prize2003, e D. Yudin. Entretanto, o desempenho prático do algoritmo de Khachiyan é desapontaste (DE BELLIS, 2004).

Portanto, o método simplex é o mais eficiente. Sua grande importância é que ele encoraja a pesquisa dos métodos de pontos interiores. Ao contrário de algoritmo simplex, que apenas evolui ao longo de pontos na fronteira da região factível, métodos de ponto interior podem se mover pelo interior da região factível.

Em 1984, Narendra Karmarkar propôs seu método projetivo, que se tornou o primeiro algoritmo a apresentar um bom desempenho tanto na teoria como na prática: seu pior caso de complexidade é o polimonial e os problemas práticos de experiência mostram que ele é razoavelmente eficiente em comparação com o algoritmo simplex.

Desde o método de Karmarkar, muitos outros métodos de pontos interiores têm sido propostos e analisados.

Um método bastante popular é o Método *Preditor-corretor de Mehrotra*, cuja atuação possui bom desempenho na prática, ainda que pouco se saiba sobre ele na teoria.

A opinião mais recente entre os estudiosos é que a eficiência das boas implementações dos métodos baseados em simplex e dos pontos interiores são similares para a aplicação de rotina no programa linear.

A recomendação do uso de programação linear se da a problemas de grande porte, que em sua maioria utiliza-se de várias possibilidades. Por isso a criação de algoritmos computacionais tem sido a maior preocupação dos pesquisadores da PL. (DE BELLIS, 2004)

A base do Timetabling se estabelece na PL, dando vida a solução dos PPHs como no caso da geração de horários de unidades escolares, como o que esta esboçado neste projeto.

4. Metodologia de desenvolvimento ágil Scrum.

Para resolver de tudo hoje em dia as pessoas buscam por agilidade e eficácia, na informática na questão de desenvolvimento de softwares isso não é diferente. Para isso usam-se metodologias de desenvolvimento ágil. Essas metodologias de desenvolvimento estão sendo usadas por profissionais do mundo inteiro. As metodologias usadas são muitas, mas podemos destacar: Scrum Extreme Programming, Lean Software Development, entre outros (MACHADO 2009).

4.1 Introduções ao Scrum

Nas palavras de Schwaber (2009), a metodologia Scrum não é processo e nem uma técnica é um framework com objetivo de transparecer a eficácia. Utilizando o scrum podemos utilizar diversos processos e técnicas, o Scrum é baseado em uma linguagem interativa e incremental, o que permite prever e controlar os riscos. Para o Scrum a documentação não é o objetivo do projeto e sim apenas a documentação suficiente para que o produto pronto seja entregue o mais rápido possível e de uma resposta do que esta sendo desenvolvido ao cliente. Com isso o cliente pode aprovar ou não a idéia assim se houver algum desagrado às mudanças podem ser feitas rapidamente sendo mais eficaz que seguir com um plano. (Machado, 2009) diz que o Scrum vem para mostrar uma nova visão onde metodologias de desenvolvimento tradicionais são formadas de muita disciplina e organização para que se tenha uma habilidade de ser flexível com a estabilidade. Em (Schwber, 2009, p.4) mostra que o Scrum possui Times Scrum onde seus papeis estão associados, eventos com duração fixa, com regras que vão delimitar a atuação dos diferentes papeis.

4.1.1 Times Scrum

Para que se utilize uma metodologia Time Scrum e preciso ter um trabalho organizado tendo como objetivo a produtividade e flexibilidade, possuindo as responsabilidades divididas entre os membros da equipe (SCHWABER, 2009), e dessa maneira um trabalha interagindo com o outro. O papel do Time Scrum pode ser divido

em: Scrum Master, o Product Owner e o time. A função do Scrum Master é garantir que o processo seja entendido e seguido pelos outros membros, analisando se o Time Scrum esta obedecendo aos valores, princípios e regras do Scrum. O time Scrum não possui gerente, pois ele já e organizado por natureza. O Product Owner tem a função de gerenciar a lista de requisitos que e desenvolvida em todo Times Scrum. Ele também e responsável pelo Time, garantindo que todos sigam a lista de requisitos. E por ultimo trabalho do Time que garantir a entrega do produto dentro do prazo baseando na disciplina e organização trabalhada (SCHWABER, 2009).

4.1.2 Eventos

O Scrum trabalha com Eventos que possuem duração fixa e regular. A duração fixa pode ser bem localizada em: Reuniões para planejamento de entrega dos trabalhos, nas reuniões diária conhecida como Sprint, a qual possui revisão e analisadas a sua Retrospectiva. As prioridades dos requisitos a serem seguidos são traçados através das metas que são estabelecidas em reuniões dessa maneira sabem-se os riscos a serem envolvidos e quando uma nova versão será entregue. O planejamento tem por única função levar a satisfação ao cliente trazendo a ele o retorno do investimento fazendo que o tempo inteiro o projeto exija sujeito a mudanças. Não somente as reuniões diárias que acontece, mas a Sprint significa todos os eventos que possuem duração fixa, dessa maneira ele e controlado pelo Srum Master o que garante que a Sprint não possua mudanças sem planejamento. A Sprint e formada de reuniões de oito horas, na qual e dividida em duas partes de quatro horas, onde na primeira parte se discutir os requisitos que serão trabalhados na Sprint. As decisões sobre o que será produzido ficam por conta do Product Owner. Na segunda parte o produto será discutida pelo Time, a melhor maneira em transformar os requisitos escolhidos em software pronto. Todas as Sprints são corrigidas pelo Time Scrum, nesta reunião discuti-se o sucesso e os problemas da equipe. O trabalho deve estar pronto para o Product Owner,o qual deve identificar os requisitos da Sprint que foram trabalhados e discutir os requisitos que serão trabalhados, agendando a data da próxima entrega (SCHWABER, 2009).

4.1.3 Software Pronto

A principal regra do Scrum é que ao final de toda Sprint deve-se ter um resultado que o usuário possa usar. De maneira que todas as etapas desde a analise aos testes esteja pronta juntamente com o projeto (SCHWABER, 2009).

• Sprint é a parte do processo, onde todas as partes do projeto são defini dadas, levando em consideração o tempo de desenvolvimento de cada etapa, os requisitos, a qualidade do trabalho e o retorno do projeto. Normalmente, serão utilizadas múltiplas Sprints para a evolução incremental do sistema.

• Arquitetura – Nesta fase do projeto as modificações do design da arquitetura do sistema são realizadas. Ela é divida em etapas, são elas: Revisão e a ajustes na arquitetura, identificar as mudanças que devem ser feitas, fazer a análise do domínio, enxugar a arquitetura do sistema e identificar possíveis problemas.

• Planejamento – Definição de um novo release baseado no Backlog do Produto (BP) e estimativa de cronograma e custo. Se for o início de um novo projeto, essa fase consiste num conceito e análise do produto. Se for um projeto em andamento, esta fase é limitada à análise. Os passos dessa fase são: o Desenvolvimento claro e objetivo do BP; o Definição da data de entrega e funcionalidades de uma ou mais Sprints; o Definição do release mais apropriado para o início do desenvolvimento; o Mapeamento e estimativa das atividades a serem incluídas no Backlog do Produto; o Definição do Time Scrum (TS); o Avaliação e controle de riscos envolvidos no projeto; o Avaliação das ferramentas de desenvolvimento e infra-estrutura do projeto; o Estimativa de custos.

Os passos dessa fase são (SCHWABER, 2009):

A Reunião de Planejamento da Sprint com a finalidade de definir as atividades a serem incluídas na iteração corrente. O Reuniões Diárias com o TS para revisar o andamento do projeto; a Revisão e ajustes nos requisitos do projeto; o Sprints iterativos até que se tenha “Software Pronto”. “Sof Nesta fase, é importante que as definições da Sprint, sejam plenamente seguidas pelo TS. e Postgame:

• Fechamento – Entrega e empacotamento do “Software Pronto”.

Preparação do produto desenvolvido para a entrega. Documentações, testes, materiais de desenvolvido treinamento e marketing são artefatos típicos dessa fase.

O Scrum foi criado de maneira simples e aberto, permitindo que características sejam adicionadas sem que a filosofia básica. É possível concluir que

Scrum é uma abordagem baseada na flexibilidade, adaptabilidade e produtividade, com ênfase no gerenciamento do projeto (SCHWABER, 2009).

5. Estudo de Caso

Nessa parte será trabalhado o estudo do ambiente no qual o aplicativo poderá ser utilizado, o problema levantado e suas solução computacional.

5.1. Aplicações Web

Visando antemão um desenvolvimento que esteja atualizado com as tendências tecnológicas, voltamos à criação da solução do aplicativo em uma plataforma Web[3].

Trata-se de um conjunto de ferramentas, no nosso caso adotado o PHP, Java Script, CSS, HTML; que é executado em um servidor de HTTP(Web Host). O desenvolvimento da tecnologia web está relacionado, entre outros fatores, a necessidade de agilidade a atualização e manutenção, mantendo o código-fonte em um mesmo local

Uma Aplicação web também é definida em tudo que se é processado em algum servidor, exemplo: quando você entra em um e-commerce a página que você acessa antes de vir até seu navegador é processada em um computador ligado a internet que retorna o processamento das regras de negócio nele contido. Por isso se chama aplicação e não simplesmente site web (WIKIPEDIA).

5.2 Apresentações do Negócio

A base de estudos deste projeto nasceu da Instituição de Ensino Fundamental, Escola Dimensão, da cidade de Sanclerlândia, refletindo junto a esta a grande necessidade de sanar a dificuldade e a complexidade na geração de horários escolares.

---

[3] Software desenvolvido para aplicações em plataforma Web é um sistema projetado para ter como base de execução um navegador, ou *browser*, na internet ou em redes privativas (MACEDO, 2004).

Levando em conta as inúmeras regras de exceções que se pode encontrar na formulação de um horário escolar, esta criação de horários escolares gera na instituição conflitos operacionais, dificultando o trabalho administrativo da instituição e deixando a desejar ao corpo discente e docente desta.

As regras que podem ser utilizadas são:

- Quantidade de aulas para cada matéria;
- Grande Escolar de cargas horárias, distintas de turma a turma, já estipulados pela secretaria da educação.
- Disponibilidade do professor da instituição.
- Folga obrigatória aos professores, não podendo este exceder sua carga horária.
- Aulas vagas para os docentes.
- Professores com janelas no horário.

São alguns dos pontos que levam a dificuldade na geração de horários escolares, em especial quando este trabalho é feito manualmente, como é o caso da instituição de base para este projeto.

5.3 Modelagens de Negócio

Nesta fase**,** foi fundamentada uma entrevista junto à diretoria, quer por si cuida também da parte administrativa da instituição, para poder entender melhor requisitos básicos a serem elaborados para solução dos problemas da instituição, e assim sanar dúvidas e dificuldades reais encontradas neste problema, para assim a fundamentação e elaboração do aplicativo.

5.4 Casos de Uso de Software

Este diagrama descreve as chamadas e respostas das funcionalidades de uso do aplicativo, mostrando os caminhos e requisições de todo o software. Mostrando as

telas a serem exibidas ao usuário para o uso do programa criado.

5.5 Documentos Visão (Apêndice A)

Foi elaborado como modo do usuário final (o cliente) poder verificar e entender funcionalidades do software, e assim conhecer o sistema de um modo mais especificado, num modo mais explicativo de utilidade deste software. Ofertando assim também ao cliente a maneira que seria desenvolvida esse aplicativo, no caso um e-*business*.

5.6 Glossários de requisitos

Foram elaboradas tabelas para expor as características dos requisitos funcionais e não funcionais levantados para construção desse software. Informando requisito por requisito e todas as suas características. A após foi abordado uma rastreabilidade destes requisitos podendo mapear as funcionalidades e desenvoltura do aplicativo. Foi de ampla abstração, porém de grande dificuldade de elaboração devido à amplitude de métodos utilizados neste projeto.

5.7 Prototipação

Esta etapa foi a de grande desafio para a equipe do projeto, devido a metodologia adotada, o Timetabling, ser considerado de uma complexidade de programação NP-Completo, conseguimos apenas formular os caminhos a serem processados e uma probabilidade de resultado, mais não uma solução exata deste problema. Devido as regras deste estudo, levarem apenas a uma melhor solução que nem sempre é única.

6. Diagramas

Colocaremos uma melhor explicação como uso de diagramas.

6.1 Casos de Uso

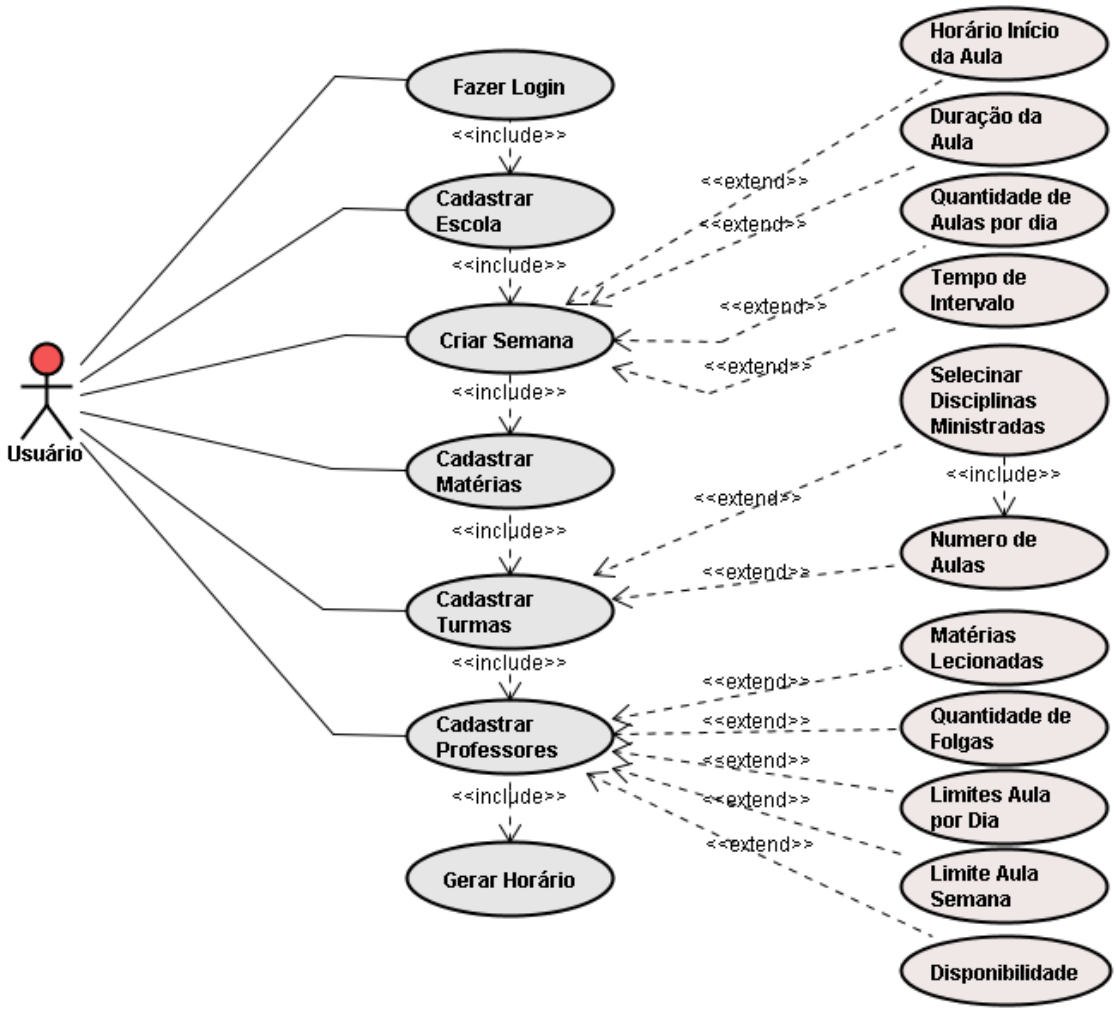

Figura2: Diagrama de Caso de Uso

6.2 Diagrama de Classes

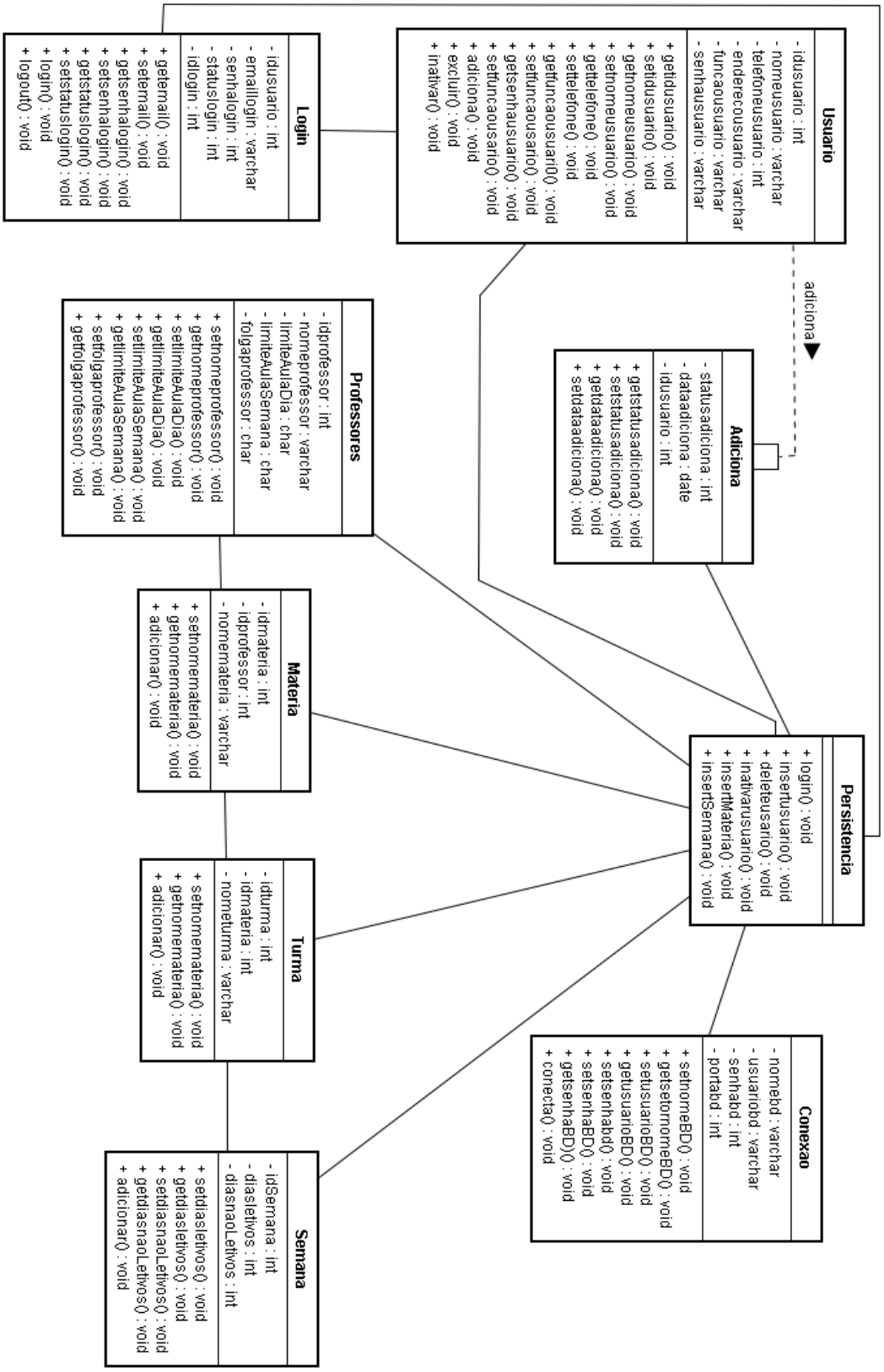

Figura3: Diagrama de Classes

6.3 Diagrama de Componentes

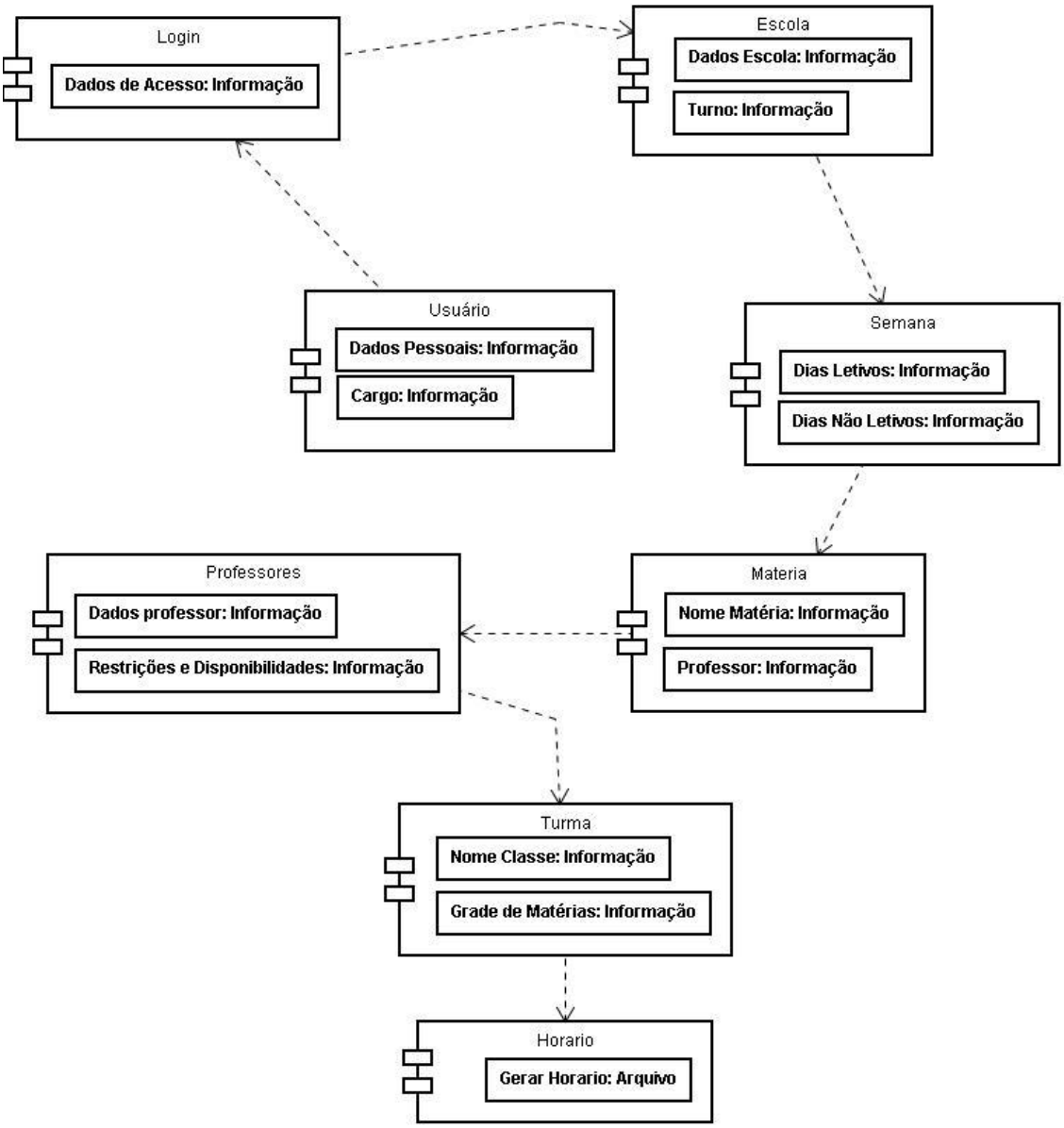

Figura4: Diagrama de Componentes

6.4 Diagrama de Atividades

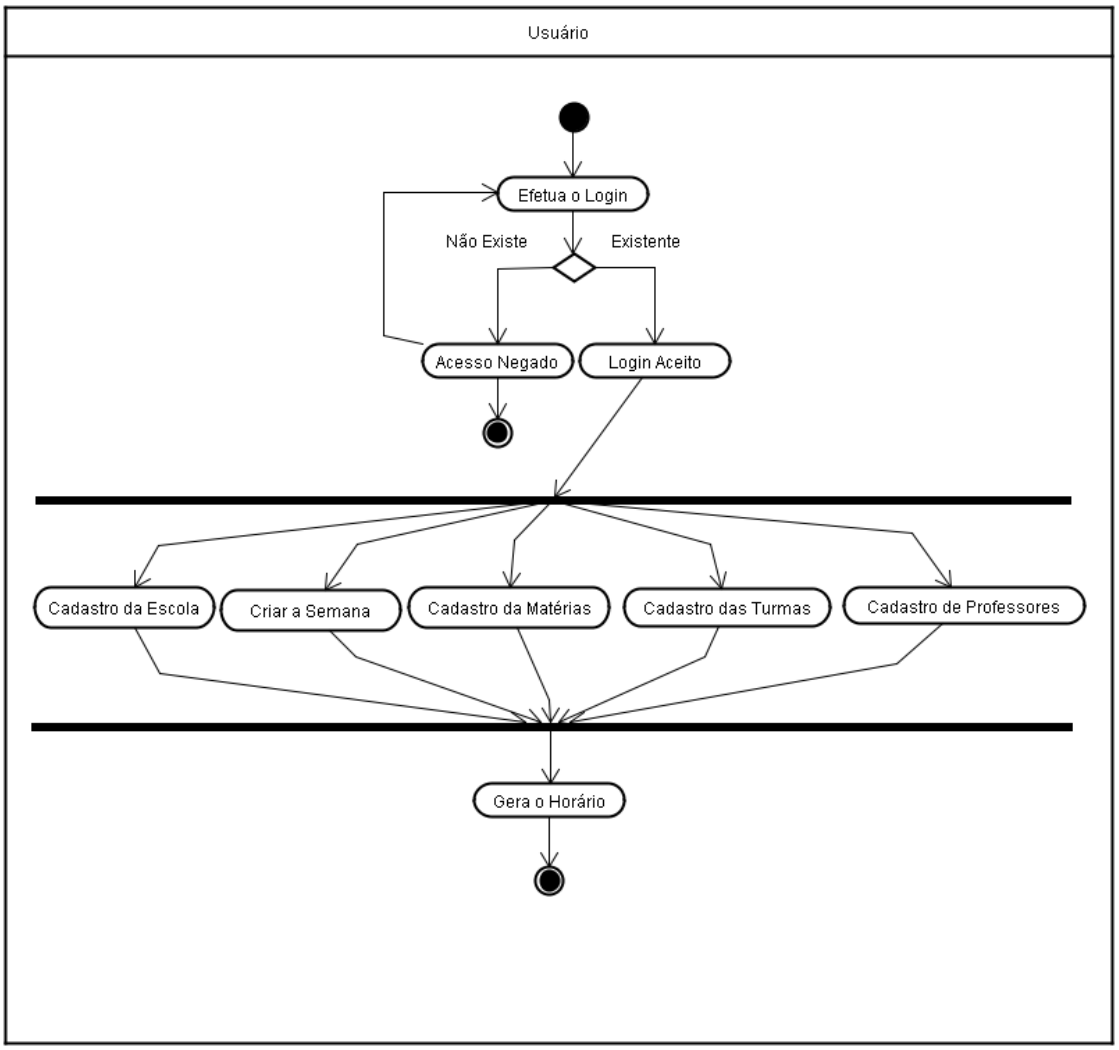

Figura5: Diagrama de Atividades

6.5 Diagrama de Sequência

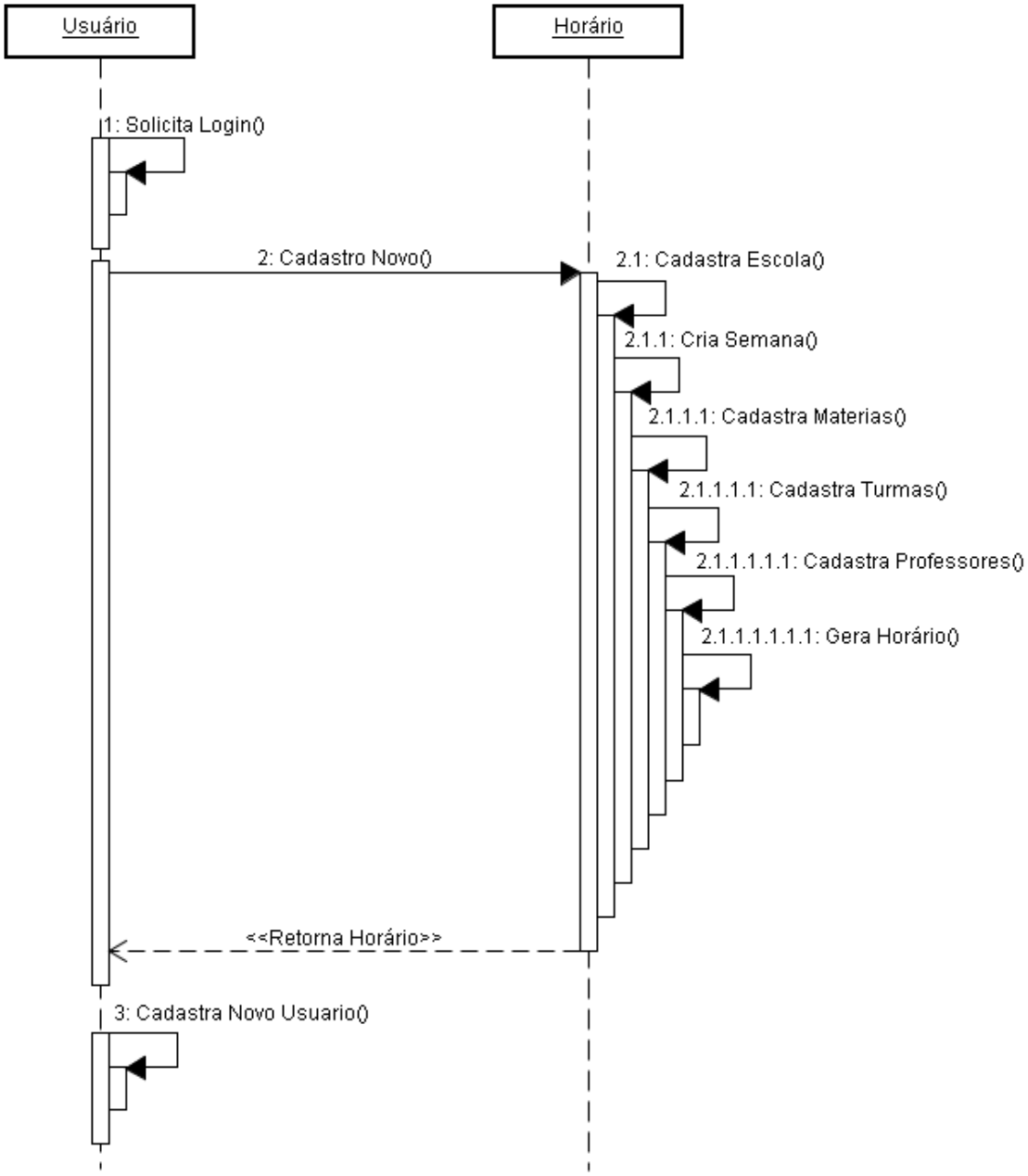

Figura6: Diagrama de Sequência

7. Banco de dados

Nessa parte mostraremos de forma detalhada de como será feito o banco de dados.

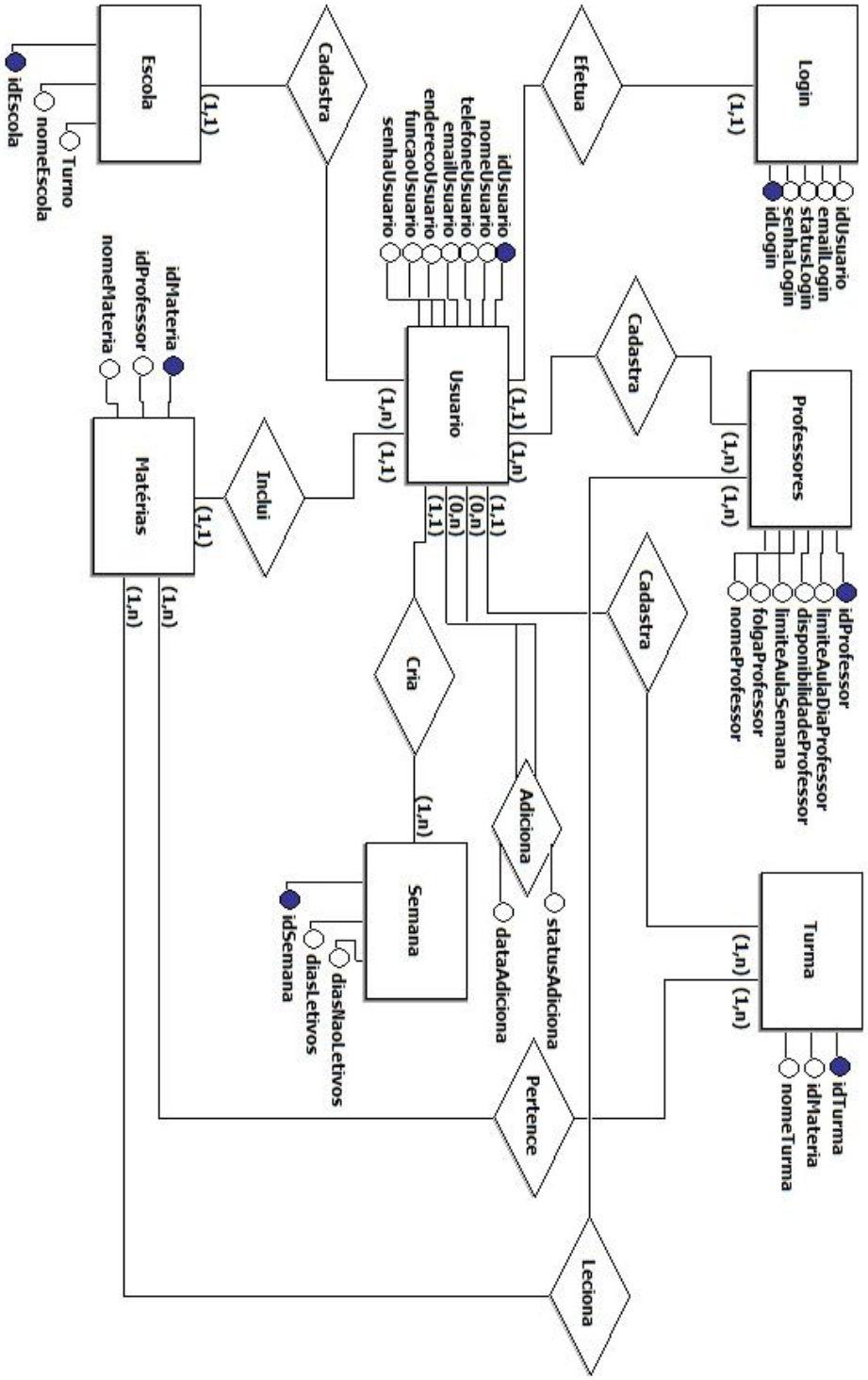

Figura7: Esquema Conceitual

7.2 Dicionários de dados

adiciona

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
idUsuario	**INTEGER**	PK	NN				AI
statusAdiciona	INTEGER		NN			0-em espera, 1 - aceito	
dataAdiciona	DATETIME		NN			data em que se adiciona	

IndexName	IndexType	Columns
PRIMARY	PRIMARY	idUsuario

login

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
idLogin	**INTEGER**	PK	NN				AI
idUsuario	INTEGER		NN				
emailLogin	VARCHAR(100)		NN			Verifica se o e-mail foi cadastrado	
senhaLogin	VARCHAR(20)		NN			Verifica se a senha confere com o e-mail	
statusLogin	INTEGER		NN		1	0-inativo ,1-ativo	

IndexName	IndexType	Columns
PRIMARY	PRIMARY	idLogin
login_FKIndex1	Index	idUsuario

usuario

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default	Comment	AutoInc

				Value		
idUsuario	**INTEGER**	PK	NN			AI
nomeUsuario	VARCHAR(100)		NN			
telefoneUsuario	CHAR(13)					
emaiUsuario	VARCHAR(100)		NN		É obrigatorio pois da acesso ao usuario no login	
enderecoUsuario	VARCHAR(100)		NN			
funcaoUsuario	VARCHAR(100)		NN			
senhaUsuario	VARCHAR(20)		NN		É obrigatorio pois da acesso ao usuario no login	

IndexName	IndexType	Columns
PRIMARY	PRIMARY	idUsuario

Professores

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
idProfessor	**INTEGER**	PK	NN				AI
nomeProfessor	VARCHAR(100)		NN				
limiteAulaDiaProfessor	CHAR(2)		NN			Seleciona a quantidade de aulas por dia.	
limiteAulaSemanaProfessor	CHAR(2)		NN			Seleciona a quantidade de aulas por semana.	
dispobilidadeProfessor	CHAR(2)		NN			Selecionar os dias em que este	

folgaProfessor	CHAR(2)		NN				poderá dar aula Selecionar quantidade de folgas do professor

IndexName	IndexType	Columns
PRIMARY	PRIMARY	idProfessor

Escola

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
idEscola	**INTEGER**	PK	NN				AI
nomeEscola	VARCHAR(100)		NN				
Turno	CHAR(2)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	idEscola

Matérias

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
idMateria	**INTEGER**	PK	NN				AI
idProfessor	INTEGER		NN				
nomeMateria	VARCHAR(50)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	idMateria
materias_FKIndex1	Index	idProfessor

Semana

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
idSemana	**INTEGER**	PK	NN				AI
diasLetivos	CHAR(2)		NN				
diasNaoLetivos	CHAR(2)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	idSemana

Turma

ColumnName	DataType	PrimaryKey	NotNull	Flags	Default Value	Comment	AutoInc
idTurma	**INTEGER**	PK	NN				AI
idMateria	INTEGER		NN				
nomeTurma	VARCHAR(50)		NN				

IndexName	IndexType	Columns
PRIMARY	PRIMARY	IdTurma

7.3 Esquemas Lógicos

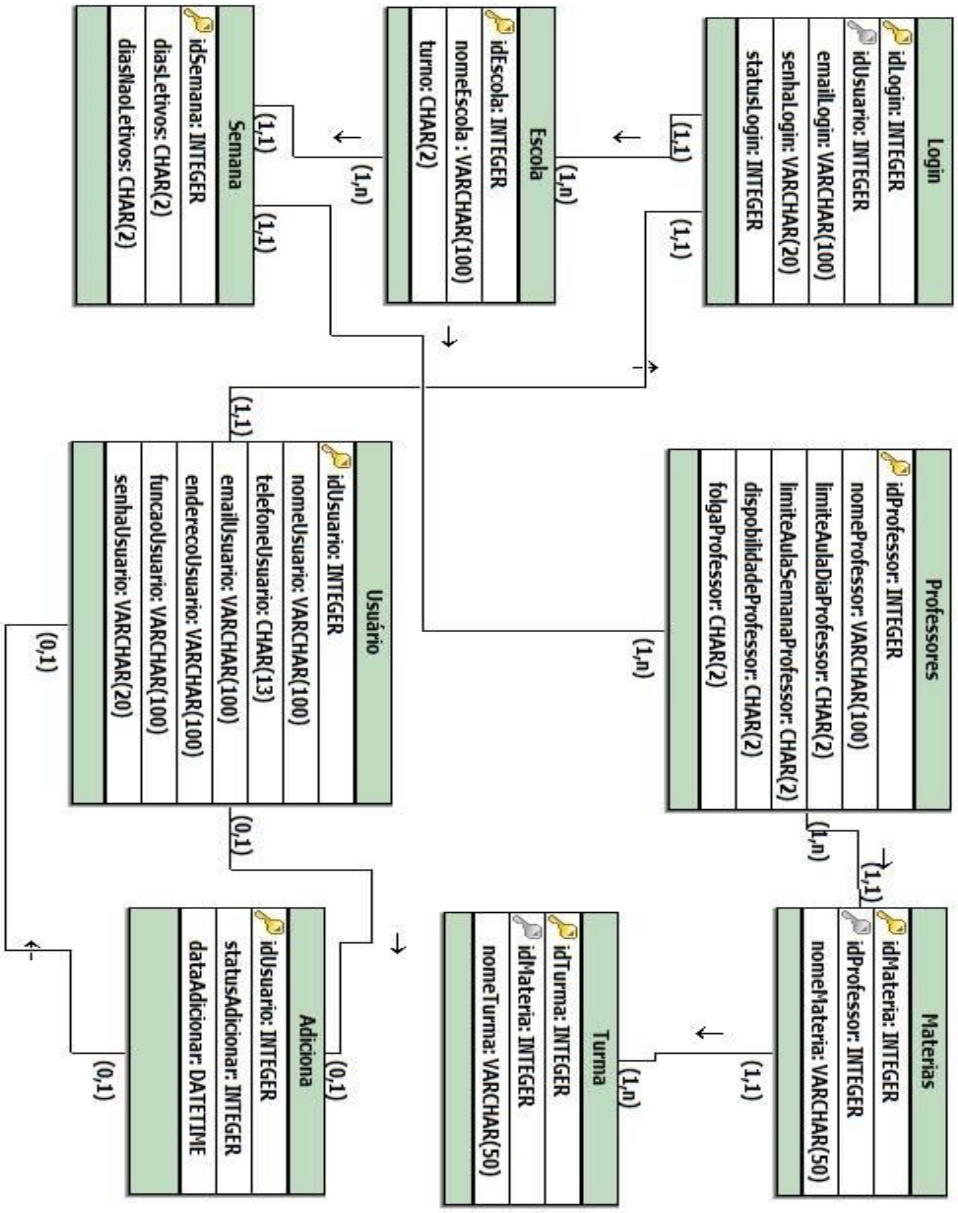

Figura8: Esquema Lógico

8. Prototipação do Sistema

As telas que serão utilizadas no sistema.

8.1 Telas de Login

Esta tela tem como sua principal função controlar o acesso de usuários no sistema, podendo assim gerenciar os responsáveis da criação de certo horário e restringindo o acesso de pessoas não autorizadas ao uso do TimeSchool.

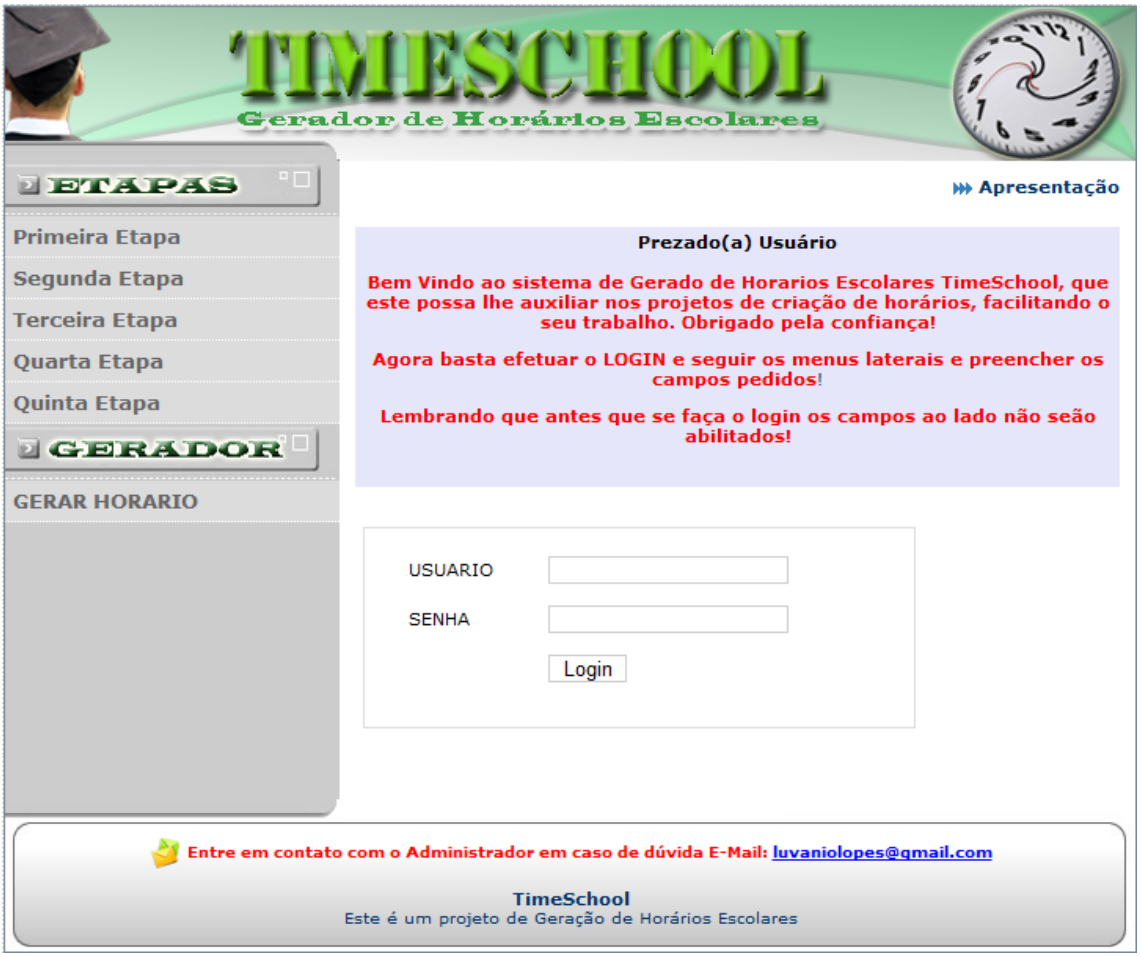

Figura9: Interface Tela Login

8.2 Primeira Etapa

Esta etapa consiste no cadastro do Nome da Escola e a Seleção do Turno de funcionalidade da Instituição de Ensino.

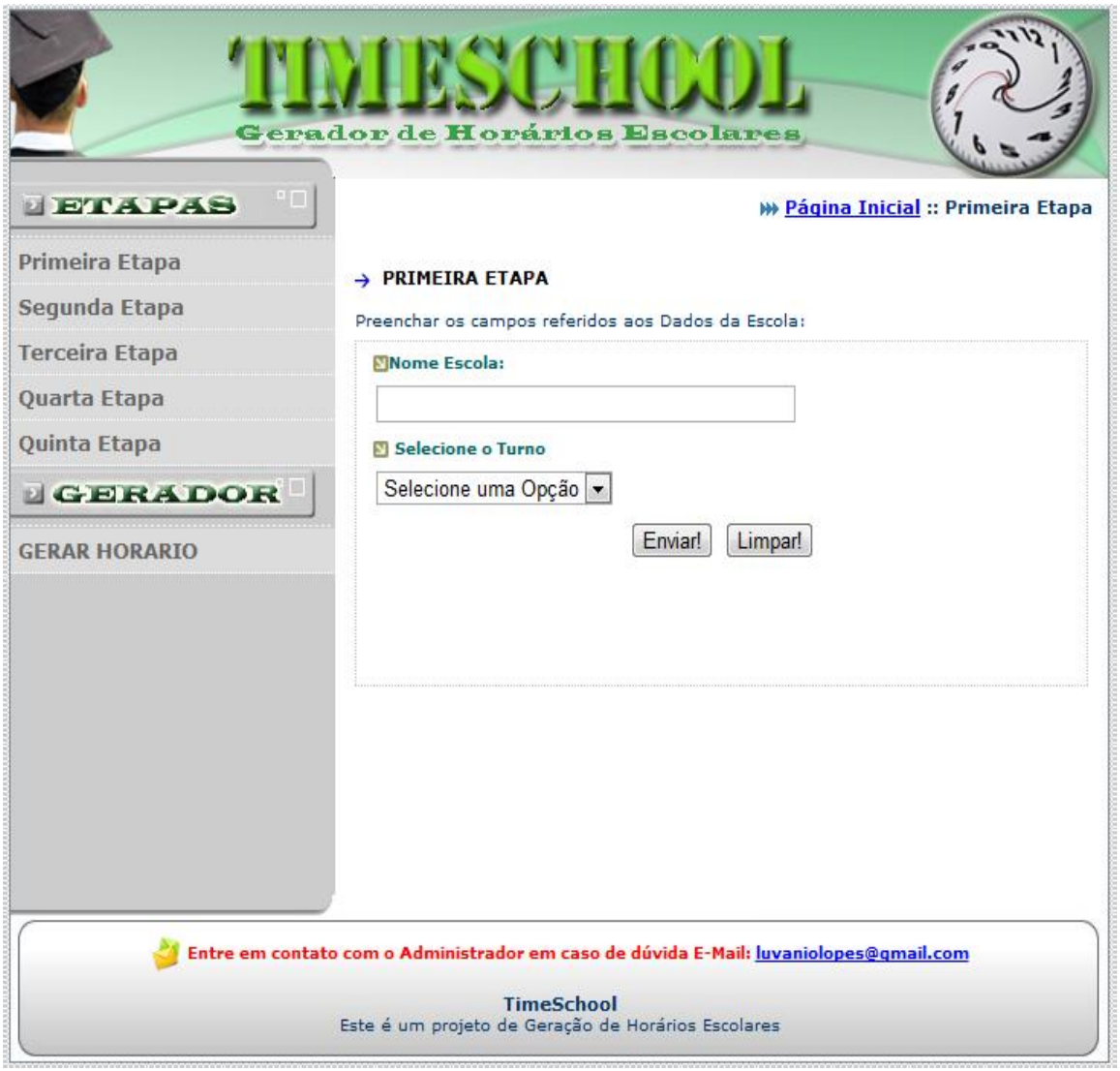

Figura10: Interface Primeira Etapa

8.3 Segunda etapa

Esta etapa tem como principal função selecionar os dias que são letivos durante a semana, definir o horário da primeira aula, o tem de duração de cada aula, tempo de intervalo, quantas aulas os alunos terão por dia.

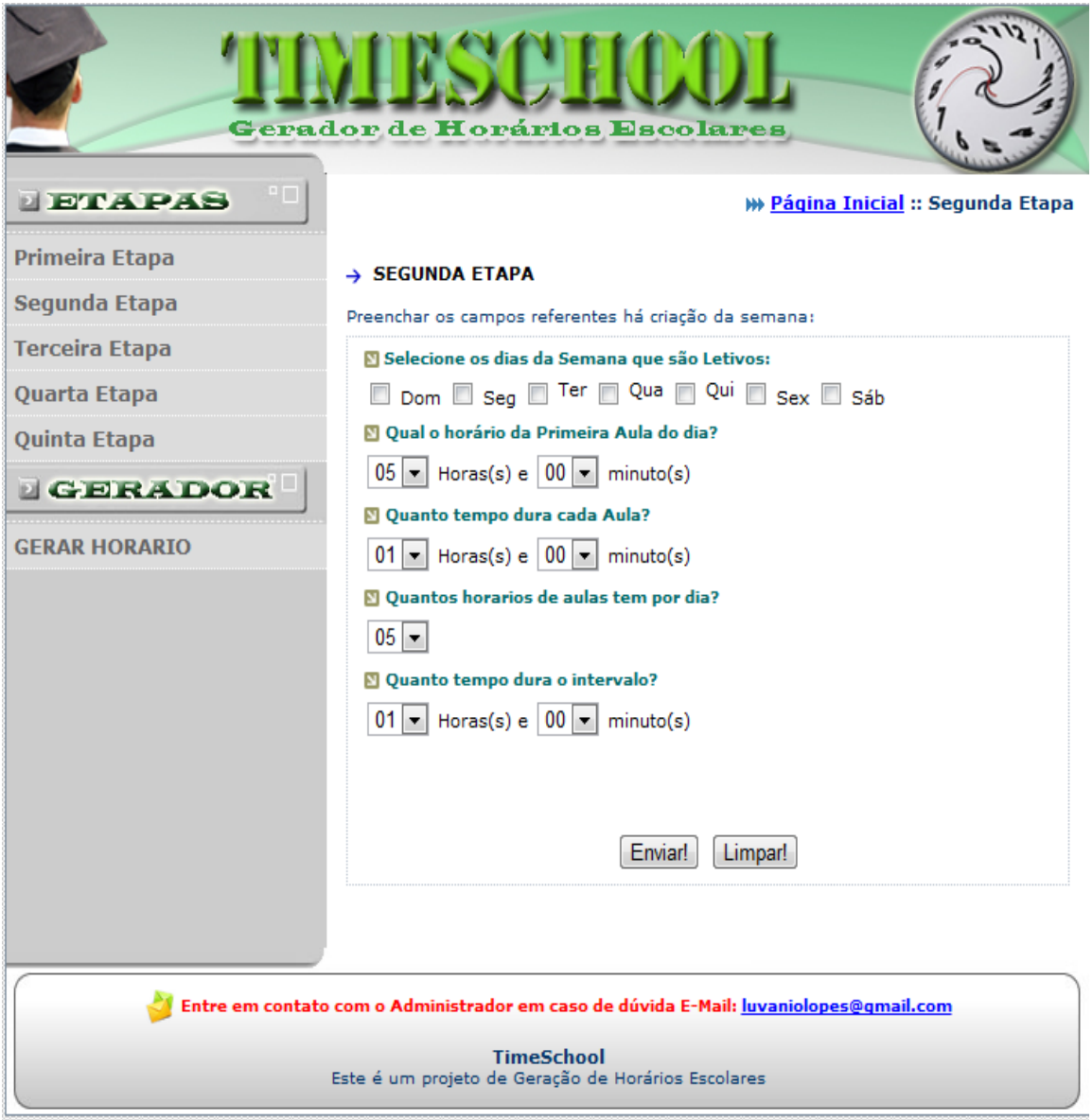

Figura11: Interface Segunda Etapa

8.4 Terceira etapa

Esta etapa tem como funcionalidade o cadastro de todas as disciplinas lecionadas na Instituição de Ensino, com uma funcionalidade de disciplinas já pré-cadastradas.

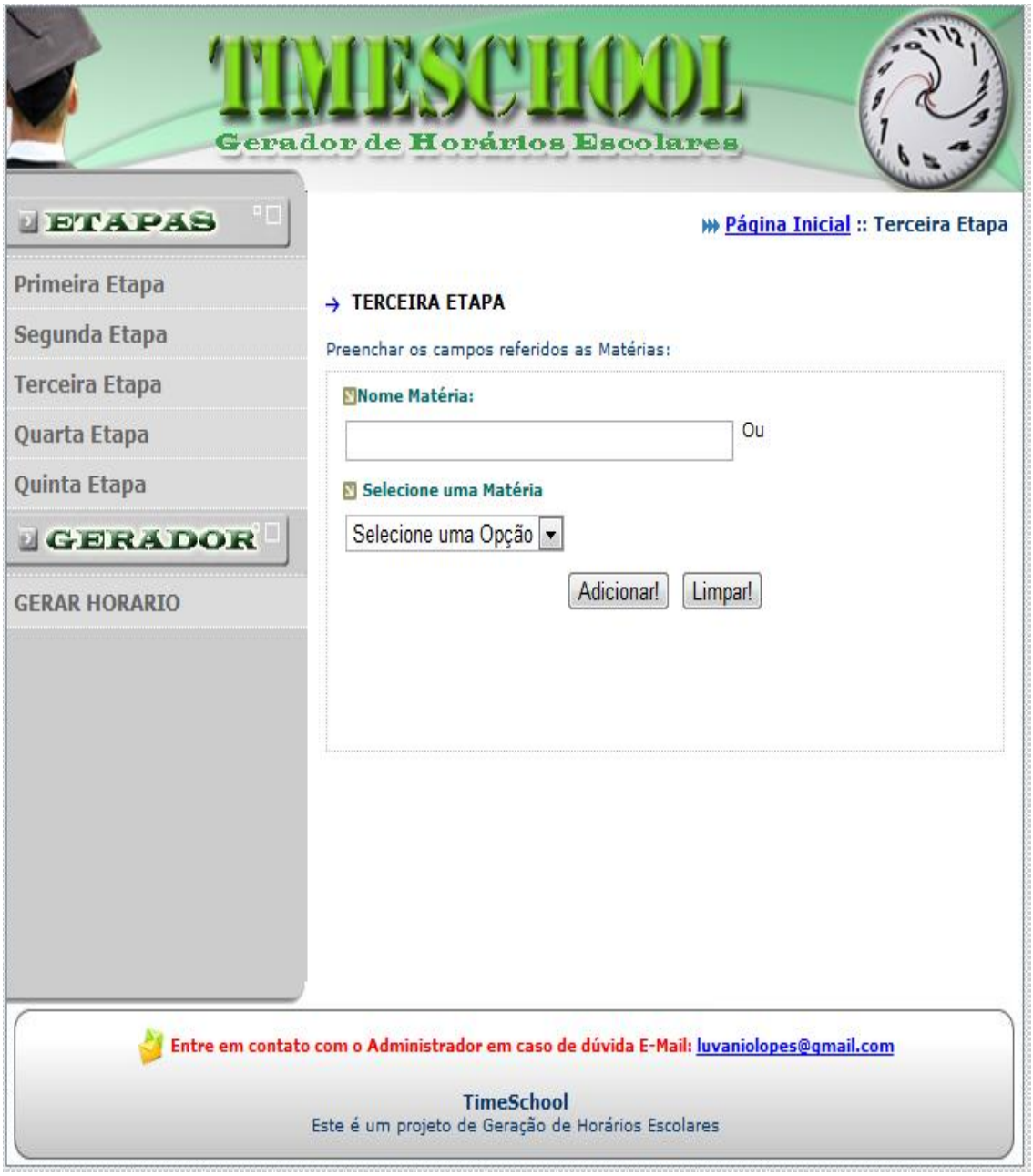

Figura12: Interface Terceira Etapa

8.5 Quarta etapa

Tem como funcionalidade o cadastro das Turmas existentes na Instituição, como os campos de nome da turma e definir as matérias de cada turma com as respectivas quantidades de aulas por semana de cada uma destas.

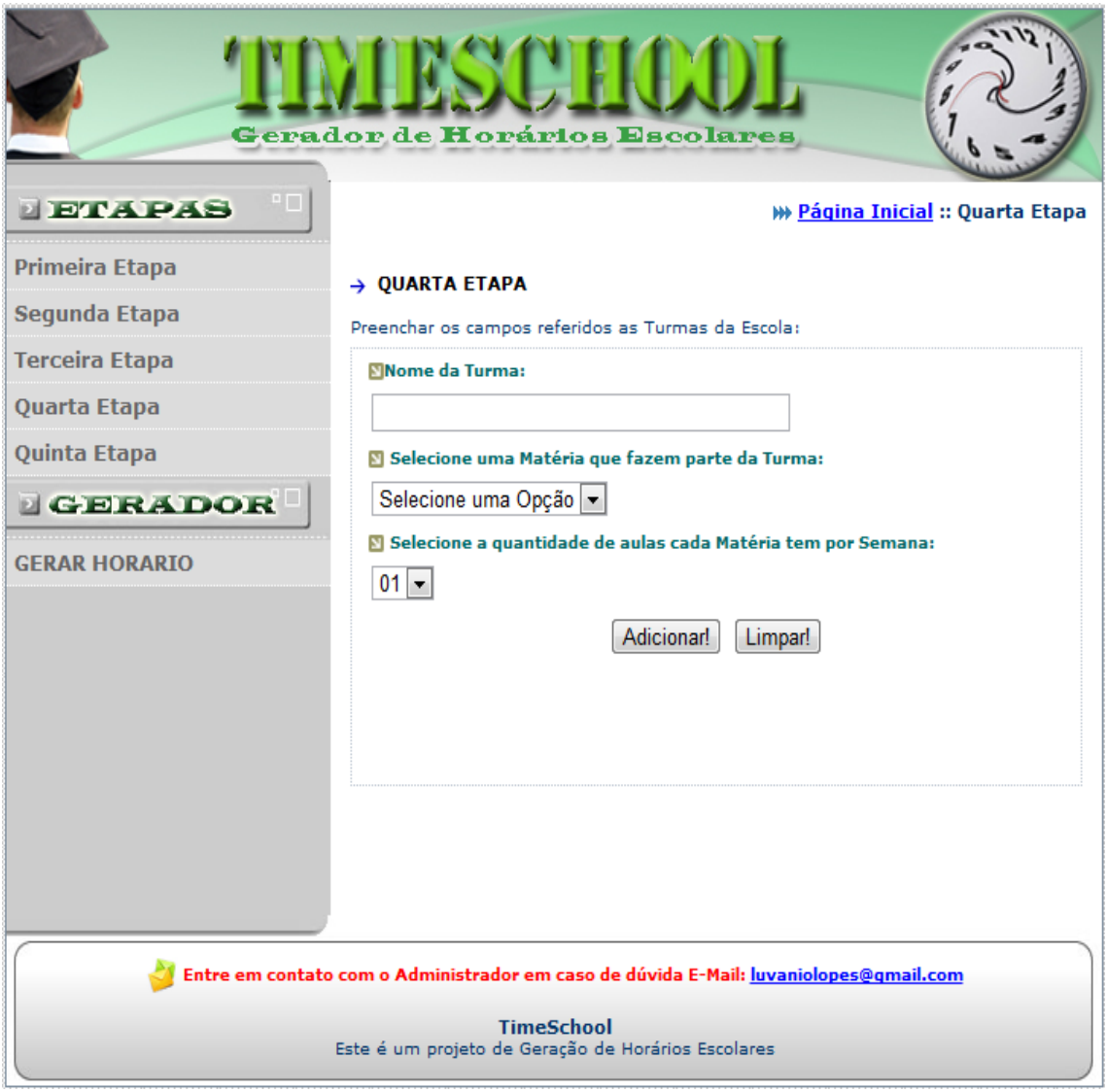

Figura13: Interface Quarta Etapa

8.6 Quinta etapa

Segue a parte que encerra o cadastro, que é o cadastro dos professores, ou seja, os docentes desta instituição, definindo o nome do professor, as disciplinas lecionadas por cada professor, a folgas da semana, quais os dias da semana que o professor poderá ter folga, a quantidade permitida de aula lecionadas por dia, quantidade limite de aulas do professor na semana e os dias disponíveis na semana.

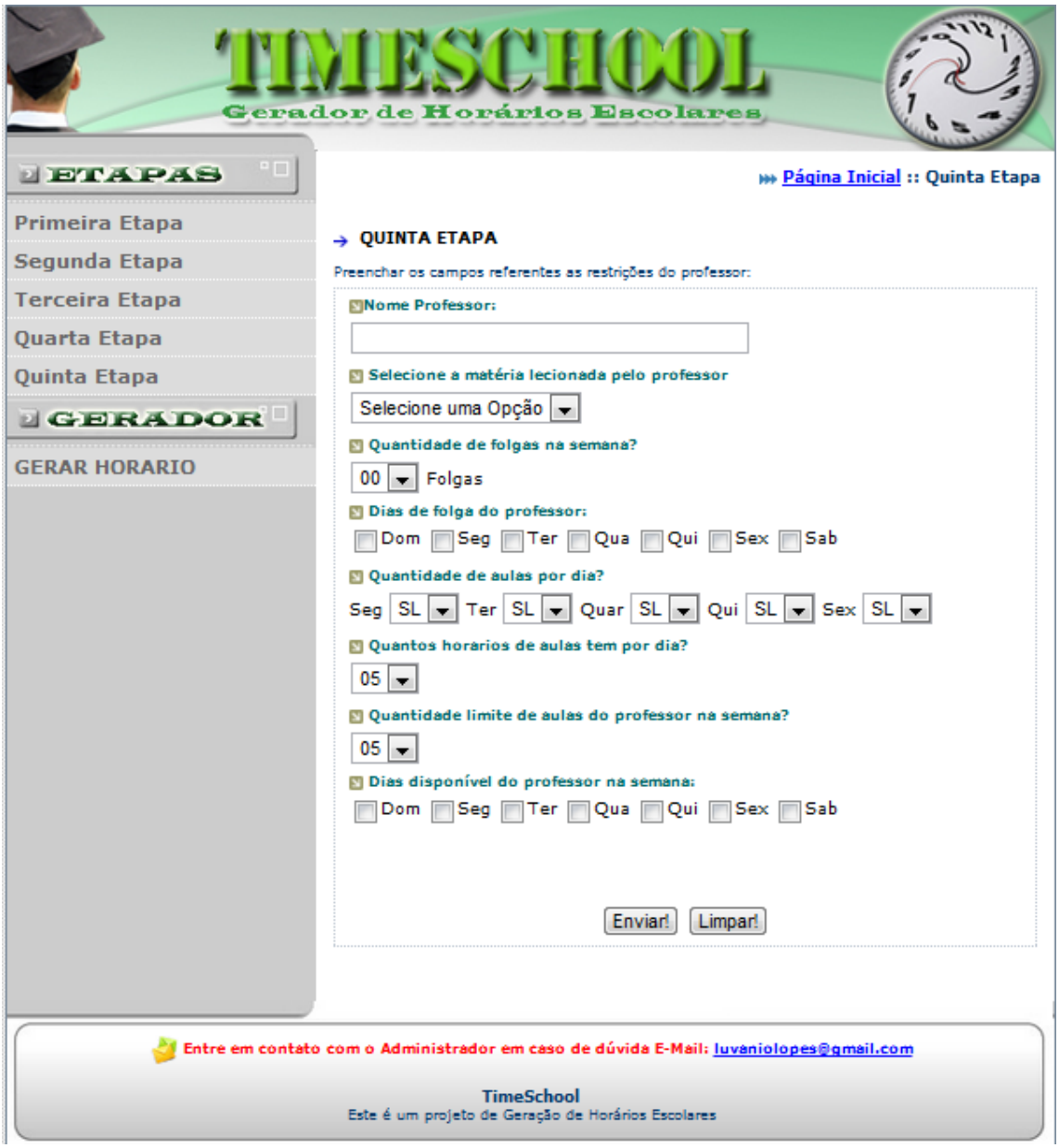

Figura14: Interface Quinta Etapa

8.7 Gerar Horário

Esta etapa é responsável pela geração do horário, pelo levantamento de dados alcançado nos cadastros anteriores.

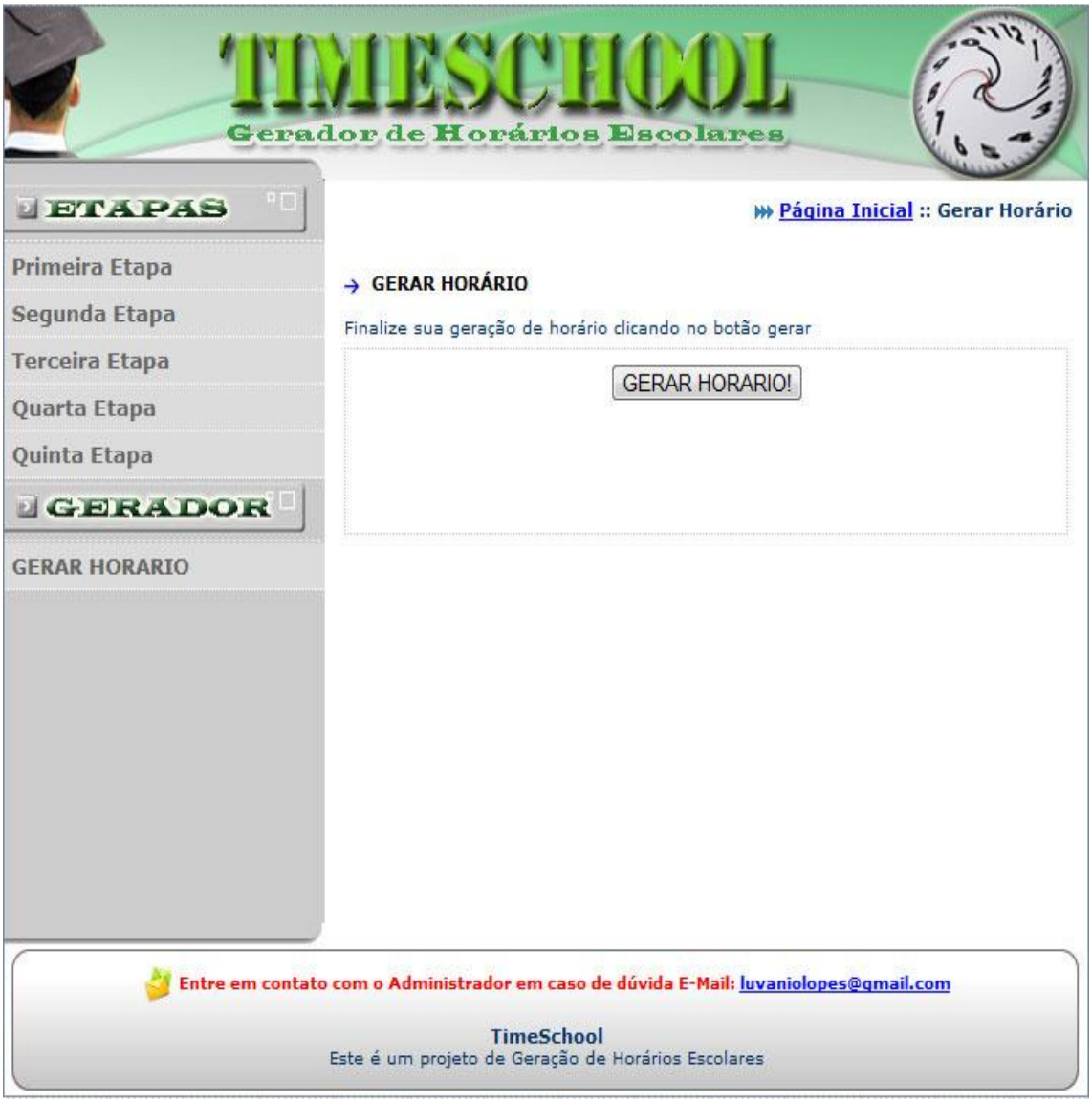

Figura15: Interface Gerar Horário

9. Características do Ambiente Operacional

Colocaremos de forma detalhada de que recursos o sistema esta funcionando.

9.1 Recursos de Hardware

Para o processo cliente da Aplicação (usuário final), será necessário um computador com as seguintes configurações e periféricos para o acesso a internet:

- Processador com velocidade mínima ou superior á 166 MHz;
- Capacidade de 64 Megabytes de memória;
- Capacidade de 04 Gigabytes de armazenamento para arquivos;
- Placa de Rede 100,0/1.000,0 Mbps para comunicação dos pacotes de dados IP na interface web.
- Modem Router para acesso a internet.

9.2 Recursos de Software

Ferramentas de Apoio para documentação do Projeto: Microsoft Word 2007.

Modelagem do Sistema em UML - Construção dos diagramas de: Caso de Uso, Sequencia, Classes, Componentes, Implantação:

- JUDE Comunity 5.2.1 (Model Version: 27)
- NetBeans IDE 6.8 (Build 200912041610)
- The GIMP 2.6.11

Linguagem de Programação: PHP

Construção e Gerenciamento do Banco de Dados: Mysql

10. Resultados

Nesta etapa foi elaborada a validação e a implementação computacional do sistema.

A verificação eficaz da metodologia implantada testada de diferentes situações para que seja feito o estudo de caso, para os testes foi levado em consideração, à preferência de horários por professores, a disponibilidade dos professores para um determinado horário e também levar em consideração a grade dos professores que esta sendo efetuada.

Para que haja a montagem do horário foi necessário fazer as seguintes analises:

1. A menor quantidade de horários disponíveis para a elaboração do horário;
2. A igualdade na distribuição dos horários disponíveis dentro da grade;

Foi analisado também o número de disciplinas ministradas por professor. Sendo assim as disciplinas possuem uma carga horária semanal que por regra deve ser cumprida. Dessa maneira dependendo da disciplina o professor deve disponibilizar essa quantidade de horas para que o horário possa ser formado.

Os professores devem ser distribuídos de maneira igual de modo que todo o horário seja preenchido nas diferentes posições que o horário possa proporcionar, respeitando o numero mínimo de disponibilidade.

A divisão do horário pode ser prejudicada quando os horários não são distribuídos igualmente, onde podendo ocorrer que todo o professores escolham o mesmo horário, assim vai haver a falta de professores nas outras partes do horário. Dessa maneira a formação de horários se torna impossível.

11. Conclusão

O problema de alocação de aulas em horários escolares mostrados grandes importância para instituições e ensino de médio ou grande porte, como as que encontramos em áreas urbanas das cidades brasileiras.

O problema foi considerado como sendo também de formação de agrupamentos. O objetivo foi criar agrupamentos de duplas professor/turma de modo a evitar conflitos de simultaneidade de aulas tanto de professores quanto de turmas. Para se aproximar mais da situação real das escolas, restrições de preferência de horários pelos professores e de janelas em suas grades de aulas foram também consideradas.

Houve preocupação com a diferença de intensidade com que os tipos de restrições são considerados em situações reais. Foi considerada até mesmo a precedência que eventualmente alguns professores possam ter sobre os demais na elaboração dos horários.

A heurística de associação utilizada teve base numa medida de dissimilaridade entre seqüências binárias. No entanto, ao invés de associar dois elementos de uma estrutura ou de um esquema (um deles sendo uma semente de um agrupamento), a associação foi feita entre os elementos e os agrupamentos, estes representados por seqüências binárias obtidas com a mescla das seqüências dos elementos já pertencentes aos agrupamentos.

Foram usadas três formulações para as funções de avaliação f e g. Cada formulação levando em conta um tipo de restrição: viabilidade, preferência de horários e janelas nas grades dos professores. Foram usados pesos para ponderação do uso dessas funções.

Como mutação foi feito um processo dividido em três fases. A primeira visa garantir a viabilidade da solução, evitando os conflitos de simultaneidade tanto para professores quanto para turmas. A segunda e a terceira fase visam à melhoria no atendimento das restrições de menor intensidade, restrições de preferências dos professores por determinados horários e de redução do número de janelas nas grades dos professores.

Os testes foram feitos com instâncias reais, e foram experimentadas combinações dos valores dos pesos de cada tipo de restrição. Os resultados se mostraram satisfatórios porque em todos os testes foram encontradas soluções viáveis, e

o ajuste dos pesos pode ser utilizado para a melhoria do atendimento das restrições de menor intensidade.

Os resultados com as instâncias reais não foram comparados com aqueles obtidos manualmente pela administração das escolares porque alguns dados, como disponibilidade de professores e seus níveis de precedência, não estavam disponíveis e foram apenas inferidos para realização dos testes.

Sem dúvida o problema requer mais estudos, principalmente na análise dos efeitos dos parâmetros do algoritmo sobre os resultados.

A formação do algoritmo para a resolução de timetabling é considerado como um problema NP-Completo, de um problema NP-Árduo, na idéia de algoritmos não-determinísticos.

É importante que ressaltemos que um problema P (polinomial) é dito tratável se ele possui um algoritmo de complexidade que possa ser resolvido, e os problemas do tipo NP são do tipo que não possui uma solução encontrada ou que podem até serem considerados intratáveis.

A partir do desenvolvimento deste software de geração de horário verificamos as vantagens de sua utilização, com os devidos ajustes, podem proporcionar: economia de tempo; resultados mais precisos, podendo deixar os horários formados com mais aproveitamento da instituição de ensino e dos professores. Ficando claro assim que esse meio de aplicação não se trata apenas em automatizar rotinas que podem ser consideradas antigas, mas que os resultados são aqueles que em uma manipulação normal gastaria muito trabalho, necessitando de uma mão-de-obra mas especializada, sem dizer que o risco do horário formado ser um horário não satisfatório. Outro fator que comprova que a utilização do software é eficaz, pois se o horário encontrado ainda não ficar de agrado da instituição e dos professores, é só alimentar o sistema com atributos diferentes que um novo horário poderá ser gerado.

.

12. REFERÊNCIAS BIBLIOGRÁFICAS

1. Aplicação Web: Disponível em: <http://pt.wikipedia.org/wiki/Aplica%C3%A7%C3%A3o_Web> Ultimo acesso em 29 de setembro de 2011.
2. BARROS, P. F. R. UML – Linguagem de Modelagem Unificada.1998. Disponível em: <http://www.eribeiro.com.br/pablo/uml/index.html> Ultimo acesso em 20 de outubro de 2011.
3. COLLEMAN, D. ; ARNOLD, P.; BODOFF, S. Object oriented development – the fusion method . New Jersey, EUA : Prentice.
4. DE BELLIS, Patrícia Maria, Pinho, Alexandre F. e Pamplona, Edson de O. **Definição de Mix de Produção com uso de programação linear e custos empresariais, Anais do XI Congresso Brasileiro de Custos**. Porto Seguro, Bahia, julho de 2004.
5. DENNIS, Alan, Análise de Projeto de Sistemas.Ed. LTC, Rio de Janeiro ; 2005
6. Definição de Orientação a Objeto: Disponível em: <http://pt.wikipedia.org/wiki/Orienta%C3%A7%C3%A3o_a_objetos> Ultimo acesso em 09 de setembro de 2011.
7. Documentação da linguagem PHP: Disponível em: <http://www.php.net> Ultimo acesso em 18 de outubro de 2011.
8. HAFFEMANN, L. J. Protótipo de Gerador de Código Fonte baseado em Diagramas de Sequência, 2000.
9. LARMAN, G. Utilizando UML e Padrões Uma Introdução à Análise e ao Projeto Orientado a Objeto, Ed. Bookman, 2000.
10. LEWIS, R... A survey of metaheuristic – based techniques for university timetabling problems. Or Spectrum, 2008.
11. MACHADO, M.; MEDINA, S. G. SCRUM – Método Ágil: uma mudança cultural na Gestão de Projetos de Desenvolvimento de Software. Revista Científica Intr@ciência. São Paulo: UNIESP, 2009.

12. MCCOLLUM, B... A perspective on bridging the gap between theory and practice in university timetabling, 2007.

13. MULLER, P. A. Instant UM. Canadá: Wrow Press Ltd., 1997.

14. OMG, UML Versão Especificação Superestrutura 2,2 . fevereiro de 2009

15. PRESSMAN, Roger S., Engenharia de Software. 6ª edição. São Paulo ; 2006.

16. SCHAERF, A... A survy of automated timetabling. Artificial Intelligence Review, 1999.

17. SCOLA, A. C. Orientação a objetos uma saída para a crise do software. Revista Developers Magazine, Rio de Janeiro, n 4, p. 16-17, dez 1996.

18. SCHWABER, Ken. Agile Project Management with Scrum. Microsoft Press, 2009

19. SOMMERVILLE, I. Software Engineering (International Computer Science Series). 5ª edição. Reading: Addison-Wesley ; 1996.

13 Documento de Visão (Apêndice A)

Projeto: TimeSchool.

Responsável: Luvânio Lopes

Rodrigo Alves

Autor: Luvânio Lopes

Rodrigo Alves

Históricos de Revisões

Data	Versão	Autor	Descrição
21/03/2011	1.0	Luvânio/Rodrigo	Criação do documento
28/03/2011	1.1	Luvânio/Rodrigo	Definição dos Requisitos e Suas prioridades
11/04/2011	1.1	Luvânio/Rodrigo	Definição de Requisitos Funcionais e Não-Funcionais
25/04/2011	1.2	Luvânio/Rodrigo	Rastreabilidade
17/05/2011	1.3	Luvânio/Rodrigo	Elaboração do Diagrama de Caso de Uso
24/05/2011	1.3	Luvânio/Rodrigo	Criação do Diagrama de Classes
06/06/2011	1.3	Luvânio/Rodrigo	Criação do Diagrama de Componentes
20/06/2011	1.3	Luvânio/Rodrigo	Elaboração do Diagrama de Atividades e Sequêncial
27/06/2011	1.4	Luvânio/Rodrigo	Escopo e Criação do Esquema Conceitual e Lógico do Banco de Dados
17/07/2011	1.4	Luvânio/Rodrigo	Criação do Dicionário de Dados
17/08/2011	1.5	Luvânio/Rodrigo	Prototipação
14/09/2011	1.6	Luvânio/Rodrigo	Desenvolvimento do Software
21/09/2011	1.6	Luvânio/Rodrigo	Desenvolvimento do Software
28/09/2011	1.6	Luvânio/Rodrigo	Desenvolvimento do Software
18/10/2011	1.7	Luvânio/Rodrigo	Resultados Alcançados

1 Introdução

No discorrer deste apêndice deixaremos de uma maneira mais clara as fases de criação deste software, com seus requisitos e funcionalidades, abordando as técnicas usadas e o resultado final.

1.1 A Escola de Campo

Como o desenvolvimento deste software baseia na produção de horários escolares, entrevistamos a diretoria, secretariado e corpo docente da Escola Dimensão, buscando levantar os principais requisitos para a criação deste software de Timetabling no qual será desenvolvida.

1.2 Propósito do Documento

O propósito deste documento é especificar, analisar e definir os principais requisitos e funcionalidades a serem desenvolvidos no sistema TimeSchool.

Esse documento contém uma visão geral dos requisitos mais importantes, sendo base do acordo com o cliente quanto às funcionalidades básicas do sistema e está organizado conforme descrito abaixo:

Seção 1 – Introdução: apresenta o propósito do documento de visão, o público alvo do sistema TimeSchool e as definições, convenções e termos a serem utilizados ao longo do projeto.

Seção 2 – Visão geral do sistema: apresenta uma visão geral do sistema, analisando os problemas encontrados e, baseado nos requisitos, apresenta uma proposta de solução, focando os objetivos a serem atendidos.

Seção 3 – Envolvidos no Sistema: lista os usuários que irão interagir com o sistema e os seus respectivos perfis.

Seção 4 – Requisitos: especifica os requisitos funcionais e não funcionais do sistema.

1.3 Público Alvo

Este documento destina-se principalmente a equipe de desenvolvimento do sistema TimeSchool, mais também, para que o usuário possa compreender do que se trata este projeto e suas funcionalidades.

1.4 Identificação dos Requisitos

A referência aos requisitos é realizada através de um identificador atribuído ao requisito, seguido de uma descrição sucinta do mesmo. Os itens abaixo são informações sobre a composição da identificação dos requisitos e sobre como proceder com a identificação em caso de exclusão de requisitos.

• Para requisitos funcionais será utilizada a seguinte convenção de identificação:

[RF. MÓDULO. SEQÜÊNCIA], onde:

RF = Requisito Funcional;

MÓDULO = Identificador do módulo do sistema do projeto TimeSchool no qual o requisito foi definido;

SEQÜÊNCIA = identificador numérico seqüencial do requisito funcional no módulo.

• Para requisitos não funcionais será utilizada a seguinte convenção de identificação:

[RNF. MÓDULO. SEQÜÊNCIA], onde:

RNF = Requisito Não Funcional;

MÓDULO = Identificador do módulo do sistema do projeto TimeSchool no qual o requisito foi definido;

SEQÜÊNCIA = identificador numérico seqüencial do requisito não funcional no módulo.

• Para restrições será utilizada a seguinte convenção de identificação:

[REST. MÓDULO. SEQÜÊNCIA], onde:

REST = Restrição;

MÓDULO = Identificador do módulo do sistema do projeto TimeSchool no qual a restrição foi definida;

SEQÜÊNCIA = identificador numérico seqüencial da restrição no módulo.

• Os requisitos e restrições devem possuir um identificador único;

• A numeração da SEQÜÊNCIA, tanto para requisitos funcionais como para não funcionais ou restrições, deve iniciar em 001 e ser incrementada em 1 (uma) unidade a cada novo requisito ou restrições;

• A identificação dos requisitos deve aparecer entre colchetes, conforme os exemplos: [RF. 01.001], [RNF. 01.001], [REST.01.001];

• Caso um requisito e/ou restrição seja excluído, a sua numeração não deve ser reaproveitada de forma a evitar inconsistência com referências externas.

1.5 Prioridade dos Requisitos

Para estabelecer a prioridade dos requisitos, que serão definidos na Seção 4 deste documento, foi adotada a escala “essencial”, “importante” e “desejável”, onde essencial é a mais alta prioridade da escala e desejável a mais baixa prioridade.

“Essencial”: Refere-se ao tipo de requisito sem o qual o sistema não pode funcionar, ou seja, são requisitos imprescindíveis, que têm que ser implementados

impreterivelmente.

“Importante”: Refere-se ao tipo de requisito que é necessário para maior qualidade do sistema, contudo ele pode funcionar sem ele ainda que de forma não satisfatória. Dessa forma, os requisitos importantes devem ser implementados, mas, se não forem, o sistema ainda poderá ser implantado e utilizado.

“Desejável”: É o requisito que não compromete as funcionalidades básicas do sistema, isto é, o sistema pode funcionar de forma satisfatória sem ele. Requisitos desejáveis podem ser deixados para versões posteriores do sistema, caso não haja tempo hábil para implementá-los na versão que está sendo especificada.

2 Visões Gerais do Sistema

Colocaremos de forma detalhada o que acontecia antes e o que poderá ser feito depois da implantação do sistema.

2.1 Situação Atual

Observando a complexidade de se formular horários, cientistas da computação buscaram uma maneira ágil e fácil na construção destes, nascendo assim o Timetabling, a partir da programação Linear, também conhecido como PPH, ou seja, Problema de Programação de Horários. Essa complexidade se da pelas inúmeras possibilidades que a geração de um horário possui como disponibilidades das partes envolvidas na formulação destes, entre outros.

Visando focar um meio educacional este projeto adotou métodos que solucionasse o problema de geração de horários para Instituições de Ensino. Já se encontra disponível softwares que automatizam este processo, porém todo o desktop.

Prendendo o usuário ao meio físico.

Buscando métodos de inovação e portabilidade este projeto visa implementar um software de PPH em uma plataforma web, propondo um programa capaz de gerar o horário para o usuário de forma rápida, fácil, sem necessidade de instalação e podendo ser usado de qualquer browser necessitando apenas de uma conexão de internet. Criando assim um novo método de Software para PPH em função de atender Instituições de Ensino.

3 Identificações dos Problemas

Problema	Pessoas Atingidas	Impacto do Problema	Benefícios
Demora para criação de horários	Usuário	O usuário gasta tempo procurando soluções que mantenham todos os requisitos a ele solicitado e conseguir criar o melhor horário.	Ganho no tempo, agilizando assim a criação.
Dificuldade de encontrar soluções para criar e finalizar o horário.	Usuário	Na criação do horário são fornecidos as disponibilidades dos envolvidos para criação e em sua maioria o usuário quando esta produzindo o horário chega em um lugar onde não há lugar para seguir, precisando reiniciar seu trabalho.	Ganho no tempo, pois o usuário não precisar encontrar as melhores possibilidades para criação, o próprio software se encarrega deste.
Dificuldade em instalação de um programa.	Usuário	A dificuldade de um usuário convencional de pode estar instalando um programa nos computadores da instituição.	Facilidade em apenas abrir um browser e pelo site já criar seu horário.
Diminuir a quantidade de janelas de aulas no horário.		Gerar um horário que tenha a menor quantidade possível de aulas vagas ou janelas de aulas.	Por meios de funções o próprio sistema trata essa possibilidade de aulas vagas.

Tabela 2 – Problemas identificados no contexto atual

4 Escopo

O Objetivo do sistema TimeSchool será oferecer ao usuário uma interface de fácil entendimento. Sendo viável o seu uso por uma pessoa com baixo conhecimento na área tecnológica.

Buscando caminhar ainda nos conceitos de facilidade todo o software utilizará de recursos web, sendo assim, não havendo a necessidade de instalação do programa no computador, disponibilizando essa ferramenta 24 horas por dia, necessitando apenas de conexão com a internet.

O Software apresentará formulários simples de forma a garantir maior facilidade ao usuário e maior aproveitamento dos dados no ato da inserção para solicitação de criação dos horários. Esses dados serão redirecionados ao um servidor da instituição e armazenados em um banco de dados. E logo após será disponibilizado em um arquivo em PDF para salvamento ou impressão.

Criar uma interface simples e criativa para levar o usuário a se familiarizar com a metodologia de criação de horários do software e facilitar os caminhos a serem tomados pelo usuário.

A aplicação será executada no sistema operacional do cliente através do browser e se o usuário preferir todas os horários informações serão gravadas em um banco de dados no servidor utilizando-se do SGBD Mysql ou apenas gerar um arquivo em PDF para impressão. O sistema devera contar com um forte esquema de segurança para preservar e manter a integridade dos dados dos usuários, todo o sistema devera ser desenvolvido utilizando-se dos padrões web W3C, e utilizando a arquitetura MVC (Model-View-Controller), e devera também ser desenvolvido utilizando-se da linguagem de programação web PHP, e devera funcionar nos três principais browsers (Internet Explorer 8, Mozilla Firefox e Google Chrome).

5 Premissas e Restrições

Algumas premissas devem ser consideradas

ID	Descrição
[PR.01]	O Software deve ser desenvolvido para a plataforma web, se utilizando das tecnologias PHP, HTML, Javascript, Ajax, XHTML, CSS, Mysql seguindo os padrões estabelecidos pelo W3C e utilizando-se do paradigma de Programação Orientado a Objetos e em camadas no modelo MVC (Model-View-Controller).

Tabela 3 – Premissas e Restrições

6 Stakeholders

Foram identificados os seguintes interessados (stakeholders) para o sistema TimeSchool:

Interessado	Descrição	Interação com o Sistema
Usuário	Fornecimento dos Dados da Instituição de Ensino, tanto sobre elas mesmas, quanto sua função na instituição de ensino.	Cadastramento de dados da Instituição e os dados Pessoais do responsável pelo software, cadastro das informações para gerar o horário como: turno, professor, matéria, disponibilidade, quantidade de aula por matéria, quantidade de aulas na semana, quantidade de aulas no dia.

Tabela 4 – Stakeholders do sistema

7 Necessidades

ID	Descrição
[NEC. 01]	O Sistema necessita de autenticação de usuário (Login).
[NEC. 02]	O Sistema necessita de cadastramento de usuário informando seus dados pessoais.
[NEC. 03]	O Sistema necessita de cadastramento da Instituição de Ensino no qual o usuário deseja executar suas funções.
[NEC. 04]	O Sistema necessita do cadastro dos turnos de funcionamento da instituição.
[NEC. 05]	O Sistema necessita do cadastro de turmas existentes.
[NEC. 06]	O sistema necessita do cadastro de matérias.
[NEC. 07]	O Sistema necessita do cadastro dos professores, com suas disponibilidades e as matérias lecionadas.
[NEC. 08]	O Sistema necessita da inclusão da quantidade de horas/aulas semanais e diárias.

Tabela 5 – Necessidades

8 Requisitos

Nesta seção serão descritos os requisitos funcionais, não funcionais e restrições referentes ao sistema TimeSchool. Cada requisito funcional pode representar uma ou mais funcionalidades do sistema. Esta seção do documento não tem como objetivo descrever detalhes técnicos das funcionalidades, ou seja, serão apresentadas apenas as funcionalidades em sua essência.

8.1 Requisitos Funcionais

ID	Descrição	Status	Prioridade
[RF. 01. 001]	Validação de Login	Aprovado	Essencial
Detalhamento:	Para que usuário possa utilizar de quaisquer recursos do software TimeSchool, será necessário informar o usuário e senha, a qual será validada no servidor. Caso a validação seja positiva o software terá sua interface de aplicações liberada, caso contrário a tela de login será exibida novamente com a notificação de dados incorretos.		
Restrições:	Deve-se Cadastrar no primeiro momento a senha e o usuário para a validação que será feita no próprio servidor e deverá haver conexão com a internet para poder submeter os valores dos campos ao servidor.		

ID	Descrição	Status	Prioridade
[RF. 01. 002]	Cadastro de usuário	Aprovado	Essencial
Detalhamento:	O Cadastro de usuários é muito importante para que o software funcione corretamente, pois somente com essas informações já cadastradas os eventos poderão acontecer. Segue as informações que deverão ser preenchida pelo stakeholder: Nome do usuário, Email, Senha, Telefone, Nome da Instituição de Ensino, Função do Usuário.		
Restrições:	Deve-se cadastrar no inicio o mínimo de informações referentes a usuários, pois os demais eventos só serão possíveis com essas informações já cadastradas.		

ID	Descrição	Status	Prioridade
[RF. 01. 003]	Escolher Turno	Aprovado	Essencial
Detalhamento:	É necessário escolher o Turno de que se tratará o horário a ser criado.		
Restrições:	Esse requisito tem como restrição para a sua funcionalidade o login rf. 01.001 pois este só se disponibiliza após a validação do login.		

ID	Descrição	Status	Prioridade
[RF. 01. 004]	Criar a quantidade de aulas semanais.	Aprovado	Essencial
Detalhamento:	O sistema disponibilizará que o usuário já logado, informe quais dias de semana haverá aulas. Ex: Segunda e Terça.		
Restrições:	Esse requisito tem como restrição para a sua funcionalidade o acontecimento prévio dos requisitos rf.01.001 e rf.01.003		

ID	Descrição	Status	Prioridade
[RF. 01. 005]	Cadastrar a quantidade de aulas diárias	Aprovado	Essencial
Detalhamento:	É necessário que usuário informe a quantidade de aulas que a instituição tem por dia e informar também o tempo de intervalo.		
Restrições:	Esse requisito tem como restrição para a sua funcionalidade o cadastramento prévio de rf.01.004		

ID	Descrição	Status	Prioridade
[RF. 01. 006]	Incluir as turmas	Aprovado	Importante
Detalhamento:	O usuário incluirá as turmas referentes ao turno cadastrado.		
Restrições:	Esse requisito tem como restrição para a sua funcionalidade o acontecimento prévio dos requisitos rf.01.001, rf.01.003.		

ID	Descrição	Status	Prioridade
RF. 01. 007]	Cadastrar as matérias	Aprovado	Importante
Detalhamento:	O usuário ira cadastrar todas as matérias, quantidades de aula de cada matéria semanal.		
Restrições:	Esse requisito tem como restrição para a sua funcionalidade o acontecimento prévio dos requisitos rf.01.001 e rf.01.003.		

ID	Descrição	Status	Prioridade
[RF. 01. 008]	Cadastrar os Professores	Aprovado	Essencial
Detalhamento:	Haverá um cadastro onde o usuário cadastrará o professor, juntamente com sua carga horária de aulas, disponibilidades e o vinculo do professor com a matéria que este leciona.		
Restrições:	Esse requisito tem como restrição para a sua funcionalidade o acontecimento prévio dos requisitos rf.01.007.		

ID	Descrição	Status	Prioridade
[RF. 01. 09]	Gerar Horário	Aprovado	Essencial
Detalhamento:	O usuário após ter cadastro todos os subitens anteriores será informado que sua grade já pode ser gerada, informando-lhe os dados cadastros, disponibilizando a ele editar os arquivos ou finalizar e gerar o horário com os dados informados.		
Restrições:	Esse requisito tem como restrição para a sua funcionalidade o acontecimento prévio de todos os requisitos que o antecede.		

8.2 Requisitos não Funcionais

- **Desempenho**

ID	Descrição
[RNF.01.001]	A aplicação deverá emitir alertas de erros ao usuário. Serão informados erros de conexão, de dados incorretos e de falhas de aplicação.

- **Segurança**

ID	Descrição
[RNF.01.002]	A aplicação deve garantir segurança aos dados dos usuários, para que nada seja visualizado e nem disponibilizado a gerar o horário sem ter efetuado a validação de login e que somente os usuários donos das respectivas informações possam ter a possibilidade de editá-los. Garantir segurança contra “sql injection” e “php injection” e outras formas de ataques.

- **Usabilidade**

ID	Descrição
[RNF.01.003]	O sistema devera apresentar interfaces de fácil entendimento e com dicas que possam explicar determinados campos ou atividades.

- **Disponibilidade**

ID	Descrição
[RNF.01.004]	A aplicação deve estar disponível 24 horas por dia, bastando haver apenas a necessidade do usuário estar conectado a internet.

- **Tecnologias envolvidas**

ID	Descrição
[RNF.01.005]	O sistema será uma aplicação web, então algumas tecnologias se fazem necessárias como: HTML, CSS, Javascript, XML e uma linguagem de programação, neste caso, PHP e banco de dados Mysql. A aplicação deve utilizar os padrões de construção de sites estabelecidos pelo W3C (World Wide Web Consortium) e também será utilizado o padrão de arquitetura MVC (Model-View-Controller).

- **Manutenibilidade**

ID	Descrição
[RNF.01.006]	A aplicação será construída utilizando-se da arquitetura MVC (Model-View-Controller) que ajuda a garantir uma boa manutenibilidade do sistema.

9 Rastreabilidade

NEC x RF

ID	RF.01.001	RF.01.002	RF.01.003	RF.01.004	RF.01.005	RF.01.006	RF.01.007	RF.01.008	RF.01.009
NEC.01	X	X	X	X	X	X	X	X	X
NEC.02	X	X	X	X	X	X	X	X	X
NEC.03			X	X	X		X	X	X
NEC.04				X	X		X	X	X
NEC.05				X	X			X	X
NEC.06				X	X			X	X
NEC.07				X	X				X
NEC.08				X	X				X

RF X RF

ID	RF.01.001	RF.01.002	RF.01.003	RF.01.004	RF.01.005	RF.01.006	RF.01.007	RF.01.008	RF.01.009
RF.01.001	X	X	X	X	X	X	X	X	X
RF.01.002	X	X	X	X	X	X	X	X	X
RF.01.003			X	X	X	X		X	X
RF.01.004				X	X		X	X	X
RF.01.005				X	X			X	X
RF.01.006				X	X	X	X	X	X

RF.01.007					x	x	x	x	x
RF.01.008					x	x		x	x
RF.01.009								x	x

RF X RNF

ID	**RF.01.001**	**RF.01.002**	**RF.01.003**	**RF.01.004**	**RF.01.005**	**RF.01.006**	**RF.01.007**	**RF.01.008**	**RF.01.009**
RNF.01.001	x	x	x	x	x	x	x	x	x
RNF.01.002	x	x	x	x	x	x	x	x	x
RNF.01.003	x	x	x	x	x	x	x	x	x
RNF.01.004	x	x	x	x	x	x	x	x	x
RNF.01.005	x	x	x	x	x	x	x	x	x
RNF.01.006	x	x	x	x	x	x	x	x	x

14. APÊNDICE B - Registro da Entrevista

Ferramenta de Levantamento de Requisitos.

14.1 Anexo – Informações Adquiridas com o Usuário

ANEXO A – Documento de Entrevista

Empresa: Escola Dimensão.

Entrevistado: Ivânia Aparecida de Oliveira Lima

Número: 01

Data: 15/04/2011

Duração: 1,5 h.

Conteúdo:

Obtenção de informações básicas referentes ao esboço didático usado pela instituição de ensino para elaboração dos horários, levantando suas bases, que em sua maioria são já predefinidas pela secretaria da educação.

Agendamento de próxima reunião para a data de 20/04/2011.

Empresa: Escola Dimensão

Entrevistado: Ivânia Aparecida de Oliveira Lima .

Número: 02

Data: 20/04/2011

Duração: 1,5 h.

Conteúdo:

Aprofudamento referindo aos métodos utilizados para se chegar a melhor solução de horário escolar, como regras de exceções, disponibilidades de professores, quantidade de aulas por matéria semanal, dias letivos, quantidade de aula diária, folga de professores, aulas vagas, dentre outras regras.

Agendamento de próxima reunião para a data de 30/04/2011.

Empresa: Escola Dimensão

Entrevistado: Ivânia Aparecida de Oliveira Lima e Simone Maria de Oliveira Martins. .

Número: 03

Data: 30/04/2010

Duração: 1 h.

Conteúdo:

Apresentação de possíveis soluções e discussão sobre as regras criadas para chegar ao horário escolar em cima de probabilidades. Conclusão de requisitos e agendamento de metas para versão do software, com apresentação de soluções que seriam usadas e tipos de tecnologia.

14.2 ANEXO B - Documentação colhida do Usuário

Horário Escolar

5º ANO

Segunda-Feira	Terça-feira	Quarta-feira	Quinta-feira	Sexta-feira
Inglês	Ed. Física	Matemática	Geografia	Ciências
Matemática	Matemática	Matemática	História	Arte
Matemática	História	Ciências	Ensino Religioso	Ciências
Gramática	Leitura e Interpretação	Gramática	Gramática	Redação
História	Inglês	Geografia	Geografia	Redação

6º ANO

Segunda-feira	Terça-feira	Quarta-feira	Quinta-feira	Sexta-feira
Gramática	Ed. Física	Ensino religioso	Espanhol	Gramática
Inglês	Geografia	Espanhol	Geografia	Arte
História	Matemática	Geografia	Matemática	História
Matemática	História	Matemática	Inglês	Ciências
Ciências	Leitura e Interpretação	Ciências	Gramática	Matemática

7º ANO

Segunda-feira	Terça-feira	Quarta-feira	Quinta-feira	Sexta-feira
Arte	Ed. Física	Inglês	Ensino Religioso	História
Gramática	Gramática	Geografia	Leitura e Interpretação	Matemática
Ciências	Inglês	Matemática	Geografia	Matemática
História	Matemática	Ciências	Matemática	Espanhol
Geografia	História	Espanhol	Ciências	Redação

9º ANO

Segunda-feira	Terça-feira	Quarta-feira	Quinta-feira	Sexta-feira
Arte	Ed. Física	Gramática	Ensino Religioso	Inglês
Redação	História	História	Ciências	Ciências
Geografia	Espanhol	Espanhol	Gramática	Leitura e Interpretação
Inglês	Geografia	Geografia	História	Matemática
Matemática	Matemática	Matemática	Matemática	Ciências

Figura16: Horário Manual