UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS VII – GOVERNADOR ANTÔNIO MARIZ
CENTRO DE CIÊNCIAS EXATAS E SOCIAIS APLICADAS
CURSO DE LICENCIATURA EM COMPUTAÇÃO

DAMIÃO RODRIGUES DE SOUSA

ANÁLISE DA IMPORTÂNCIA DA MODELAGEM DE *SOFTWARE* NAS FASES DE CONCEPÇÃO E ELABORAÇÃO DO RUP, ATRAVÉS DE UM ESTUDO DE CASO: SISTEMA DE GESTÃO ESCOLAR (SGE ARCO-ÍRIS)

PATOS – PB
2015

DAMIÃO RODRIGUES DE SOUSA

ANÁLISE DA IMPORTÂNCIA DA MODELAGEM DE *SOFTWARE* NAS FASES DE CONCEPÇÃO E ELABORAÇÃO DO RUP, ATRAVÉS DE UM ESTUDO DE CASO: SISTEMA DE GESTÃO ESCOLAR (SGE ARCO-ÍRIS)

Monografia de conclusão de curso apresentada ao Curso de Licenciatura em Computação da Universidade Estadual da Paraíba, Campus - VII, em cumprimento às exigências para obtenção do grau de Licenciado em Computação.

Orientador: Prof^{0}. MSc. Rodrigo Alves Costa

PATOS – PB

2015

UEPB - SIB - Setorial - Campus VII

S725a Sousa, Damião Rodrigues de
Análise da importância da modelagem de software nas fases de concepção e elaboração do RUP através de um estudo de caso [manuscrito]: Sistema de Gestão Escolar (SGE Arco-íris) / Damião Rodrigues de Sousa. - 2015.
71 p. : il. color.

Digitado.
Trabalho de Conclusão de Curso (Graduação em Computação) - Centro de Ciências Exatas e Sociais Aplicadas, Universidade Estadual da Paraíba, 2015.
"Orientação: Prof. Me. Rodrigo Alves Costa, CCEA".

1. RUP. 2. UML. 3. Modelagem de software. 4. Desenvolvimento de software. I. Título.

21. ed. CDD 005.1

Damião Rodrigues de Sousa

ANÁLISE DA IMPORTÂNCIA DA MODELAGEM DE SOFTWARE NAS FASES DE CONCEPÇÃO E ELABORAÇÃO DO RUP, ATRAVÉS DE UM ESTUDO DE CASO: SISTEMA DE GESTÃO ESCOLAR (SGE ARCO-ÍRIS)

Trabalho de Conclusão de Curso apresentado ao Curso de Licenciatura em Computação da Universidade Estadual da Paraíba, em cumprimento à exigência para obtenção do grau de Licenciado em Computação

Aprovado em 11 de junho de 2015

BANCA EXAMINADORA

Rodrigo Alves Costa
(Orientador)

Luiz Augusto de Macêdo Morais
(Examinador)

Wellington Candeia de Araújo
(Examinador)

RESUMO

Na atualidade há uma necessidade dos departamentos de Tecnologia da Informação (TI) nas organizações de se alinhar com os objetivos de negócios das mesmas, pois ambas, tecnologia e empresas andam lado a lado em suas atividades internas e processos. Os modelos de processos de *software*, juntamente com a atividade de modelagem, têm grande valor no desenvolvimento de sistemas, pois ambos definem um meio de comunicação uniforme entre a equipe de desenvolvimento e o resto da organização. Essa comunicação ocorre através da geração de artefatos e documentos, que são produtos de trabalho finais ou intermediários, produzidos e usados durante o projeto. Este trabalho tem como objetivo analisara importância da modelagem de *software* nas disciplinas de Modelagem de Negócios, Requisitos e Análise e *Design*, das fases de Concepção e Elaboração do *Rational Unified Process* (RUP). Para demonstração dessa pesquisa será elaborado um estudo de caso de Sistema de Gestão Escolar (SGE Arco-Íris) perante o qual serão postos em prática tais conceitos. No decorrer do trabalho pretende-se mostrar a importância que a modelagem de *software* tem nessas disciplinas, elaborando alguns artefatos do RUP em conjunto com os diagramas da UML onde será possível ter um maior entendimento dos objetivos de negócio, dos requisitos do sistema e das partes do sistema e seus acoplamentos.

Palavras-chave: RUP. UML. Modelagem de *software.* Desenvolvimento de *software.*

ABSTRACT

Nowadays, Information Technology departments in organizations have identified the need to clearly state their business objectives, since both, technology and organizational business, need to be aligned in their internal activities and processes. Software process models, along with the modeling activities, add significant earned value in systems development, by defining a uniform means of communication between the development team and the remainder of the organization.This communication occurs through the generation of artifacts and documents, which are work products either in their final or intermediate versions, produced and used during a project. This work aims to analyze the importance of software modeling in the disciplines of Business Modeling, Requirements Engineering and Analysis and Design, during the stages of conception and elaboration of the Rational Unified Process (RUP).In order to confirm the results of this research, a case study has been performed, by practicing the concepts aforementioned throughthe modeling of a School Management System (named “SGE Arco-Íris”), meanwhile pursuing to show the importance of software modeling by preparing RUP artifacts using UML diagrams, to increase understanding of business objectives, system requirements and subsystems, along with their respective couplings.

Keywords: RUP. UML. Software modeling. Development process of software.

LISTA DE FIGURAS

LISTA DE QUADROS

SUMÁRIO

1 INTRODUÇÃO

Atualmente as organizações vivenciam um cenário bastante dinâmico, tornando necessárias mais agilidade e confiabilidade em seus processos de negócios. Sendo assim, os sistemas computacionais se tornam uma parte fundamental no ambiente organizacional e vêm ganhando um importante papel nos processos de tomada de decisão das empresas.

Tais sistemas precisam ser cada vez mais adaptáveis a esse dinamismo que os dias contemporâneos impõem, gerando um grande desafio quando se cria um projeto para o desenvolvimento de *software*, pois esses projetos tendem a falhar de diversas formas. Para se prevenir disso, as organizações precisam identificar várias dificuldades que, segundo Kruchten (2003), são: incompreensão das necessidades do usuário final; deficiência em lidar com a variação dos requisitos; dificuldade em manter ou estender o *software*; baixa qualidade de *software* e falta de comunicação entre os membros da equipe.

Sommerville (2011) define um processo desenvolvimento de sistema como "um conjunto de atividades e resultados associados que geram um produto de *software"*. Os modelos de processos de *software*, juntamente com a atividade de modelagem, têm um grande valor no desenvolvimento de sistemas, pois a aplicação de métodos para desenvolvimento se dá pelo fato de que ambos definem um meio de comunicação uniforme entre a equipe de desenvolvimento e o resto da organização.

A uniformidade proporcionada pela utilização da modelagem em um processo de desenvolvimento resulta em um ganho no desempenho das atividades, pois seu uso significa basicamente que se estará representando aspectos ou visões da arquitetura de *software,* desempenhando um papel fundamental para gerenciar a complexidade inerente ao *software* (SOMMERVILLE, 2011)

Existem vários modelos de processo de desenvolvimento de software propostos na literatura, tais como: Modelo Cascata, Espiral, Prototipação, RUP (*Rational Unified Process*), métodos ágeis como o *Extreming Programing* (XP) e SCRUM. Há também a possibilidade da combinação entre vários destes modelos (PRESSMAN, 2006).

Dentre os modelos apresentados de processo de desenvolvimento de *software* propostos na literatura da engenharia de *software*, o RUP se destaca por

fornecer uma abordagem disciplinada para o desenvolvimento de *software*, nomeando tarefas e responsabilidades dentro de uma organização. Segundo Kruchten (2003) ele tem como foco assegurar a produção de *software* da mais alta qualidade, que satisfaça as expectativas dos usuários finais dentro de um cronograma e orçamento previsíveis

As atividades do RUP dão ênfase à criação e manutenção de modelos, visando minimizar a sobrecarga associada à geração e manutenção de documentos e maximizar o conteúdo das informações relevantes. Ele apresenta seis das melhores práticas que são: desenvolver o software iterativamente, gerar requisitos, usar arquiteturas baseadas em componentes, modelar software visualmente, verificar a qualidade do software e controlar as mudanças do *software*.

O RUP utiliza a UML (*Unified Modelling Language*) ou Linguagem de Modelagem Unificada. Segundo Booch *et al.* (2000), trata-se de "uma linguagem-padrão para a elaboração da estrutura de projetos de *software*, podendo ser empregada para a visualização, a especificação, a construção e a documentação dos artefatos do *software*".

Eriksson e Penker (2000) reforçam a idéia, afirmando que a UML tem a capacidade de proporcionar uma forma padrão para a preparação de planos de projetos de sistemas, inserindo aspectos conceituais tais como processos de negócios e funções do sistema, como também itens concretos como as classes escritas em determinada linguagem de programação.

Vicente (2003) afirma que "a UML é antes de tudo uma linguagem de modelagem, apresentando um vocabulário e regras próprias que norteiam a criação e a leitura de modelos bem formatados". Entretanto, pelo fato de não ser um modelo de desenvolvimento, ela por si só não associa e não relaciona sua notação com um processo de desenvolvimento de *software* especificamente: a criação e o momento da criação dos artefatos vão depender da metodologia que será adotada no projeto.

O RUP consiste em um processo de desenvolvimento que utiliza o paradigma Orientação a Objetos (OO) que tem como objetivo desenvolver modelos de análise que satisfaçam um conjunto de requisitos definidos pelo cliente, que descrevam as informações, as funções e o comportamento de seus elementos constituintes.

Sua estrutura possui quatro fases bem definidas, a saber: Concepção, com ênfase no escopo do sistema; Elaboração, com ênfase na arquitetura; Construção, com ênfase no desenvolvimento; e a Transição, com ênfase na implantação. A

modelagem tem mais ênfase nas fazes de Concepção e Elaboração onde e necessário ter uma melhor compreensão dos requisitos do sistema e uma melhor comunicação entre os *stakeholders,* que é qualquer pessoa ou organização que tenha interesse, ou seja, afetado pelo projeto. Através da geração de artefatos, que são produtos de trabalho finais ou intermediários produzidos e usados durante o projeto (RUP, 2014).

Assim este trabalho irá seguir as fases de Concepção e Elaboração realizando a modelagem de um sistema de gestão escolar (SGE Arco-íres) que é um sistema que irá gerenciar todas as atividades de expediente da escola Arco-íres. Onde será elaborado os principais artefatos nestas fases para uma futura implementação do mesmo na escola, pois ela não utiliza nenhum sistema de gestão em suas atividades de expediente.

1.1 Objetivo Geral

Este trabalho tem como objetivo analisar a importância da modelagem de *software* nas disciplinas de Modelagem de Negócios, Requisitos e Analise e *Design,* onde elas têm maior concentração nas fases de Concepção e Elaboração do RUP. Para isso, será necessário a realização de um estudo de caso de um sistema proposto, o (SGE Arco-Íris).

1.2 Objetivos específicos

- Estudar os conceitos da Engenharia de Software;
- Pesquisar a importância da modelagem de *software* no processo de desenvolvimento;
- Estudar a UML e os diagramas que a compõem;
- Desenvolver os artefatos e diagramas em UML necessários.

1.3 Justificativa

Devido à evolução tecnológica, à expansão da *internet*, à melhoria dos equipamentos de *hardware* e às constantes mudanças nos processos de negócio das organizações, as empresas passaram a ser muito mais dinâmicas, e ter

mecanismos de adaptação frente às mudanças organizacionais e priorização de investimentos para se adaptar às flutuações deste mercado globalizado.

As instituições de ensino não podem ficar pra traz neste contexto necessitando de sistemas computacionais cada vez mais dinâmicos que gerencie suas atividades diárias, nesses sistemas as funções devem atender as suas exigências atuais havendo a necessidade de projetos bem documentados e modelados.

Para que o departamento de Tecnologia da Informação (TI) se alinhe com os objetivos de negócios das organizações, surge a necessidade de aplicativos mais flexíveis às mudanças durante seu ciclo de vida. Um sistema bem documentado facilitará a compreensão de todas as partes interessadas no desenvolvimento do projeto, como também servirá de base para futuras mudanças que poderão ser feitas por outras pessoas, atendendo às futuras necessidades e objetivos da organização (TAMAKI, 2007; MAKINO, 2009).

1.4 Estrutura do Documento

Este trabalho está dividido em quatro capítulos, incluindo a presente introdução, que visam a demonstração, o desenvolvimento do trabalho e o método de realização do estudo. A pesquisa está dividida da seguinte forma:

Capítulo 1 - Introdução - Nesta seção será introduzida a problemática, a justificativa do trabalho, seus objetivos, além dos aspectos do contexto atual sobre os assuntos-chave do trabalho.

Capítulo 2 - Revisão da Literatura - Este capítulo irá tratar sobre o embasamento teórico do trabalho. É nele que serão vistos conceitos sobre os assuntos-chave na área tema desta pesquisa.

Capítulo 3 - Estudo de Caso: Sistema de Gestão Escolar (SGE Arco-Íris) - Este capítulo se responsabilizará por apresentar, contextualizar e analisar o sistema proposto.

Capitulo 4 - Considerações Finais - Neste momento serão abordados os resultados obtidos através da pesquisa.

2 FUNDAMENTAÇÃO TEÓRICA

A fundamentação teórica é o pilar de sustentação dos conceitos envolvidos, representando uma parte importante no desenvolvimento de uma monografia, pois é nela que se encontra o estado da arte com relação ao tema proposto no trabalho. Este capítulo está dividido em três seções, nas quais são discutidos assuntos relevantes ao presente trabalho.

2.1 Engenharia de Software

Engenharia de *Software* consiste na aplicação de uma abordagem disciplinada, sistemática e mensurável aos princípios da Ciência da Computação e representa o estabelecimento de princípios científicos juntamente com métodos para a implementação eficaz de processos, técnicas e ferramentas para a construção do produto de *software*, além de apresentar alternativas para o gerenciamento do processo de desenvolvimento, conhecidos como modelos ou metodologias (FUNKS, 2002; PURRI, 2006).

Sommerville (2011) descreve um processo de *software* como “uma sequência lógica de práticas que objetiva o desenvolvimento ou evolução de sistemas de *software”*. Na engenharia de software, processo é definido como um conjunto de tarefas ordenadas, que tem como objetivo entregar um produto de *software* de maneira eficiente, previsível e que atinja as necessidades de negócio da organização.

Um processo de desenvolvimento de produto de software, que utilize um método que não atenda às necessidades do projeto, ou ainda a falta de uma metodologia, pode ser a causa da qualidade baixa do produto final. Pressman (2006) afirma que “um processo de desenvolvimento claramente documentado facilita muito quando se deseja ter um controle maior em busca da qualidade”, pois, com uma boa documentação, é bem mais fácil de realizar um produto novo ou adaptar o que já está pronto.

Um processo de *software* é composto de fases e informações associadas que são solicitadas inicialmente para o desenvolvimento de um aplicativo. Apesar da grande quantidade de modelos ou metodologias de processos, todos apresentam atividades comuns: especificação (levantamento de requisitos e restrições de

software), desenvolvimento (projeto e codificação), validação (realização de testes) e evolução (evolução do software conforme as necessidades do cliente). O que os diferencia é que cada um aplica uma ênfase diferente a essas atividades, definindo um fluxo de trabalho que invoca cada atividade de maneira diferente (SOMMERVILLE, 2011).

Com o avanço dos estudos na área, vários modelos de processos vêm sendo concebidos, com o propósito de formar uma grande variedade de novos domínios de aplicações. Na próxima seção serão apresentados alguns dos principais modelos de desenvolvimento citados na literatura.

2.1.1 Modelos de Processos de Softwares

Segundo Sommerville (2011), “um modelo de processo de software é uma representação abstrata de um processo de software”. Assim,os modelos de processos de *software*s foram criados para tornar a atividade de desenvolvimento menos caótica, organizando o desenvolvimento, sendo um conjunto formado por procedimentos, técnicas, ferramentas e documentação.

Ao se utilizar um modelo de desenvolvimento, busca-se seguir um método já testado e consistente, que poderá proporcionará qualidade e eficiência. Existem vários modelos, a seguir são descritos alguns modelos genéricos amplamente utilizados (SOMMERVILLE, 2008; PRESSMAN, 2006).

- **Modelo cascata -** Desenvolvido no final da década de 1960 e começo da década de 1970, o modelo *Waterfall*, também conhecido como Modelo Seqüencial Linear, possui fases que são sistematicamente seguidas de maneira linear (PRESSMAN, 2006). Esta abordagem assume que um produto de *software* tem um ciclo de vida semelhante ao de qualquer produto, com início, meio e fim.
- **Modelo Incremental -** De acordo com Pressman (2006), esse modelo é uma adaptação do “Modelo Sequencial Linear”, e assume que o produto de *software* acrescentará novas funcionalidades. A cada nova funcionalidade, ou conjunto de novas funcionalidades,teremos um incremento que seguirá as fases do modelo linear.

- **Modelo Iterativo -** A idéia principal desse modelo é que um sistema deve ser desenvolvido de forma incremental, onde cada incremento vai adicionando ao sistema novas capacidades funcionais, até a obtenção do sistema final. A cada passo realizado, modificações podem ser introduzidas.
- **Modelo Espiral -** O Modelo Espiral é um modelo evolucionário de processo de *software* que combina a natureza iterativa da prototipagem com os aspectos controlados e sistemáticos do modelo cascata (PRESSMAN, 2006). Ao invés de representar o processo de *software* como uma sequência de atividades, o processo é representado como uma espiral (SOMMERVILLE, 2011).
- **Desenvolvimento baseado em componentes** – Esta metodologia baseia-se na existência de um número significativo de componentes reusáveis. O processo de desenvolvimento do sistema enfoca a integração desses componentes em vez de desenvolvê-los a partir do zero. (SOMMERVILLE, 2011).

De acordo com Sommerville (2011), esses modelos não são mutuamente exclusivos e frequentemente são utilizados em conjunto. Uma metodologia que combina elementos de todos esses modelos é o RUP, que fornece técnicas a serem seguidas pelos membros da equipe de desenvolvimento de *software* com o objetivo de aumentar a sua produtividade.

2.2 Processo Unificado da Rational (RUP)

Segundo Kruchten (2003), o RUP é um processo de engenharia de *software* que oferece uma abordagem em disciplinas para atribuir tarefas e responsabilidades dentro de uma organização de desenvolvimento. Em suas atividades, o RUP enfatiza a criação e manutenção de modelos utilizando a UML, proporcionando assim representações do sistema de *software* em desenvolvimento que pode ser ajustado e redimensionado para atender as necessidades do projeto.

Pressman (2006) diz que “sob a orientação do RUP, o desenvolvimento é centrado na arquitetura e suas atividades por caso de uso dispõem de arquitetura robusta que promove o desenvolvimento do *software”.* Tal foco minimiza assim o

retrabalho e aumenta a probabilidade da reutilização de componentes, como também a capacidade de manutenção eventual no sistema.

Sua arquitetura carrega muita informação que são de interesse dos *stakeholders*, a saber, analista de sistemas, arquitetos, usuários e clientes e gerentes de projeto, que possuem diferentes expectativas do projeto em seus diferentes modelos. O RUP apresenta cinco diferentes visões da arquitetura, que é descrita por um ou mais tipos de diagramas UML, e tais visões estão dispostas na seção 2.2.3 deste capítulo (RUP, 2014).

2.2.1 RUP e as melhores práticas de desenvolvimento de *software*

Segundo Kruchten (2003), o RUP tem uma filosofia que consiste em buscar a melhoria dos processos e fases de projetos seguindo as melhores práticas de desenvolvimento de *software*. Entre elas, estão:

--Desenvolvimento de software iterativamente - Segundo Kruchten (2003), o "desenvolvimento iterativo proposto pelo RUP tem por objetivo conduzir o projeto em ciclos, ou seja, em iterações". Esses ciclos possibilitam verificar os itens de maior risco em cada etapa do desenvolvimento, reduzindo significativamente o perfil de risco de um projeto. O dinamismo que um projeto pode seguir, seja por seu próprio desenvolvimento, ou por restrições na arquitetura do *software*, ou ainda devido as constantes necessidades do usuário, sugere ao desenvolvedor que ele adote uma abordagem mais flexível como esta;

--Gerenciamento de requisitos - O gerenciamento de requisitos tem por objetivo gerenciar as mudanças de requisitos do sistema, melhorando assim o entendimento dos clientes em relação às suas reais necessidades. Segundo Pressman (2006), trata-se de um modelo sistemático, que auxilia uma equipe de um projeto a identificar, controlar, rastrear, documentar e organizar os requisitos de um sistema. O RUP sugere ao desenvolvedor uma maneira de obter os requisitos;

-- Utilização de arquitetura baseada em componentes - Componente é definido por Kruchten (2004) como "uma parte do sistema que estará sendo criado, buscando executar funções bem definidas e específicas dentro do projeto". Os componentes formam a arquitetura do software, e são subdivididos por mecanismos e padrões de controle. Eles têm a capacidade de propor projetar uma arquitetura

flexível, que seja fácil realizar as mudanças, que seja intuitivamente compreensível e que, principalmente, promova a reutilização de software mais eficiente;

-- Modelagem visual de *software* - A modelagem de um *software* nada mais é que utilização de ferramentas específicas, buscando simplificar a realidade a partir de diagramas UML. Pressman (2006), diz que as abstrações visuais obtidas através da modelagem ajudam o desenvolvedor a se comunicar com os diferentes aspectos do *software*. Para isso, o modelo é criado com base em observações do mundo real, ou em uma aproximação baseada nos objetivos do sistema;

-- Verificação contínua da qualidade do software - Segundo Sommerville (2011), a verificação de qualidade de um *software* é testada durante todo o processo de criação de um projeto, pois as iterações que são implementadas em cada fase necessitam de um ato que comprove suas funcionalidades. Assim, a qualidade deve ser focada no que diz respeito aos requisitos, baseados em confiabilidade, funcionalidade, desempenho da aplicação e desempenho do sistema. Essa prática auxilia no planejamento do projeto, na implementação, na execução e na avaliação dos testes;

-- Gerenciamento e controle de mudanças - Em todo projeto de *software* há mudanças. Em alguns casos, essas mudanças são inevitáveis, por isso a capacidade de gerir a mudança e de ser capaz de acompanhá-las é essencial. Esta prática descreve como controlar, acompanhar e monitorar as mudanças, para permitir o sucesso do desenvolvimento iterativo (SOMMERVILLE, 2011).

2.2.2 Arquitetura do RUP

O RUP, em sua arquitetura, apresenta duas dimensões, a dinâmica e a estática. A dimensão horizontal representa o aspecto dinâmico do processo, o seu ciclo de vida, e faz com que o projeto do *software* seja elaborado com uma sequência de iterações incrementais. A dimensão vertical representa o aspecto estático, descrito em termos de componentes do processo: atividades, disciplinas, artefatos e papéis do processo (KRUCHTEN, 2003).

A Figura 01 representa o ciclo de vida do RUP onde pode ser observadas ambas as dimensões.

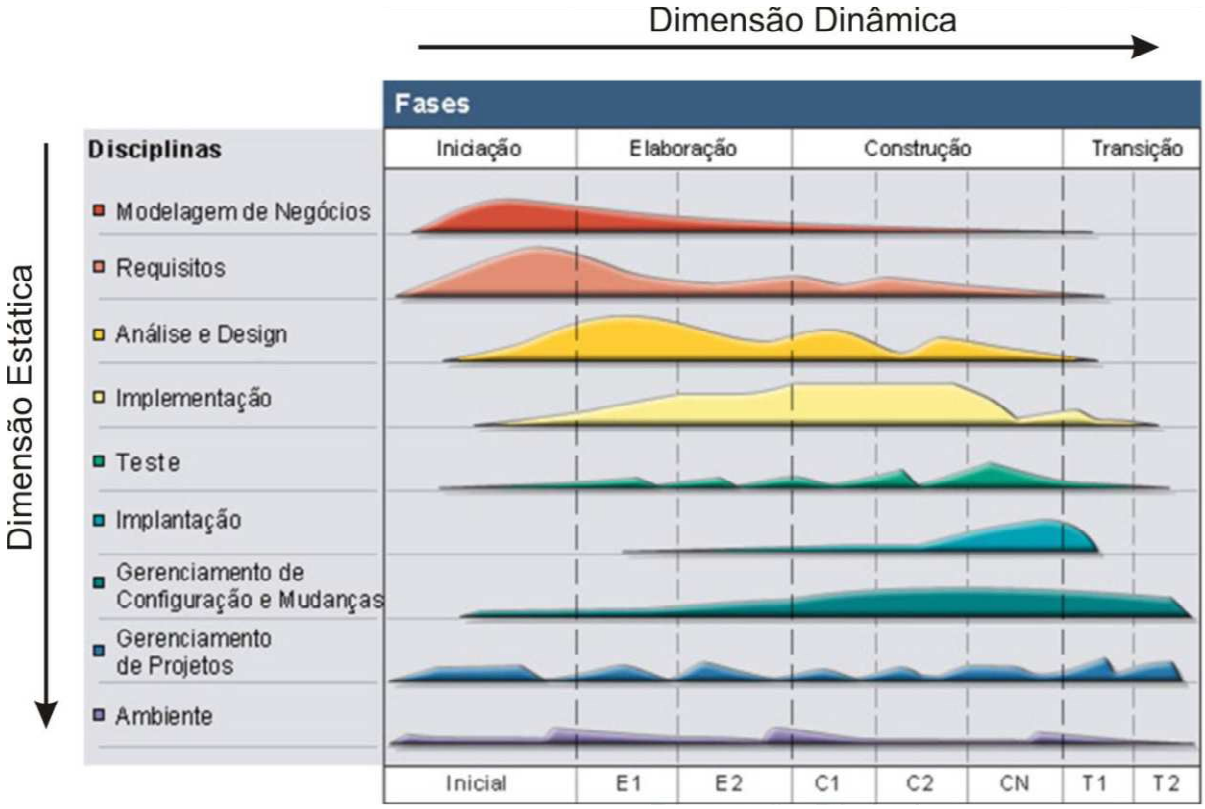


Figura 01 - As duas dimensões do RUP
Fonte: Adaptada RUP (2014)

2.2.2.1 Dimensão Dinâmica

No aspecto dinâmico, a arquitetura do RUP apresenta quatro fases: concepção, elaboração, construção e transição. Define-se “fase” como o tempo decorrido entre os principais marcos, que são pontos significativos do projeto, eventos cuja ocorrência precisa ser reportada às partes interessadas, que tomam decisões importantes acerca da continuidade do projeto, além de, se for o caso, alterar em relação ao escopo, orçamento e cronograma do mesmo (SCOTT, 2006; KRUCHTEN, 2003).Estas quatro fases estão descritas a seguir:

-- Concepção - abrange as atividades de comunicação com o cliente e de planejamento, tendo como objetivo a identificação dos requisitos, a definição do escopo do sistema e a iniciação do planejamento. Esta fase é muito importante, principalmente para os esforços dos desenvolvimentos novos, nos quais há muitos riscos de negócios e de requisitos que precisam ser tratados para que o projeto possa prosseguir (PRESSMAN, 2006);

-- Elaboração - inclui o planejamento e atividades de modelagem do processo. É nela que ocorre a expansão e o refinamento dos casos de uso elaborados pela

fase de concepção. Durante ela avalia-se os riscos técnicos e arquiteturais, incluindo cinco visões diferentes do software: o modelo de casos de uso, o modelo de análise, o modelo de projeto, o modelo de implementação e o modelo de implantação;

-- Construção - tem como objetivo desenvolver os componentes de *software* que tornam os casos de uso operacionais para os usuários finais. Para isso, os modelos de análise e projeto que foram iniciados durante a fase de elaboração são completados e os componentes são implementados conforme os requisitos levantados para o incremento de *software* referente;

-- Transição - tem como principal objetivo a entrega do sistema completo aos clientes e abrange os últimos estágios do processo e a primeira parte de implantação. Nesta fase, o *software* é implantado no cliente para testes com o intuito de se ter um *feedback* dos usuários em relação às falhas e assim realizar as modificações que se julguem necessárias.

2.2.2.2 Dimensão Estática

Segundo Kruchten (2003),a dimensão estática representa as disciplinas que agrupam as atividades que representam pacotes de trabalho que produzem resultados relevantes para o projeto e geralmente estão associadas à criação ou atualização de artefatos pelos responsáveis, de modo a agregar valor para o projeto.

Os processos do RUP definem papéis, atividades, artefatos e procedimentos que devem ser executados. Os papéis estão associados a quem são as pessoas que irão executar o projeto, sendo que uma mesma pessoa pode executar mais de um papel no mesmo projeto. Segundo Booch, *et al.* (2005),o RUP define nove fluxos de atividades, que são:

-- Modelagem do negócio - Esta disciplina visa o entendimento da estrutura e a dinâmica da organização, onde serão documentados os conceitos relativos ao negócio que será refinado posteriormente, à medida que os casos de uso forem sendo detalhados. Segundo Kruchten (2003), este modelo consiste em atores, que representam papéis externos para o negócio, e nos casos de uso, que são as atividades que os atores podem desempenhar no sistema.

-- Requisitos - Conforme Kruchten (2003),esta disciplina estabelece as funções que o sistema deve desempenhar, esboça a interface de usuário, fornece uma tradução clara dos requisitos à equipe de desenvolvimento e também do

sistema, assim como prover informações para o planejamento do projeto, tais como conteúdo técnico para os ciclos de desenvolvimento, além de subsídios para estimativas de prazo e custo.

Os requisitos podem ser funcionais, capturados através dos casos de uso, que documentam as entradas, as saídas e os processos, e não funcionais, que são compostos por características não associadas ao comportamento esperado do sistema, como usabilidade, confiabilidade, desempenho e suporte (SOMMERVILLE, 2011).

-- Análise e Design - Vislumbra transformar os requisitos em projeto, construir uma arquitetura para o sistema e adaptar o projeto ao ambiente. O objetivo deste processo é traduzir os requisitos numa especificação que explica como o sistema deve ser desenvolvido, por exemplo, características como a arquitetura, o ambiente operacional e a escala do projeto. Ela pode ser dividida em duas fases, Análise e *Design*. Na Análise o foco está nos requisitos funcionais do sistema, definindo um modelo mais simples para o mesmo, colhido através dos casos de uso. Já o *Design* possui foco na solução escolhida, sendo a mesma dirigida pelos requisitos não-funcionais do sistema. (KRUCHTEN, 2003).

-- Implementação - No entendimento de Kruchten (2003),a implementação visa estruturar o código e implementar classes e objetos, testando e integrando as unidades.Durante este processo serão desenvolvidas várias versões do sistema, ou partes dele, de modo que estes representem suas diferentes funcionalidades. Cada uma destas versões é construída através da integração de componentes novos ou subsistemas

-- Testes - Nesta disciplina o sistema é verificado para a identificação de erros, que também devem ser documentados. É possível validar o sistema de acordo com o que foi especificado e responder se o sistema foi desenvolvido de acordo com o projetado (SOMMERVILLE, 2011).

-- Implantação - Na disciplina de implantação, Kruchten (2003) afirma que se busca a certificação da entregado sistema para os usuários finais. O maior nível de atividades deste processo ocorre na fase de transição, entre o novo sistema e o anterior, caso exista.

-- Gerência de Configuração e Mudança - Nesta disciplina são controlados os artefatos produzidos no desenvolvimento do projeto, decorrentes de correções de erros, melhorias e inclusão de novos requisitos ou necessidades. Todas as

solicitações devem ser analisadas, registradas, recusadas ou aprovadas e desenvolvidas (KRUCHTEN, 2003).

-- Gerenciamento e Planejamento - Esta disciplina controla o gerenciamento de riscos. Além de disponibilizar guias para planejar, executar, acompanhar e monitorar o projeto. O fato de pouco projetos terem 100% de sucesso é um indicador de como se trata de uma tarefa complexa.

-- Ambiente - Este processo tem por objetivo definir os processos e ferramentas que proporcionem a implementação do sistema. Deve-se avaliar a cultura atual da empresa e definir uma lista de ferramentas que podem ser utilizadas e modelos de documentos que serão necessários (KRUCHTEN, 2003).

2.2.3 Visões de Arquitetura

No RUP, a arquitetura é representada por uma série de visões diferentes, se inicia com um conjunto típico de visões, denominado "modelo de visão 4+1"que, em sua essência, são fragmentos que ilustram os elementos dos modelos(KRUCHTEN, 2003).

Desenvolver um sistema complexo envolve vários aspectos, tais como: o aspecto funcional, não funcional e aspectos organizacionais, que utilizam-se de visões. Cada visão é descrita por um ou mais tipos de diagramas UML, que contêm informações que dão ênfase aos aspectos particulares do sistema. A Figura 02 apresenta as cinco visões estabelecidas por Kruchten (2003).

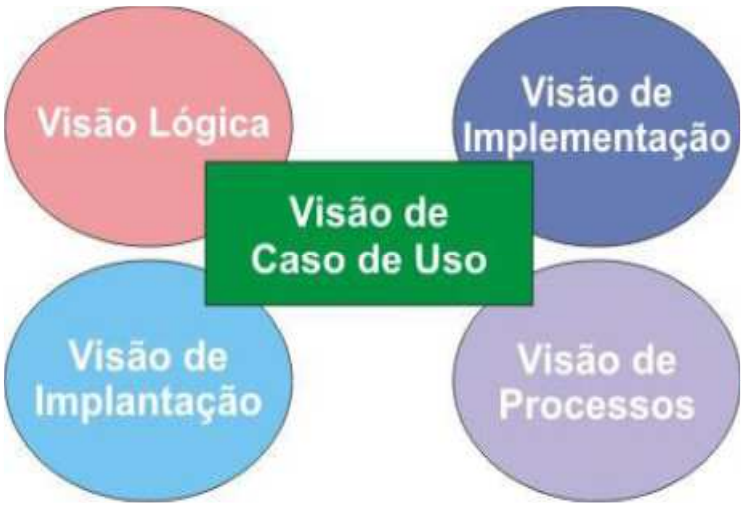


Figura 02 - Visões do RUP
Fonte: Adaptada Kruchten (2003).

Segundo Kruchten (2003), o uso de visões permite endereçar os requisitos de vários interessados no sistema: usuários finais, desenvolvedores, engenheiros de sistemas e gerentes de projetos. As descrições das visões podem ser vistas da seguinte forma:

-- Visão de Caso de Uso - Apresenta ou descreve a visão do usuário final e projetista, capturando os requisitos funcionais que irão definir a estrutura do sistema. Os elementos estáticos são capturados através do diagrama de caso de uso, sendo as características dinâmicas documentadas pelos diagramas de comunicação, sequência, estados e atividades.

-- Visão Lógica - Descreve como a funcionalidade do sistema será implementada, sendo elaborada principalmente pelos analistas e desenvolvedores. Ela descreve e especifica a estrutura estática do sistema, apresentando o sistema através de sua arquitetura, demonstrando as suas funcionalidades presentes. Os diagramas UML que representam os elementos estáticos são os de classes e objetos, sendo os aspectos dinâmicos capturados através dos diagramas de Interação (Sequência, Estados e Atividades).

-- Visão de Processo - Apresenta a concorrência do sistema, através dos processos e permite o melhoramento da utilização do ambiente em que se encontrará. Os diagramas relacionados para tratar dos aspectos dinâmicos e estáticos são os mesmos empregados na visão lógica, mas com foco nas classes ativas, que representam processamento concorrente.

-- Visão de Implementação - Apresenta os componentes subsistemas que compõem o sistema, sendo exatamente os elementos que apresentam o conjunto de interfaces requeridas e fornecidas. O diagrama UML utilizado para mostrar a estrutura estática do sistema é o Diagrama de Componentes, que possui características dinâmicas capturadas através dos diagramas de comunicação, estados e atividades.

-- Visão de Implantação - Apresenta finalmente como serão utilizados os periféricos e a topologia de *hardware* que serão utilizados pelo sistema e utiliza-se para mostrar os aspectos estáticos desta visão o diagrama de implantação, sendo que, para os aspectos dinâmicos, utilizam-se os diagramas de comunicação, estados e atividades.

Na próxima seção será explanado sobre o que é modelagem de *software* e UML, e seus principais diagramas.

2.3 Modelagem

Modelagem é uma técnica bastante utilizada quando se deseja passar informações visualmente, permite que o conhecimento seja transferido entre os diferentes grupos de intervenientes (técnicos e não técnicos), facilita e promove a comunicação entre todos envolvidos no projeto e provê uma visão mais adequada sobre o produto a ser desenvolvido (SILVA e VIDEIRA, 2001).

Booch *et al.* (2005), destaca os benefícios de se ter uma boa modelagem quando se projeta um sistema de *software*, pois ela estabelece conceitos únicos a partir de visões diversas, serve de instrumento de comunicação entre os *stakeholders*, favorece o processo de verificação e validação e gera maior entendimento da entidade real a ser construída. Na próxima seção será explanado um pouco sobre a UML, que é uma linguagem de modelagem visual para construir sistemas orientados a objetos e baseados em componentes. (BOOCH; JACOBSON; RUMBAUGH, 2005).

2.3.1 Linguagem Unificada de Modelagem (UML)

A UML é uma linguagem gráfica que auxilia na especificação, visualização e documentação de modelos de sistemas de software, incluindo sua estrutura e projeto. Ela é mantida e definida pela *Object Management Group* (OMG), um consórcio internacional da indústria da computação sem fins lucrativos que tem um dos papéis centrais a especificação de uma Arquitetura Dirigida a Modelos, a *Model-Driven Architecture* (MDA) (OMG, 2014).

Ela segue uma tendência a modelar sistema com o paradigma da Orientação a Objetos (OO). Esse paradigma baseia-se em conceitos da realidade (objetos, estados e estímulos) para desenvolver e modelar sistemas. Assim nota-se que a OO traz uma forma diferente de modelar e desenvolver sistemas, onde dados e processos são organizados e manipulados por objetos (BOOCH, RUMBAUGH e JACOBSON, 2005).

A UML é uma linguagem de modelagem independente de linguagem de programação, bem como de processo de desenvolvimento. Através dela é possível que haja uma boa comunicação entre as partes de um sistema facilitando a assim a

integração das decomposições do sistema (BOOCH; JACOBSON; RUMBAUGH, 2005).

A estrutura de conceitos da UML constitui um conjunto variado de notações, as quais podem ser aplicadas em diversos domínios de problemas e a diferentes níveis de abstração. Conforme Booch et al.(2005), o vocabulário da UML estende-se a três tipos básicos de blocos de construção:Itens, Relacionamentos e Diagramas, definidos a seguir.

2.3.1.1 Itens

Existem quatro tipos de itens na UML: itens de estrutura, comportamento, agrupamento e notação; segundo Booch et al. (2005),esses itens constituem os blocos de construção básicos orientados a objetos da UML .

Itens de estrutura configuram as partes mais estáticas do modelo, representando os elementos conceituais ou físicos. Os principais itens estruturais que se pode incluir em um modelo da UML são:

- Classes - descrições de conjuntos de objetos que compartilham os mesmos atributos, operações, relacionamentos e semântica.
- Interface - conjunto de operações que especificam serviços de uma classe ou componente.
- Caso de Uso - descrição de um conjunto de sequência de ações realizadas pelo sistema que proporciona resultados observáveis de valor para um determinado ator.
- Componentes - partes físicas e substituíveis de um sistema, que proporcionam a realização de um conjunto de interfaces.
- Nó - elemento físico existente em tempo de execução que representa um recurso computacional, geralmente com pelo menos alguma memória e, frequentemente, capacidade de processamento.
- Ator - se descreve como um conjunto de papéis que os usuários desempenham quando interagem com os casos de uso. Exemplos de atores são os seres humanos, um dispositivo de hardware ou até mesmo outro sistema.

Na Figura 03 pode-se ter uma visualização dos principais elementos de estrutura da notação da UML que são utilizadas para elaborar os principais diagramas e elementos são classificados como: classes, interfaces, casos de uso, atores, componentes e nós.

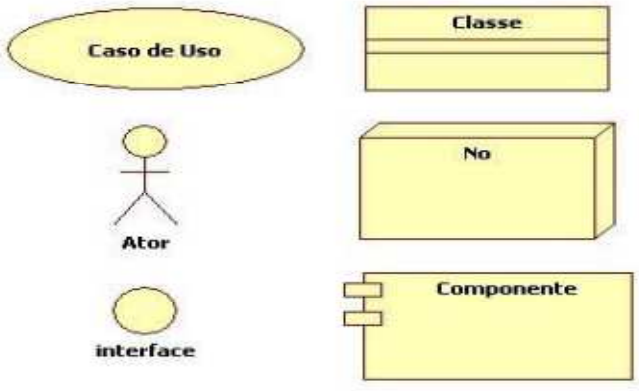


Figura 03 - Itens de Estrutura UML
Fonte: adaptado SILVA e VIDEIRA (2001).

Os itens comportamentais são as partes dinâmicas dos modelos de UML, representando comportamentos no tempo e no espaço. Ao todo, existem dois tipos principais de itens comportamentais (SILVA; VIDEIRA, 2001):

- Interação - abrange um conjunto de mensagens trocadas entre um conjunto de objetos em determinado contexto para a realização de propósitos específicos.
- Máquina de estado - especifica as sequências de estados pelos quais objetos ou interações passam durante sua existência em reposta a eventos, bem como suas respostas a esses eventos.

Os itens de agrupamento são as partes organizacionais dos modelos de UML e consistem n os blocos em que os modelos podem ser decompostos. Existe apenas um tipo principal de itens de agrupamento, chamado pacote, que é um mecanismo de propósito geral para a organização de elementos em grupos. Os itens estruturais, os itens comportamentais e até outros itens de grupos podem ser colocados em pacotes.

Os itens de anotação são as partes explicativas dos modelos de UML. São comentários, incluídos para descrever, esclarecer e fazer alguma observação sobre

qualquer elemento de modelo. Existe um único tipo de item de anotação, chamado nota, que é apenas um símbolo para representar restrições e comentários anexados a um elemento ou a uma coleção de elementos.

A Figura 04 ilustra outros elementos básicos do UML, elementos de comportamento (estados e mensagens), de agrupamento (pacotes) e de anotação (anotações ou notas).

Figura 04 - Elementos de comportamento, agrupamento e anotação.
Fonte: Adaptada SILVA e VIDEIRA (2001).

2.3.1.2 Relacionamentos

Booch *et al.* (2005) descreve que os relacionamentos são divididos tipos conforme descrito no Quadro 01.

Quadro 01 - Tipos de Relacionamentos

Tipo	Descrição	Notação
Dependência	São relacionamentos entre itens, nos quais as alterações de um podem afetar os aspectos de outro.	
Associação	Relacionamentos que representam uma conexão entre objetos de classes diferentes, possibilitando que eles conheçam uns aos outros.	
Agregação	São relacionamentos entre classes diferenciando a classe todo da classe parte. Onde a parte é um atributo do todo, somente criados se o todo ao qual estão agregados seja criado.	
Composição	É uma forma de agregação, relacionamento entre classes diferenciando a classe todo da classe parte. A parte só depende do todo e são criadas e destruídas junto com o mesmo	
Generalização ou herança	É um relacionamento que define uma classe como derivada de outra. Como o nome já cita a classe 'filha' herda todos os métodos e atributos visíveis da classe pai,	

Fonte: Adaptado Jacobson, Rumbaugh e Booch (2005).

Os Relacionamentos são as associações entre os itens de um diagrama e descrevem como os objetos interagem entre si.

2.3.1.3 Principais Diagramas

Os diagramas na UML servem para modelar um sistema, e cada diagrama representa uma visão diferenciada e parcial do sistema. Booch *et al*. (2005)o definem como sendo a "apresentação gráfica e um conjunto de elementos, geralmente representada como um gráfico conectado de vértices (itens) e arcos (relacionamentos)".

Os diagramas ajudam os stakeholders do projeto a compreendê-lo de pontos de vista diferente, já que cada tipo de diagrama proporciona uma visão do sistema. A UML versão 2.0 define 13 tipos de diagramas que podem ser organizados em diagramas estruturais, comportamentais e de interação, como exposto no Quadro 02 abaixo:

Quadro 02 - Diagramas UML

Diagramas Estruturais	Diagramas Comportamentais	Diagramas de Interação
Classes Objetos Pacotes Componentes Estrutura de Composição Implantação	Caso de Uso Gráfico de Estados Atividades.	Sequência Comunicação Tempo Visão Geral de Interações

Fonte: Adaptado Jacobson, Rumbaugh e Booch (2005).

Diagramas de Classes - Booch *et al.* (2005) mostra o diagrama de classes como sendo um dos mais utilizados na modelagem de sistemas orientados a objetos, por mostrar um conjunto de classes, interfaces e colaborações e seus relacionamentos.

O diagrama de classe possibilita a visualização dos aspectos estáticos do sistema, mais especificamente de blocos de construção e seus relacionamentos e a especificação dos detalhes de construção.

Na figura 03 tem-se um exemplo de um diagrama de classes, onde o retângulo representa uma classe contendo os atributos e métodos, e também pode

ser visto a ligação entre uma classe a outra, representando seus relacionamentos existentes entre elas.

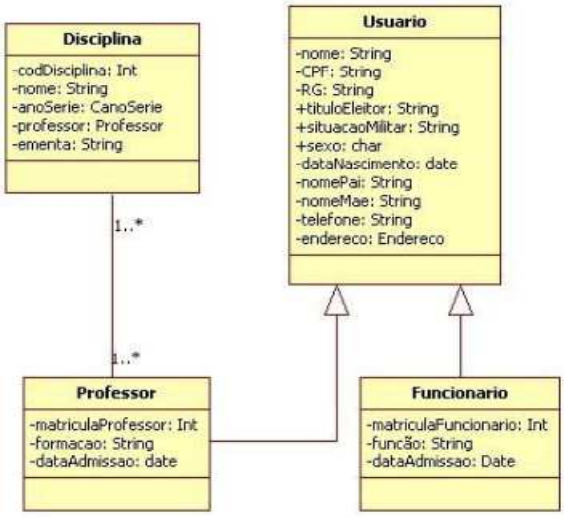


Figura 05 - Exemplo de Diagrama de Classes
Fonte: adaptado Jacobson, Rumbaugh e Booch (2005).

Diagrama de componentes - Jacobson, Rumbaugh e Booch (2005), definem um diagrama de componentes como aquele que mostra as partes internas, os conectores e as portas que implementam um componente. Componentes são coisas que podem ser executadas em nós (como processadores e dispositivos), por exemplo: executáveis, bibliotecas, tabelas, arquivos e documentos. Na Figura 06 temos um exemplo de diagrama de componentes.

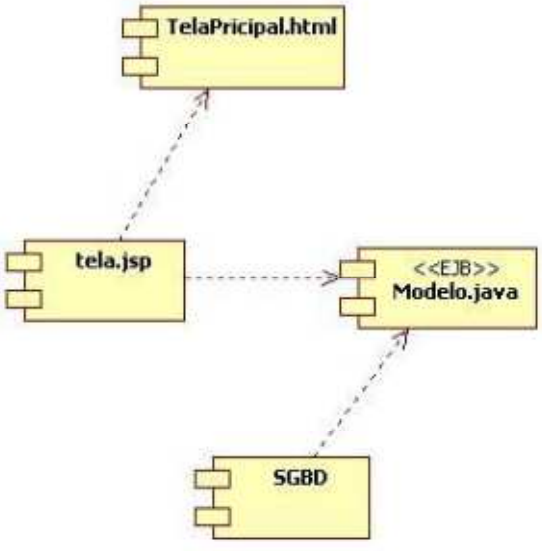


Figura 06 - Exemplo de Diagrama de Componentes
Fonte: Adaptado Jacobson, Rumbaugh e Booch (2005).

Diagrama de pacotes - Um pacote é um mecanismo de propósito geral para organizar elementos semanticamente relacionados em grupos. Todos os modelos de elementos que são ligados ou referenciados por um pacote são chamados de “conteúdo do pacote”. Ele pode ser visualizado através da Figura 07.

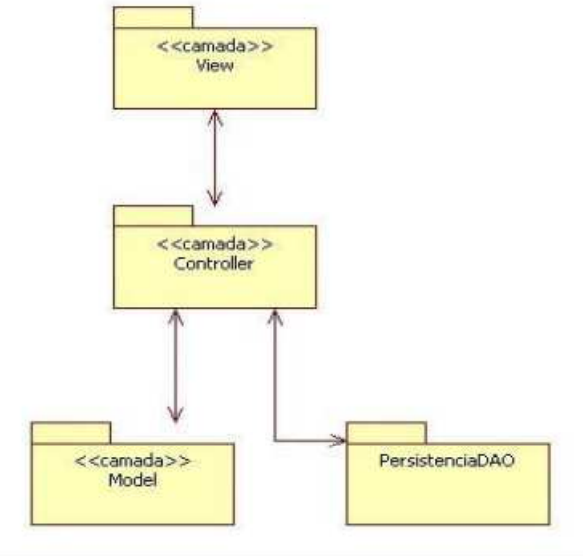


Figura 07 - Exemplo de Diagrama de Pacotes
Fonte: Adaptado Jacobson, Rumbaugh e Booch (2005)

Diagrama de implantação - Booch *et al.* (2005) o definem como um diagrama de classe que focaliza os nós do sistema, em que na maior parte envolve a modelagem da topologia de hardware em que o sistema é executado. Ele é importante para a visualização, documentação e especificação de sistemas embutidos, como sistemas distribuídos e do tipo cliente/servidor. Pode-se ver um exemplo de diagrama de implantação na Figura 08.

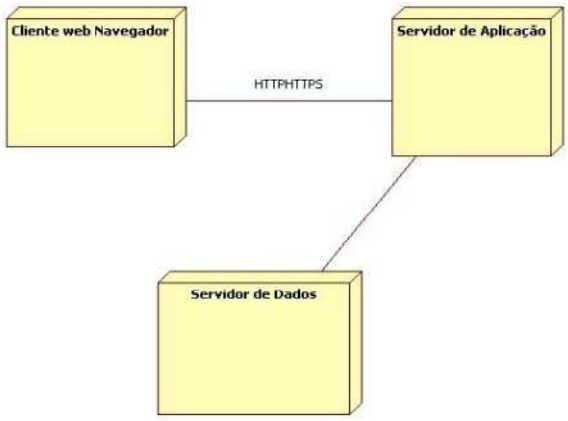


Figura 08 - Exemplo de Diagrama de Implantação
Fonte: adaptado Jacobson, Rumbaugh e Booch (2005)

Diagrama de caso de uso - Segundo Jacobson, Rumbaugh e Booch (2005), o caso de uso é um dos diagramas da UML que possibilitam a modelagem dos aspectos dinâmicos de um sistema, pois com ele é possível expor quais são os comportamentos pretendidos no desenvolvimento de um sistema.

Através dele os desenvolvedores podem compreender facilmente como o usuário final irá interagir diretamente com o sistema, eles contêm: Assunto, Casos de Uso, Atores e Relacionamentos, que podem ser de dependência, generalização e associação. (SILVA, 2007).

Comumente, esse tipo de diagrama é utilizado para a modelagem do contexto de um assunto e para a modelagem dos requisitos de um sistema. A Figura 09 ilustra um diagrama de caso de uso.

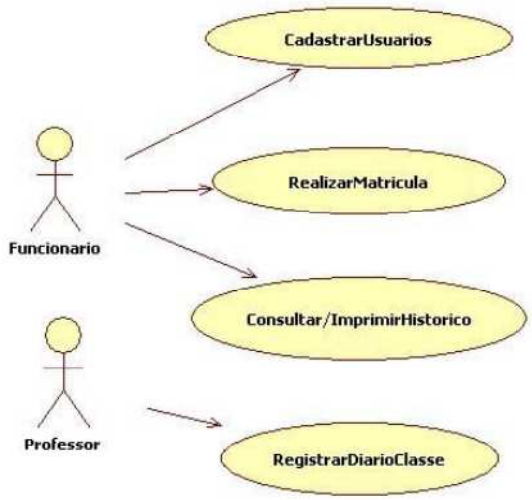


Figura 09 - Exemplo de Diagrama de Casos de Uso
Fonte: adaptado Jacobson, Rumbaugh e Booch (2005)

Concluída a parte do referencial teórico é hora de partir para o estudo de caso onde será aplicada a teoria vista neste capítulo. No capítulo seguinte será exposto o estudo de caso de um Sistema de Gestão Escolar (SGE Arco-Íris), analisado dentro dos objetivos deste trabalho.

3 ESTUDO DE CASO – SISTEMA DE GESTÃO ESCOLAR (SGE ARCO-ÍRIS)

Este estudo de caso irá utilizar o RUP de forma personalizada para a modelagem de um sistema proposto de Gestão Escolar (SGE Arco-íres), onde será possível analisar a importância da modelagem de *software.* Sua escolha se deu pelo fato de ser uma metodologia de modelagem flexível ao projeto, ou seja, ele pode ser adaptado a diversos portes e tipos de projetos de *software*.

Outro fator importante é que o RUP utiliza a UML, que é uma linguagem bastante utilizada na modelagem de *software* e tem como característica facilitar o entendimento e a comunicação por parte dos *stakeholders,* pois ajuda na descrição do projeto. Serão apresentados os artefatos elaborados seguindo a orientação do RUP durante o desenvolvimento do trabalho.

Devido aos objetivos e à natureza acadêmica deste trabalho, nem todas as disciplinas foram utilizadas para a implementação do modelo do sistema proposto, os modelos serão apresentados nas disciplinas com maior incidência de trabalho nas fases de Concepção e Elaboração. Foram aplicadas as seguintes disciplinas: Modelagem de Negócios, Requisitos e Análise e *Design*:

Para que o desenvolvimento de um sistema de *software* possa ser gerenciado, os artefatos são organizados em conjuntos que correspondem às disciplinas. Durante o decorrer deste trabalho foram gerados vários artefatos, por exemplo: Documento de visão, Documento de Arquitetura de Software e Diagramas de Caso de Uso e de Classes, em suas respectivas disciplinas.

O campo de estudo deste trabalho foi o Espaço Educacional Arco-Íris, uma escola localizada na cidade de Pombal-PB a referida escola conta com 497 alunos distribuídos no ensino infantil e fundamental. Conta também com 31 professores e distribuídos nas disciplinas em suas respectivas series e 10 funcionários entre técnicos administrativos e secretários, que dão toda a base para o funcionamento da instituição.

Nas seções a seguir será descrito como aconteceu a pesquisa, desde a obtenção dos requisitos do sistema até a modelagem dos diagramas utilizando as disciplinas mencionadas acima, nas fases de concepção e elaboração do RUP, como também os artefatos necessários para atingir o objetivo deste trabalho que mostram a modelagem no processo de desenvolvimento de *software*.

3.1 Modelagem de negócios

Esse é o primeiro fluxo de trabalho de acordo com o ciclo de vida do RUP. Tem um maior enfoque na fase de Concepção, mas também tem parte na fase de Elaboração. Realizou-se o levantamento de dados, que tornou possível entender o modelo de negócios do sistema e delimitar o escopo do projeto, definindo como o sistema será utilizado por cada usuário.

O artefato elaborado nesta disciplina é o Documento de Visão, que tem como ponto importante a descrição da visão geral do sistema proposto. Esse aborda os objetivos do sistema que serve como base para próximo o artefato, o Diagrama de Casos de Uso, cuja finalidade é coletar, analisar e definir os requisitos do sistema para uma possível implantação. Durante a realização deste trabalho, o documento foi atualizado na medida em que novos requisitos foram apontados, detalhados e inseridos no escopo do projeto.

Na próxima seção será descrito como ocorreu à coleta dos dados para obtenção da visão atual dos processos da instituição, o que auxiliou na obtenção dos requisitos do sistema e no entendimento do projeto e quais as mudanças o cliente espera alcançar nas atividades diárias da escola após a implantação do sistema.

3.1.1 Levantamento de Dados e Cenário Atual

Como objetivo coletar todas as informações relevantes sobre as funções do sistema e as regras de negócio sob as quais ele deve operar, foram realizadas reuniões com o diretor da escola, a partir das quais onde foi preenchido um questionário, (Apêndice A), para levantamento e entendimento das necessidades do sistema. Nas reuniões foi possível entender a situação atual da escola e a partir daí teve-se conhecimento que não há atualmente um sistema de gerenciamento informatizado. Espera-se, portanto, uma mudança organizacional com a implantação do sistema proposto na realização das demandas da instituição.

O Espaço Educacional Arco-Íris é uma escola que atende a uma demanda relativamente grande de alunos, oferecendo à população um serviço de ensino nos níveis infantil e fundamental. Ela conta com uma grande necessidade para a

organização de seus documentos que ainda são feitos de modo tradicional onde todas as operações diárias de expediente ocorrem de forma manual.

O diário escolar está disposto em forma de caderneta, onde o professor responsável faz o controle das notas, frequência e atividades de forma manual. Isso faz com que haja uma dificuldade quando se deseja ter um relatório do desempenho dos alunos.

Ao necessitar emitir um boletim do aluno, o funcionário responsável colhe as informações dos diários de classe e transcreve para o papel, para posteriormente ser entregue aos pais ou responsáveis pelos alunos. Isso faz com que haja um excesso de trabalho por parte dos funcionários e também um atraso na chegada destas informações importantes para se ter um acompanhamento do desenvolvimento escolar de cada discente.

A escola não possui cadastro em banco de dados dos alunos, professores, funcionários e pais ou responsáveis, já que todas as informações pessoais estão arquivadas em fichas de papel. Quando há necessidade de se obter ou relatório sobre algum destes, é necessário procurar em todas as fichas, para depois fornecer os dados solicitados. Também não há um controle automatizado de distribuição de turmas e disciplinas.

Por fim, a escola também não tem controle sobre o acesso e a manipulação de informações por parte de seus funcionários. Não é possível saber a origem da inconsistência de dados ou quem foi o responsável por tal, o que faz com que a escolar fique vulnerável a erros.

Todas essas características acabam deixando os processos diários totalmente sujeitos a inconsistências e falhas, impossibilitando a escola de obter um controle preciso e ágil das informações.

A fim de solucionar todos esses problemas, a escola Arco-Íris necessita da implantação de um sistema informatizado que automatize suas rotinas diárias, coletando, arquivando e manipulando as informações de forma segura e consistente.

3.1.2 Visão Geral do Sistema Proposto

O sistema de gestão escolar será responsável por informatizar as demandas de diárias da escola como: controle de frequência de alunos; notas dos alunos por disciplina e turmas; gera boletim do aluno no momento solicitado; cadastro de:

alunos, professores, funcionários; e também oferecer um melhor e mais preciso controle das informações.

No que diz respeito a acesso ao sistema, este terá três tipos de visões: funcionário, que terá acesso ao sistema por completo, podendo modificar dados cadastrais, emitir relatórios e visualizar todas as informações cadastradas, fazer cadastros diversos, inserir informações diversas ao sistema; professores, que terão acesso ao seu horário individual e ao diário escolar eletrônico, no qual poderão registrar as notas, o controle de frequência dos alunos e as atividades desenvolvidas em aula; e aluno, que terá acesso ao seu boletim e horário de aulas.

O sistema tem como objetivo agilizar todo o processo de controle e guarda de informações necessárias para um funcionamento adequado da escola, como também será responsável por automatizar o que antes era feito de forma manual e finalmente deverá gerar relatórios de diversas informações.

A exploração da potencialidade desses recursos abre uma nova visão na escola com relação às atividades realizadas, pois a gestão escolar informatizada prioriza o processo de vinculação entre seus alunos e todos envolvidos no ambiente educacional.

3.1.3 Objetivos do Sistema

Os principais objetivos do sistema serão:

- Cadastro de usuários;
- Cadastro de disciplinas e ano/série e seus horários e turmas;
- Matrícula dos alunos;
- Registrar e manter os dados do diário de classes através de um terminal em sala de aula;
- Avaliação dos professores feita pelos alunos de forma automatizada;
- Emitir relatórios diversos do sistema;
- Emitir relatório de desempenho dos alunos e professores;
- Emitir boletim e disponibilizar o horário das aulas para os alunos e professores;
- Emitir histórico escolar dos alunos.

3.1.4 Descrição do Escopo do Projeto

O projeto tem como objetivo o desenvolvimento de um sistema de gestão escolar, para informatizar as atividades executadas na administração e também fornecer as informações necessárias neste âmbito.

O produto que se deseja alcançar no final deste projeto é um sistema que automatize as atividades de gestão escolar e possibilite um sistema de relatórios, através do qual será possível recuperar as informações do sistema. Ao final do projeto espera-se ter um sistema que atenda as exigências impostas pelo cliente, que alie funcionalidade com bom desempenho, e que seja capaz de realizar as tarefas a ele impostas com eficácia.

3.1.5 Impactos Esperados pelo Projeto

- Automatização das atividades de expediente da escola;
- Economia do uso de papel;
- Centralização das informações em uma base de dados informatizada;
- Mais rapidez na recuperação de informações;
- Relatório de desenvolvimento de aluno e professor de forma automática;
- Diário de classe informatizado dando assim mais agilidade no seu registro.

Todos esses dados colhidos foram documentados utilizando o modelo do Documento de Visão de Negócio da Plataforma RUP que está disponível no Apêndice B.

3.2 Disciplina Requisitos

Esta disciplina acontece durante as fases de Concepção e Elaboração e para atingir os seus objetivos, é importante, antes de tudo, entender a definição e o escopo do problema que tentamos resolver com o sistema proposto.

O Modelo de Negócios desenvolvido durante a Modelagem do Negócio servirá como base de informações nesse fluxo de trabalho para obtenção dos requisitos do sistema. Nessa disciplina, foram identificadas e registradas as

funcionalidades que o sistema irá oferecer a seus usuários e também o modo como ele vai realizar suas operações. Apresentam-se no Quadro 03 os requisitos funcionais do sistema, que correspondem às funções que o sistema deve realizar.

Quadro 03 - Requisitos funcionais do sistema.

Código	Requisitos Funcionais
RF01	Efetuar *login* (para ter acesso ao sistema)
RF02	Cadastrar e manter (alunos, professores, funcionários)
RF03	Registrar disciplinas, turmas e ano/serie. (conjunto de registro)
RF04	Realizar matricula dos alunos. (efetua a matricula no ano letivo)
RF05	Registrar no diário de classe (registro de notas, freqüência dos alunos e atividades realizadas)
RF06	Registrar Horários (datas importantes, feriados, início e término dos semestres e ano letivo no sistema)
RF07	Realizar a avaliação dos professores de acordo com a série (ao final do ano letivo os alunos farão uma avaliação do professor)
RF08	Consultar / imprimir desempenho individual dos alunos para o Censo escolar (Esta consulta será separada por ano/serie)
RF09	Consultar / imprimir a avaliação dos professores (Visualização do resultado da avaliação dos professores dividido por ano/serie)
RF10	Consultar boletim do aluno (O aluno poderá acessar o sistema e visualizar as suas notas)
RF11	Enviar ou/e emitir boletim do aluno (envia o boletim para o e-mail dos pais, ou emitir a pedido dos pais ou aluno)
RF12	Emitir Histórico escolar
RF13	Emitir confirmação de matricula (no ato da matricula)
RF14	Consultar Horário Aluno (O aluno poderá entra no sistema e conferir seu horário de aulas)
RF15	Consultar Horário professor (O professor poderá entra no sistema e conferir seu horário de aulas)

Fonte: Autor

O Quadro 04 apresenta os requisitos não funcionais do sistema, que são regras colocadas sobre o modo como o sistema irá realizar os requisitos funcionais.

Quadro 04 - Requisitos não funcionais do sistema.

Código	Requisitos não funcionais
Usabilidade:	
RNF01	Separar a interface do restante da aplicação.
RNF02	A interface gráfica do sistema será baseada nas cores padrão da escola e dever ser intuitiva ao usuário.
Manutenibilidade	
RNF03	Uso de interfaces com ocultação de informações específicas sobre a implementação dos módulos.
RNF04	Uso de um intermediário para isolar o mecanismo de persistência de dados.
RNF05	Uso de um intermediário para tratar as requisições da interface.
Segurança:	
RNF06	Só será possível *logar* no sistema usuários já cadastrados.
RNF07	Autenticar usuários usando *login* e senha.
RNF08	Mostra as visualizações diferentes do sistema de acordo com o tipo de conta do usuário: Funcionário, Professor e Aluno.
Desempenho:	
RNF09	Estabelecer uma configuração de hardware mínima para comportar o sistema.
Disponibilidade:	
RNF10	O sistema será acessível de modo desktop conectado a uma intranet.
RNF11	As funcionalidades do sistema devem estar disponíveis em qualquer dia da semana.
Portabilidade:	
RNF12	Uso da linguagem Java e de bibliotecas e mecanismos de persistência capazes de rodar nos sistemas operacionais Windows e Linux.

Fonte: Autor

Os Diagramas de Caso de Uso são importantes no processo de desenvolvimento de um sistema, pois se tratam de um conjunto de ações que o sistema pode realizar, produzindo um resultado de valor observável para determinado ator, como também a maneira como eles interagem com o sistema. Na Figura 10 está representado o diagrama dos casos de uso do sistema onde é possível ter a visão de suas interações

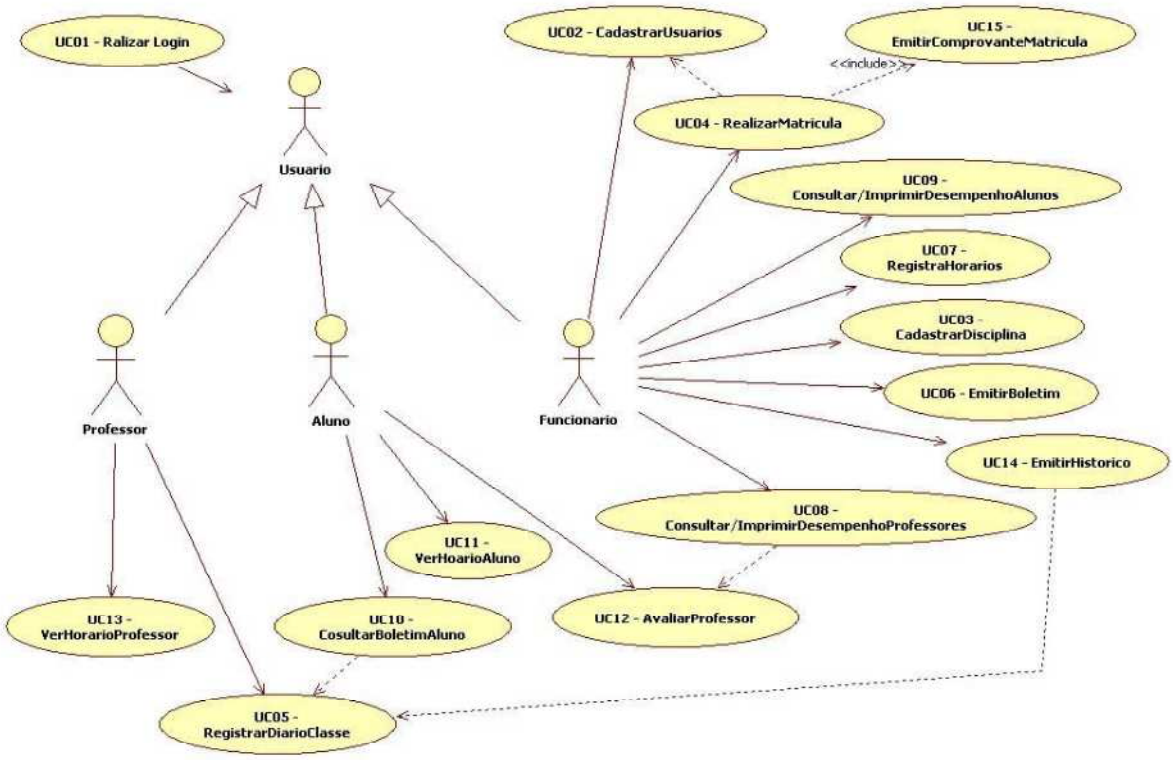


Figura 10 - Diagrama de Caso de Uso do sistema
Fonte: Autor do projeto

No Quadro 05mostra-se uma relação simplificada contendo os casos de uso identificados e seus respectivos atores.

Quadro 05 - Casos de Uso do sistema

ID	Nome	Ator(es)	Ref. ao Requisito
UC01	Realizar *login*	Funcionário, alunos e professores	RF01
UC02	Cadastrar Usuários	Funcionário	RF02
UC03	Cadastrar Disciplinas	Funcionário	RF03
UC04	Realizar Matricular	Funcionário	RF04 e RF15
UC05	Registrar Diário de Classe	Professor	RF05
UC06	Emitir Boletim	Funcionário	RF11
UC07	Registrar Horários	Funcionário	RF06
UC08	Consultar/Imprimir desempenho dos professores	Funcionário	RF08
UC09	Consultar/Imprimir desempenho dos alunos	Funcionário	RF09
UC10	Consultar Boletim do aluno	Aluno	RF10
UC11	Ver Horário Aluno	Aluno	RF14
UC12	Avaliar Professor	Aluno	RF07
UC13	Ver Horário Professor	Professor	RF15
UC14	Emitir Histórico	Funcionário	RF12
UC15	Emitir comprovante de matrícula	Funcionário	RF13

Fonte: Autor

No documento de Casos de Uso (Apêndice C) têm-se as fichas desses, levantados contendo uma breve descrição e um fluxo de informações, denominado de fluxo básico, como também os fluxos alternativos, caso o básico não seja seguido corretamente. A descrição desses fluxos de eventos é um conteúdo importante de um caso de uso, pois o mesmo é definido em linguagem natural simples, geralmente com um vocabulário reduzido e consistente.

3.2.3 Disciplina Análise e *Design*

A disciplina Análise *e Design* têm seu esforço mais concentrado na fase de Elaboração do projeto, e possui como objetivos: transformar os requisitos em um *design* do sistema a ser criado; desenvolver uma arquitetura para o sistema; adaptar o *design* para corresponder ao ambiente de implementação.

Ela pode ser dividida em duas fases, Análise e *Design*. Na Análise o foco está nos requisitos funcionais do sistema, definindo um modelo mais simples para o mesmo, colhido através dos casos de uso. Já o *Design* possui foco na solução escolhida, sendo a mesma dirigida pelos requisitos não-funcionais do sistema.

Entre seus artefatos, destaca-se o Documento de Arquitetura de Software, vide Apêndice D. No decorrer desta seção serão descritas essas duas fases em separado.

3.2.3.1 Análise

O papel do fluxo de análise consiste, entre outras tarefas, na análise dos requisitos do modelo de casos de uso objetivando a coleta de informações necessárias para a definição das classes que representam o domínio do negócio em questão. Essa atividade vai identificar quais as classes necessárias para implementação do projeto.

O projeto consistirá na melhoria do modelo gerado a partir da atividade de análise, a ponto de ser gerada facilmente a implementação do sistema. Seu objetivo é levar em consideração restrições inerentes à tecnologia a ser utilizada na solução de um problema. O diagrama de classe da Figura 11 representa as classes de domínio, ou seja, as classes entidades do sistema. Após esse modelo será o gerado o Modelo de Design.

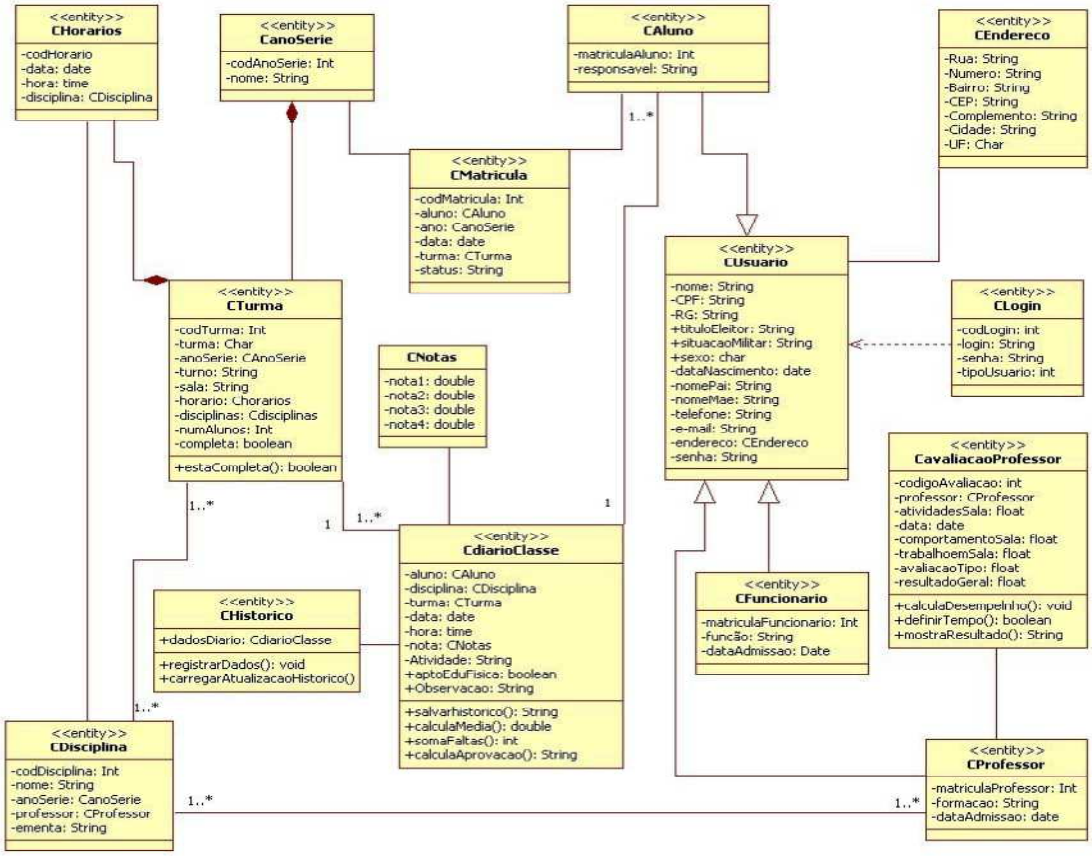


Figura 11 - Diagrama Classes das Entidades
Fonte: Autor do projeto

3.2.3.2 Design

Na modelagem arquitetural do sistema foi aplicado um *padrão de projeto* MVC (*Model, View, Controller*). A idéia do MVC é dividir o programa em três conjuntos, cada um com uma determinada atividade. Assim, a camada de *View* seria responsável pela apresentação dos dados; *Controller* seria incumbido de receber os dados inseridos pelo usuário, manipulados utilizando-se da camada de Modelo, e retornar alguma informação para a camada de Visão; o *Model* teria a função de definir o modo como os dados são organizados no meio persistente (GAMMA, 2000).

Será utilizado o padrão de projeto *Facade*, que é um padrão estrutural que fornece uma interface unificada para um conjunto de interfaces em um subsistema. A idéia é servir de interface entre o *View* e o *Controller*, definindo um único ponto de acesso de nível mais elevado que faz com que o subsistema fique mais fácil de ser utilizado (FOWLER, 2003).

Será também utilizado outro padrão o DAO *(Data Access Object) que* é utilizado para abstrair da camada de negócios as fontes de dados e persistência. Assim, um objeto DAO provê métodos para adicionar, editar, excluir, consultar objetos de dados independentemente da forma de persistência dos dados. (GAMMA, 2000).

Segundo Fowler (2003), a utilização desses padrões tem como objetivo facilitar o reuso do código, a modularização dos sistemas,como também o desacoplamento dos módulos do sistema, facilitando a manutenção do código, além de deixar mais compreensível o desenvolvimento do sistema. A Figura 12 mostra o diagrama de pacotes com um foco especial a camada de Visualização e sua comunicação com a classe do *Facade* da camada Controle e as classes DAO.

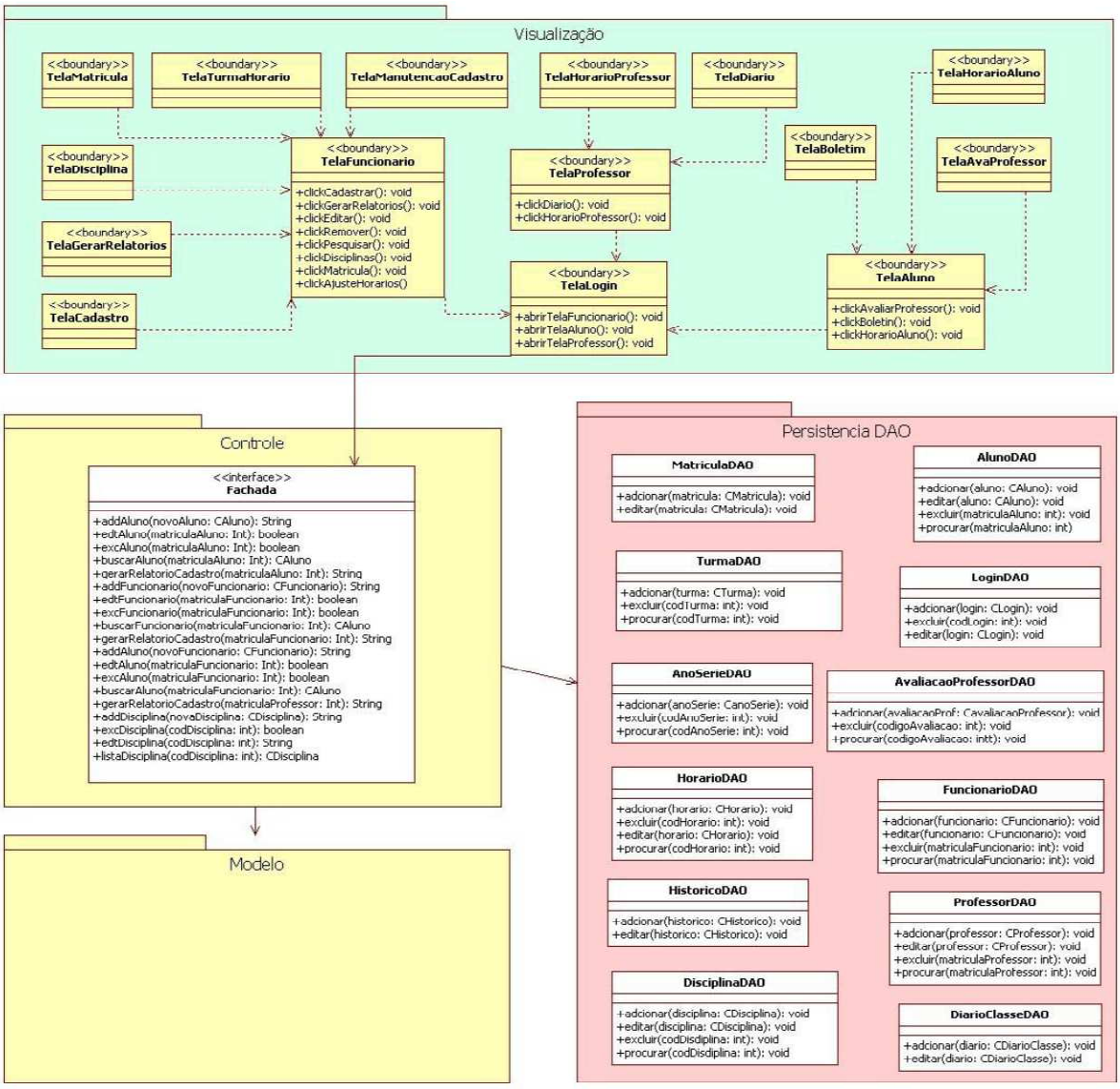


Figura 12 - Diagrama de Pacotes
Fonte: Autor do projeto

A Figura 13 mostra o diagrama de classes completo da camada de Controle onde pode-se encontrar a classe *Facade* e as outras classes que a compõe.

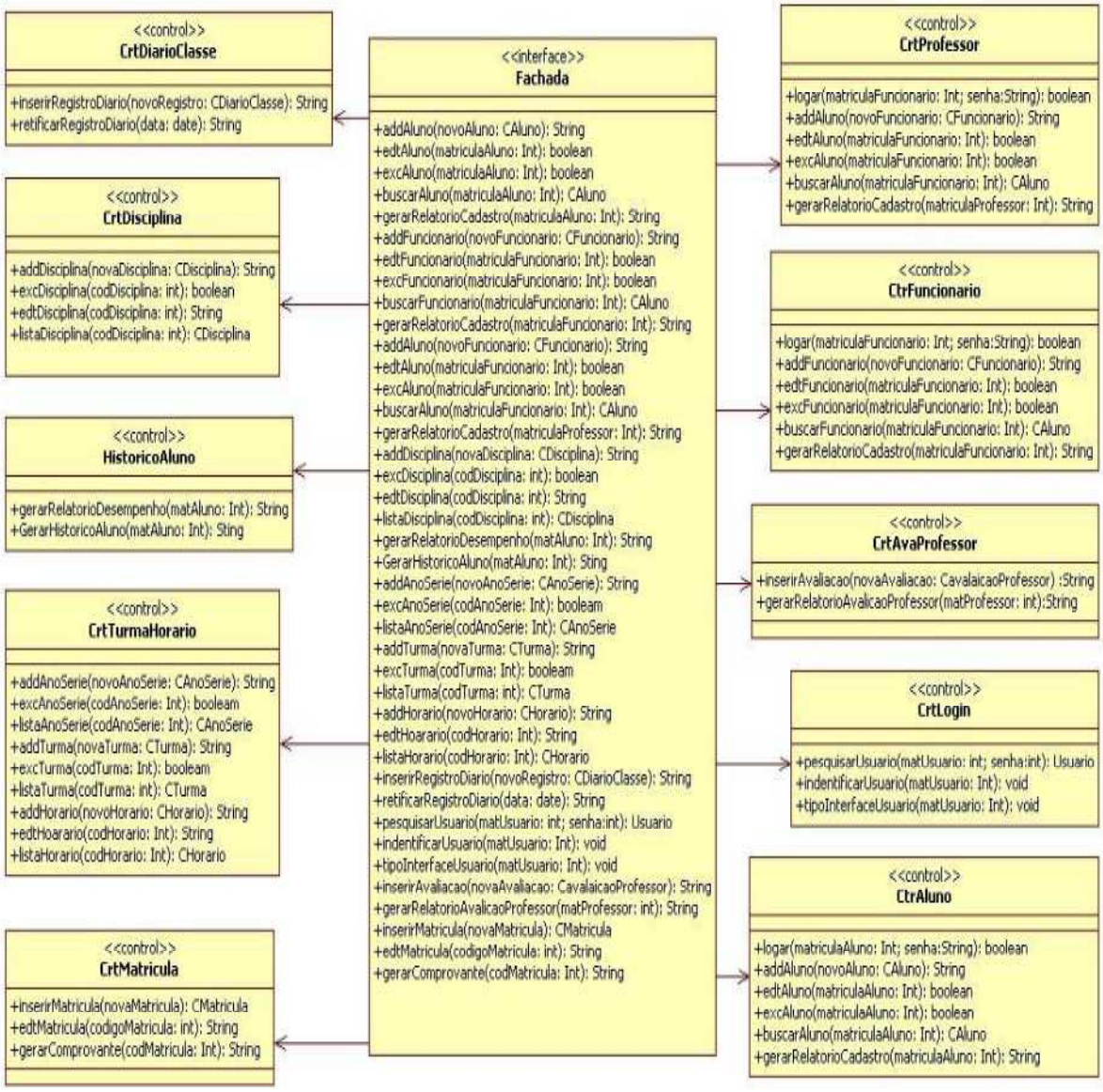


Figura 13 - Diagrama de Classes da camada Controle
Fonte: Autor do projeto

A representação da arquitetura está baseada na referência J2EE (*Java Enterprise Edition)*. Assim, muitos dos mecanismos técnicos adotados, como persistência, ambiente e distribuição, utilizaram esta arquitetura de referência.Nessa referencia o padrão MVC define uma camada de Visualização utilizando JSP (*Java Server Pages*) e *Servlet*, e na de Controle EJB (*Enterprise Java Beans)* e objetos de negócio como também uma de persistência para permitir acesso ao banco de dados (J2EE, 2014).

No lado cliente, as páginas HTML (*HyperText Markup Language)*, que é uma linguagem de marcação utilizada para produzir páginas interpretadas pelos navegadores. Será utilizado JavaScript para validação do preenchimento e algumas outras funções, como por exemplo, verificar se todos os campos obrigatórios de um formulário foram preenchidos, validar preenchimento e assim evitando tráfegos desnecessários na rede(JAVASCRIPT, 2014).

No lado servidor, será utilizado o Tomcat, um servidor de aplicações Java para Web (*World Wide Web)*, que é entendido como um sistema de documentos em hipermídia (reunião de várias mídias num ambiente computacional, suportada por sistemas eletrônicos de comunicação) são interligados e executados na rede. Ele é responsável por aceitar pedidos de clientes (navegadores) e servi-los com respostas solicitadas (TOMCAT, 2014; BAIRON, 2011).

A Figura 14 mostra como estão dispostos os componentes em seus respectivos nós de implantação.

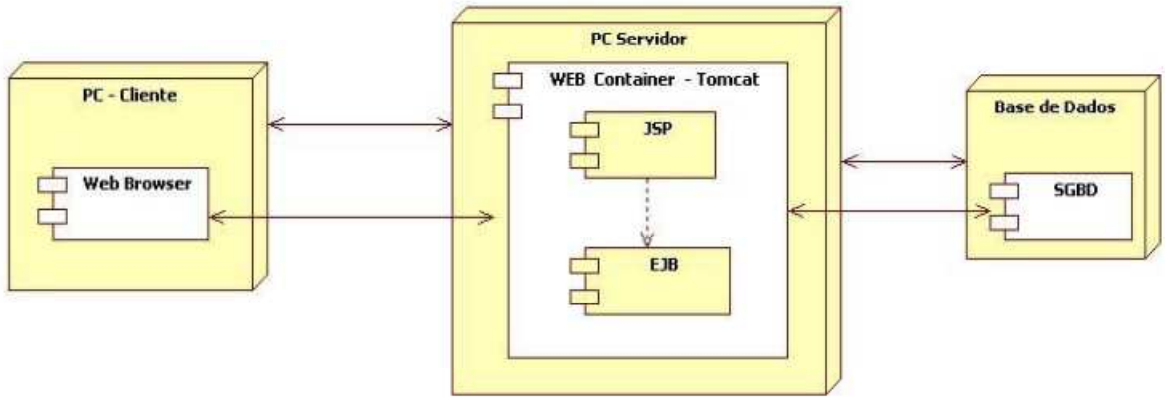


Figura 14 - Diagrama de Implantação
Fonte: Autor do projeto

O RUP está sendo utilizado de forma personalizada neste projeto,o que mostra a importância da modelagem de software em algumas de suas disciplinas.Devido a sua natureza incremental, os artefatos elaborados durante a execução deste projeto serão modificados de acordo com as necessidades de mudança, adaptação e correções de erros e falhas.As disciplinas de Implementação, Teste, Implantação, Gerenciamento de Configurações e Mudanças, Gerenciamento de Projeto e Ambiente não foram utilizadas, pois não eram objetos de estudo do tema proposto neste trabalho.

4 CONSIDERAÇÕES FINAIS

Com os estudos realizados durante o desenvolvimento deste trabalho, pode-se constatar que, com a adoção do RUP como metodologia de desenvolvimento de sistemas, é possível ter uma visão geral de vários aspectos do projeto suportado por uma boa modelagem proporcionada pela UML, facilitando assim uma compreensão uniforme do sistema como um todo, já que muitas vezes o projeto deverá ser entendido por diversos *stakeholders* de várias áreas e todos poderão ter um bom entendimento do projeto.

Sabe-se que a concepção e a elaboração de um sistema são fases importantes do desenvolvimento, pois essas darão base ao projeto de *software*, nestas fases há todo um conjunto de artefatos que juntos dão uma visão do modelo do sistema e assim facilita a sua posterior codificação, facilitando o entendendo das reais necessidades do cliente.

No decorrer do trabalho pôde-se mostrar a modelagem de *software* nas disciplinas de Modelagem de Negócios, Requisito e Análise e *Design*. Nota-se que ela está muito presente nas fases iniciais de um projeto, por ter utilidade para ilustrar graficamente as informações relevantes do sistema, facilitando significativamente o entendimento do produto como um todo.

Durante o desenvolvimento deste trabalho foi possível elaborar alguns artefatos do RUP, lembrando que alguns artefatos são opcionais. Assim foi possível ter um conhecimento maior dos objetivos de negócio, dos requisitos do sistema e das partes do sistema e seus acoplamentos, através de diagramas da UML. Utilizar o RUP em projetos pequenos é possível devido à flexibilidade proporcionada através de sua estrutura e documentação, já que muitos dos seus artefatos são opcionais.

O fato que se deve levar em consideração é que um processo que pede a elaboração de vários artefatos de documentação como RUP tem seus pontos positivos e negativos. Por exemplo, o tempo gasto na elaboração desses artefatos poderia ser utilizado para a implementação do produto em si. Isso pode ser visto no emprego da metodologia utilizada pelos modelos ágeis que estão cada vez mais sendo adotados.

Uma proposta pra trabalhos futuros seria realizar estudos para analisar se realmente a elaboração destes artefatos ocasiona atrasos no projeto, ou se ela é útil quando se deseja modificar o software no futuro por outros profissionais. Outra

proposta é fazer uma comparação entre a modelagem em metodologias ágeis e em metodologias clássicas. O que pode responder se realmente e viável elaborar a documentação e modelar o sistema durante o desenvolvimento do produto em si.

REFERÊNCIAS BIBLIOGRÁFICAS

BAIRON, Sérgio. Hipermídia. São Paulo Brasiliense, 2011.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML: guia do usuário**. Ed. Campus, Rio de Janeiro: Elsevier, 2005.

ERIKSSON, H. E.; PENKER, M. **Business Modeling with UML**. New York: John Wiley & Sons, 2000. 459 p.

FOWLER, Martin. **Patterns of Enterprise Application Architecture**. 2. ed. Addison Wesley. 2003.

FUNKS, H; RAPOSO, A.B. & Gerosa, M.A. **Engenharia de groupware:** desenvolvimento de aplicações colaborativas. XXI Jornada de Atualização em Informática, Anais do XXII Congresso da Sociedade Brasileira de Computação, V2, Cap. 3, ISBN 85-88442-24-8. Ano 2002.

GAMMA, Erich et al. ***Padrões de Projeto:*** *Soluções reutilizáveis de software Orientado a Objetos.* Porto Alegre: Bookman, 2000.

J2EE Documents, http://docs.oracle.com/javaee/1.4/tutorial/doc/, acessado em 20 setembro de 2014.

JavaScript, http://www.w3schools.com/js/default.asp, acessado em25 de setembro de 2014.

Kruchten, Philippe, **Introdução ao RUP – Rational Unified Process**, 3. ed. – Rio de Janeiro: Ciência Moderna, 2003.

MAKINO. A. **Abordagem da metodologia RUP no desenvolvimento de um sistema de gestão comercial**. 85 f. (Trabalho de Conclusão de Curso). Faculdade de Tecnologia de Taquaritinga. Centro Estadual De Educação Tecnológica "Paula Souza". Taquaritinga. 2009.

MARTINS, José Carlos Cordeiro. **Gerenciando Projetos de Desenvolvimento de Software** – 2° e 4° edições, 2007.

OMG, Object Management Group. Disponível em: http://www.omg.org,acessado em20 de setembro de 2014.

PRESSMAN, R.S. **Engenharia de Software**. 6ª Ed, McGraw-Hill, 2006.

PURRI, Marcelle Cristina M. e S**. Estudo e propostas iniciais para a definição de um processo de desenvolvimento para *software* científico.** Dissertação (mestrado) - Mestrado em Modelagem Matemática e Computacional (MMC), Centro Federal de Educação Tecnológica de Minas Gerais, Belo Horizonte; 2006.

Rational Unified Process: Visão geral. Disponível em http://www.wthreex.com/rup/,acessado em 20 de setembro de 2014.

SCOTT, Kendall. **O PROCESSO UNIFICADO EXPLICADO: UML**, 2006.

SILVA, A. M. R. da**;** VIDEIRA, C. A. E. **UML, Metodologias e Ferramentas CASE:** Linguagem de Modelação UML, Metodologias e Ferramentas CASE na Concepção e Desenvolvimento de Software**.** Porto – Lisboa: Centro Atlântico, 2001. 578 p. 1ª edição.

SILVA, D. M. Guia de consulta rápida: UML. São Paulo: Novatec Editora, 2001.

SOMMERVILLE, I. **Software Engineering.** 9. ed. [s.l]: Pearson Education Inc, 2011.

TAMAKI, P. A. O. **Uma Extensão de RUP com ênfase no gerenciamento de projetos do PMBOK baseada em *Process Patterns*.** 2007. 211 f. Dissertação (Mestre em Engenharia) – Escola Politécnica da Universidade de São Paulo, São Paulo, 2007.

Tomcat, http://tomcat.apache.org/ 25, acessado em 25 de setembro de 2014.

VICENTE, Leonardo Silva Sciammarella**, Modelagem de Processos e Linguagem de Modelagem Unificada (UML) uma análise crítica**, Rio de Janeiro, 2003.

APÊNDICE A - Questionário

Questionário para obtenção dos requisitos

No dia 19 de setembro de 2014, o diretor do Espaço Educacional Arco-Íris responde a um questionário para a obtenção dos requisitos de um projeto de software para o desenvolvimento de um sistema de gestão escolar. O Diretor se mostrou entusiasmado com possibilidade de informatizar as atividades da escola. Foram levantadas questões para saber o que o cliente no espera que o sistema automatizado faça.

1. Como é a situação atual da escola, ou seja, como é organizado e realizado as atividades diárias, tem algum processo já informatizado?

Diretor (cliente): Na escola as atividades diárias são realizadas ainda de modo tradicional, ou seja, em forma de fichas de papel para cadastro de alunos, diário escolar onde cada professor é responsável pelo controle das notas de cada aluno, frequência e registro de atividades, o boletim do aluno também é feito de forma manual, o computador só é utilizado para acessar a internet e digitação de documentos não há um sistema que gerenciamento para as atividades da escola isso faz com que haja um excesso de trabalho por parte dos funcionários e também um atraso atividades diárias.

2. Quais são as tarefas que o sistema devera informatizar?

Diretor (cliente): O sistema deve informatizar as atividades diárias da escola desde a parte administrativa até o controle de aulas, então o sistema deverá ter o controle de notas, aulas, atividades desenvolvidas em sala de aula e controle de frequência isso através de um diário eletrônico, como também deverá ser capaz de cadastrar e manter em um banco de dados as informações dos funcionários, alunos, professores, diário eletrônico, como também será responsável por coletar informações pra um relatório do desempenho dos alunos e professores

3. Quais os usuários irão interagir com o sistema?

Diretor (cliente): O sistema deverá dar acesso a três tipos de usuários: o funcionário, que terá acesso a todas as informações do sistema e será responsável

pelo cadastro dos outros usuários e impressão dos relatórios do sistema; o professor, que terá acesso ao diário escolar onde poderá registrar as notas, frequência e atividades; e o aluno, que poderá acessa o boletim do aluno e um formulário para a avaliação do professor durante um tempo pré-determinado.

4. Como será feito o acesso ao sistema?

Diretor (cliente):Só poderá ter acesso ao sistema quem já estiver cadastro nele.Para acessar o sistema o usuário deverá fazer um *login* através de senha e nome.

5. Como será interface do usuário?

Diretor (cliente): A idéia é que o sistema fique com as cores padrão da escola e, a respeito da questão das visualizações do sistema, cada usuário terá uma visualização deferente dependendo do seu tipo de conta.Serão disponibilizadas para visualizações apenas as funções que o usuário tem autorização para executar.

6. Como será disponibilizado o sistema?

Diretor (cliente): O sistema poderá ser acessado de qualquer computador dentro da escola, pois eles serão interligados através de uma intranet.

7. Quais os computadores e equipamentos que tem na escola e quais as configurações?

Diretor (cliente): A escola conta uma multifuncional e quatro computadores de configurações diversas - a intenção é que o sistema possa ser executado em qualquer sistema operacional.

APÊNDICE B - Documento de Visão do Negócio

Sistema de Gestão Escolar (SGE Arco-Íris)

Documento de Visão do Negócio

Versão 1.1

Histórico da Revisão

Data	Versão	Descrição	Autor
23/02/2015	1.0	Versão Inicial do Documento de Visão	Damião Rodrigues
06/05/2015	1.1	Versão Final do Documento de Visão – Atualização.	Damião Rodrigues, Rodrigo Alves Costa

1 INTRODUÇÃO

Este documento apresenta um planejamento de alto nível para o projeto de criação de um Sistema de Gestão Escolar (SGE Arco-Íris), solicitado pelo cliente Sr. Francisco Nivaldo, diretor da escola, apresentando os problemas a serem solucionadas com a implantação do sistema, como também atender as necessidades de negócio da instituição através das funcionalidades esperadas no sistema.

1.1 Objetivos

O presente documento tem por objetivo descrever as principais características do sistema a ser desenvolvido, possibilitando uma melhor noção das necessidades e características dos requisitos do sistema, para uma posterior implementação. Dessa forma, este documento visa esclarecer os objetivos e decisões que serão tomadas na construção do sistema.

1.2 Escopo

O SGE Arco-Íris será responsável por informatizar as atividades realizadas na escola na qual será implantado. Desde controle de frequência de alunos, atividades desenvolvidas na sala de aula como também as de expediente que são executadas na administração escolar, será possível também fornecer informações como: desempenho, histórico escolar e frequência de alunos. Este sistema disponibilizará também um controle melhor e mais preciso das informações importantes para o ambiente escolar.

2 POSICIONAMENTO

Por meio desse documento de visão teremos uma noção geral do nosso problema e apresentaremos prováveis soluções.

2.1 Descrição do Problema

A referida escola não possui sistema para gerenciamento das informações importantes para suas atividades que ainda estão organizadas em fichas de papel.Também não tem controle sobre o acesso e a manipulação de informações por parte de seus funcionários. Não é possível saber a origem da inconsistência de dados ou quem foi o responsável por tal, o que faz com que ela fique vulnerável a erros.

Todas essas características acabam deixando os processos diários totalmente sujeitos a inconsistências e falhas, impossibilitando a escola de obter um controle preciso e ágil das informações.

O problema de	Dificuldade em manipular e recuperar informações importantes para as atividades diárias, falta de controle sobre os documentos acessados pelos funcionários, falta de relatório de desempenho dos alunos e professores.
Afeta	Funcionários, gestores, alunos e responsáveis
uma boa solução seria	O desenvolvimento de um sistema que informatize estas atividades.

2.2 Sentença de Posição do Produto

Para	Diretor da escola
O (nome do produto)	Sistema de Gestão Escolar (SGE Arco-Íris)
Que	Possam interagir da melhor forma através do sistema,tendo acesso a informações essenciais.
Diferente de	Produtos disponibilizados no mercado que traz soluções prontas onde a escola é que tem que se adaptar ao produto.
Nosso produto	Será desenvolvido atendendo as necessidades da instituição. Trará funcionalidades bem definidas para cada usuário do sistema (funcionários, professores e alunos), disponibilizará informações das disciplinas, alunos e professores, permitirá uma avaliação de desempenho dos professores e alunos, e facilitará o processo da matrícula.

3 DESCRIÇÕES DOS ENVOLVIDOS E DOS USUÁRIOS

Os principais envolvidos neste projeto estão descritos abaixo.

3.1 Resumo dos Envolvidos

Nome	Descrição	Responsabilidades
Francisco Nivaldo	Cliente	• Sugerir as funcionalidades do sistema; e • Monitorar o andamento do projeto.
Damião Rodrigues de Sousa	Desenvolvedor	• Levantamento de requisitos segundo RUP; • Análise e desenho da solução segundo RUP; • Garantir a documentação e andamento do projeto segundo a metodologia RUP; • Implementação do sistema; • Teste do sistema • Implantação no ambiente; • Treinamento dos usuários
Rodrigo Alves Costa	Gerente de Projeto	• Responsável pelo desenvolvimento do sistema; • Responsável pelo cronograma e eventuais revisões de escopo; • Administração do plano e programa de trabalho; • Execução do cronograma.

3.2 Resumo dos Usuários do Sistema

Nome	Descrição	Responsabilidades
Alunos	Usuário cadastrado como tal e que esta ou já esteve matriculado em ano/serie.	Terá acesso as suas respectivas funções: boletins, horário e avaliar desempenho de professores.
Professores	Usuário cadastrado como tal ao qual esteja lecionando alguma disciplina na escola.	Terá acesso as suas respectivas funções: registrar o diário de classe e ver seu horário individual.
Funcionários	Usuário responsável por cadastros em geral, matrículas e expedição de relatórios.	Tem acesso a suas respectivas funções.

3.3 Ambiente dos usuários

O acesso deve ser possível que qualquer máquina conectada a intranet da escola, independentemente do sistema operacional que ela possua.

4 FUNCIONALIDADES

Funcionalidade	Descrição
Cadastro de usuários	O sistema será capaz de manter o cadastro dos usuários: alunos, professores, funcionários, em um banco de dados, onde será possível uma recuperação fácil destes dados quando solicitado.
Cadastro de disciplinas e ano/série e seus horários.	O sistema será capaz de manter o cadastro das disciplinas, descriminadas por ano/série deverá conter também o professor que a leciona e seu respectivo horário.
Matrícula dos alunos	O sistema será capaz de realizar a matriculas dos alunos em seus respectivos ano/série e turmas disponíveis. Ao final ele emitirá um comprovante de matrícula.
Registra e manter os dados do diário de classes	O sistema disponibilizará o diário de classe onde será possível ter um controle e persistência da frequência dos alunos, notas e atividades realizadas em sala de aula.
Avaliação dos professores	Ao final de cada semestre o sistema disponibilizará um questionário onde os alunos de acordo com sua idade e ano/série farão uma avaliação dos professores que o lecionaram.
Emitir boletim e histórico do aluno	No sistema será possível emitir boletim do aluno pelo funcionário ou pelo próprio aluno ou funcionário, e o histórico escolar será emitido mediante solicitação ao funcionário.
Emitir Relatórios	O sistema deverá ter a opção de poder emitir relatórios diversos, como: desempenho dos alunos, desempenho dos professores, resumo do diário escolar e dados cadastrados dos usuários.

5 RESTRIÇÕES

- Para o desenvolvimento só serão utilizadas ferramentas *open source*.
- Os prazos de entrega devem ser respeitados.
- O sistema deve seguir as regras de negócio previamente analisadas e discutidas, sem alterar o funcionamento atual do negócio.

6 OUTROS REQUISITOS

Nesta parte será descritos os padrões aplicáveis, requisitos de hardware ou de plataforma, requisitos de desempenho e requisitos ambientais.

6.1 Padrões Aplicáveis

- O sistema deve ser desenvolvido em camadas com um bom acoplamento entre elas.
- O modelo de interface deverá ser WEB de forma a suportar os seguintes navegadores: Mozilla Firefox, Google Chrome.
- A arquitetura deve seguir o padrão J2EE.
- TomCat como servidor WEB (contêiner WEB)
- Persistência de tipagem (ex: formato de CPF, Data) devem ser feitos no cliente via JavaScript.
- Persistência de obrigatoriedade, como verificar se todos os campos obrigatórios de um formulário foram preenchidos pode ser feito no cliente via JavaScript.

6.2 Requisitos do Sistema

O sistema será capaz de rodar em máquinas de configuração de hardware mínima para os padrões de atuais, sendo necessário um sistema operacional (Windows ou Linux) e um navegador de internet será necessária conectividade entre os computadores em uma intranet. Uma máquina com configuração mais avançada será escolhida para hospedar o servidor de aplicação e de banco de dados.

APÊNDICE C - Casos de Uso

CASOS DE USO

Atores

- Aluno: Caracteriza-se como aluno o usuário que está ou já esteve matriculado em turmas, este terá acesso as suas respectivas funções.
- Professor: Caracteriza-se como professor o usuário ao qual esteja lecionando alguma disciplina, este terá acesso as suas respectivas funções.
- Funcionário: Caracteriza-se como funcionário o usuário responsável pelo cadastro em geral e expedição de relatórios tem acesso a suas respectivas funções.

Lista de Casos de Uso

ID	Nome
UC01	Realizar *login*
UC02	Cadastrar Usuários
UC03	Cadastrar Disciplinas
UC04	Realizar Matricular
UC05	Registrar diário de Classe
UC06	Emitir Boletim
UC07	Registrar Horários
UC08	Consultar/Imprimir desempenho dos professores
UC09	Consultar/Imprimir desempenho dos alunos
UC10	Consultar Boletim do aluno
UC11	Ver Horário Aluno
UC12	Avaliar Professor
UC13	Ver Horário Professor
UC14	Emitir Histórico
UC15	Emitir comprovante de matrícula

A seguir será mostrado as fichas de especificação de cada caso de uso, onde será possível ter um melhor esclarecimento de como cada caso de uso irá atuar dentro do sistema

Fichas de Especificação dos Casos de Usos

CASO DE USO: REALIZAR LOGIN
ID: UC01
Atores: Funcionários, professores, alunos
Descrição: Para o usuário ter acesso as funções do sistema e necessário esta logado nele utilizando para isto o *login* e a senha.
Precondições: O usuário já está cadastrado no sistema.
Fluxo principal: Este caso de uso se inicia quando qualquer usuário em geral, quer acessar o sistema. O usuário acessa a página inicial do sistema. Fornece seus dados de *login* * e senha * e clica em Logar. O sistema verificará se o usuário é cadastrado e se o mesmo possui permissão para acessar o sistema Após isto, o usuário terá acesso à página principal do sistema. Após verificar as permissões e montar o menu de opções na tela, o sistema buscará todas as tarefas do usuário em questão ** Campos obrigatórios*
Pós-condições:A tela principal foi exibida e o usuário possui acesso a todas as operações que lhe foram concedidas.
Fluxo alternativo: <u>Login ou senha inválidos</u> Este fluxo ocorre quando o usuário digita a senha ou o login incorretamente. Emitir mensagem: "*Login* ou senha inválido". <u>Dados Incompletos</u> Este fluxo ocorre quando algum campo obrigatório não foi preenchido no formulário. Emitir mensagem dos dados que estão faltando e colocar o foco no primeiro campo incorreto. <u>Sem permissão</u> Este fluxo ocorre quando o usuário não tem permissão para acessar o sistema. Emitir mensagem "Você não possui permissão para acessar o sistema".

CASO DE USO: CADASTRAR USUÁRIO

ID: UC02

Atores: Funcionários

Descrição: Este caso de uso tem como finalidade cadastrar, alterar ou excluir USUÁRIOS que utilizarão o sistema

Precondições:O funcionário deve estar logado no sistema.

Fluxo principal:

Este caso de uso se inicia quando o funcionário vai cadastrar alterar ou excluir um funcionário.

Cadastrar Usuário
O funcionário deverá informar os seguintes dados do usuário em um formulário:

Nome * Rua *Telefone *
Filiação* Bairro *E-mail
CPF * CEP * Senha *
RG * Estado * *Login*
Data de nascimento *Cidade *Tipo usuário *

* *Campos obrigatórios*

Depois disso, deverá clicar no botão Incluir. Para a inclusão do usuário no sistema.

Os tipos de usuário são:
1 – Funcionários;
2 – Professor;
3 – Aluno.

Alterar Dados dos Funcionários
O funcionário deverá selecionar o usuário que deseja alterar. Automaticamente os dados daquele usuário serão carregados na tela.
Desativar usuário
O funcionário deverá selecionar o usuário que deseja excluir e clicar no botão Excluir .

Pós-condições:O sistema será atualizado com as informações

Fluxo alternativo:

Usuário já cadastrado
Este fluxo ocorre quando o funcionário tenta inserir um usuário com o código igual ao que já se encontra cadastrado no banco de dados. Emitir mensagem: "Usuário já cadastrado".

Dados Incompletos
Este fluxo ocorre quando algum campo obrigatório não foi preenchido no formulário. Emitir mensagem dos campos que estão faltando e colocar o foco no primeiro campo incorreto.

Usuário Não Selecionado
Este fluxo ocorre quando não foi selecionado nenhum usuário para as opções de alterar ou excluir.

CASO DE USO: CADASTRAR DISCIPLINA

ID: UC03

Atores: Funcionários

Descrição: Este caso de uso tem como finalidade cadastrar, alterar ou excluir o conjunto de dados das disciplinas**.**

Precondições:O funcionário deve estar logado no sistema**.**

Fluxo principal:

Cadastrar Disciplina
O funcionário deverá informar os seguintes dados do usuário em um formulário:

Código*
Nome *
Ementa
Série*

* *Campos obrigatórios*

Depois disso, deverá clicar no botão Registrar. Para a inclusão da disciplina no sistema.

Alterar dados das disciplinas
O funcionário deverá selecionar a disciplina que deseja alterar. Automaticamente os dados daquela disciplina serão carregados na tela. O funcionário alterará os dados necessários e clicará no botão Alterar.

Desativar Disciplina
O funcionário deverá selecionar a disciplina que deseja excluir e clicar no botão Excluir para que os dados daquela disciplina possam ser desativados no sistema.

Pós-condições: O sistema será atualizado com as informações

Fluxo alternativo:

Disciplina já cadastrada
Este fluxo ocorre quando o funcionário tenta inserir uma disciplina com o código igual ao que já se encontra cadastrado no banco de dados. Emitir mensagem: “disciplina já cadastrada”.

Dados Incompletos
Este fluxo ocorre quando algum campo obrigatório não foi preenchido no formulário.
Emitir mensagem dos campos que estão faltando e colocar o foco no primeiro campo incorreto.

Disciplina Não Selecionada
Este fluxo ocorre quando não foi selecionada nenhuma disciplina para as opções de alterar ou excluir**.**

CASO DE USO: REALIZAR MATRICULA
ID: UC04
Atores: Funcionários
Descrição: Este caso de uso tem como finalidade registrar a matricula do aluno.
Precondições:O funcionário deve estar logado no sistema.O aluno já deve estar cadastrado no sistema e as disciplinas do respectivo ano/serie também devem estar cadastradas.
Fluxo principal: **Registrar matricula** O funcionário deverá informar os seguintes dados do usuário em um formulário: Nome do aluno* Ano/serie* * *Campos obrigatórios* Será carregado todas as disciplinas do ano/serie correspondente e uma tabela, depois deverá clicar no botão Registrar. Para registrar a matricula no sistema. Conseqüentemente será disponibilizado o comprovante de matricula (UC14) como também o boleto (UC15) para pagamento das mensalidades. **Cancelar Matricula** O funcionário deverá selecionar nome do aluno e clicar no botão Cancelar matricula.
Pós-condições:O aluno será alocado em uma turma e estará matriculado no ano letivo correspondente.
Fluxo alternativo: Disciplinas não cadastrada Este fluxo ocorre quando o funcionário tenta inserir as disciplinas, e não há disciplina disponível para o ano/serie do aluno Aluno não cadastrada Este fluxo ocorre quando o funcionário tenta inserir um aluno, e ele não esta cadastrado no sistema.

CASO DE USO: REGISTRA DIARIO DE CLASSE
ID: UC05
Atores: Professor
Descrição: Este caso de uso tem como finalidade registrar as notas, freqüência do aluno e atividades realizadas na aula.
Precondições:O professor deve estar logado no sistema, o aluno deve estar matriculado. Só será disponibilizado as datas do ano letivo já cadastradas pelo funcionário (UC06).
Fluxo principal: **Registrar Notas** O Professor deverá informar as notas das atividades realizadas pelos alunos **Registrar Freqüência** O Professor deverá realizar a freqüência dos alunos e registrar os dados. **Registrar Atividades** O Professor deverá registrar as atividades realizadas na aula. Depois de feito esses registros o professor deverá clicar no botão Registrar dados. Para a inclusão dos dados no diário escolar **Alterar Dados do diário** O professor poderá alterar o diário de acordo com sua necessidade ate o termino do ano letivo depois não serão mais possível alterar.
Pós-condições: O sistema será atualizado com as informações.
Fluxo alternativo: Nenhuma

CASO DE USO: Emitir boletim do aluno
ID: UC06
Atores: Funcionário
Descrição: Este caso de uso tem como finalidade enviar o boletim do aluno ao final de cada bimestre ao e-mail cadastrado como também imprimir o mesmo a pedido.
Precondições: O funcionário deve estar logado no sistema. O professor deve ter registrado as notas no diário escolar, caso de uso UC05.
Fluxo principal: **Enviar boletim** O sistema ira enviar o boletim a cada bimestre para e-mail cadastrado **Emitir Boletim** Será disponibilizada a opção de imprimir o boletim do aluno caso seja solicitado por algum interessado.
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

CASO DE USO: REGISTRARHORÁRIO
ID: UC07
Atores: Funcionário
Descrição: Este caso de uso tem como finalidade registrar os feriados e dias importantes do ano letivo
Precondições:O funcionário deve estar logado no sistema.
Fluxo principal: **Registrar as datas do calendário letivo** O funcionário ira registrar as datas do ano letivo como também feriados e dias importantes do ano letivo Depois de feito esses registros o funcionário deverá clicar no botão Registrar datas. Para a inclusão dos dados no calendário do sistema. **Alterar Dados do diário** O funcionário poderá alterar as datas do calendário do sistema quando necessário.
Pós-condições: O sistema será atualizado com as informações.
Fluxo alternativo: Nenhuma

CASO DE USO: Consultar/Imprimir desempenho dos professores
ID: UC08
Atores: Funcionário
Descrição: Este caso de uso tem como finalidade mostra um relatório da avaliação dos professores feita pelos alunos
Precondições: O funcionário deve estar logado no sistema. Ter concluído o período de avaliação dos professores e ter uma base de dados colhidas através do caso de uso UC07.
Fluxo principal: **Consultar dados da Avaliação** O sistema ira mostra um relatório com os resultados da avaliação do professor **Imprimir Dados** Depois poderá ser feita uma impressão dos resultados.
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

CASO DE USO: Consultar/Imprimir desempenho dos alunos
ID: UC09
Atores: Funcionário
Descrição: Este caso de uso tem como finalidade mostra um relatório da avaliação dos alunos ao final do período letivo
Precondições: O funcionário deve estar logado no sistema. Ter concluído o período letivo, os dados são coletados através do diário escolar, caso de uso UC05.
Fluxo principal: **Consultar dados da Avaliação** O sistema ira mostra um relatório com os resultados da avaliação dos alunos no ano letivo divido por ano/serie. **Imprimir Dados** Depois poderá ser feita uma impressão dos resultados.
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

CASO DE USO: Consultar Boletim dos alunos
ID: UC10
Atores: Aluno
Descrição: Este caso de uso tem como finalidade visualizar o boletim atualizado do aluno
Precondições: O aluno deve estar logado no sistema. O professor deve ter registrado as notas no diário escolar, caso de uso UC05.
Fluxo principal: **Consultar boletim de notas** Este caso de uso se inicia quando o aluno solicita a visualização do boletim de notas O sistema ira mostras as notas já registradas.
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

CASO DE USO: VerHorárioAluno
ID: UC11
Atores: Aluno
Descrição: Este caso de uso tem como finalidade mostra uma tela com o horário das aulas individual por aluno.
Precondições: O aluno deve estar logado no sistema.
Fluxo principal: **Consultar dados da Avaliação** O sistema ira mostra o horário do aluno. **Imprimir Dados** Depois poderá ser feita uma impressão dos resultados.
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

CASO DE USO: AVALIAR PROFESSOR
ID: UC12
Atores: Aluno
Descrição: Este caso de uso tem como finalidade registrar as respostas colhidas através de formulário onde o aluno faz uma avaliação do professor, esta avaliação e feita de acordo com a faixa etária e ano/serie que o professor leciona.
Precondições:O aluno deve logado no sistema e esta matriculado no ano/serie e o professor deve estar ministrando aula na referida série.
Fluxo principal: **Registrar os dados da avaliação** E disponibilizado ao aluno ao final de cada bimestre um formulário eletrônico com questões de marcar (Sim ou Não) adaptado para ano/serie que o aluno estuda e o professor leciona. Depois de feito esses registros o aluno deverá clicar no botão Registrar informações. **Alterar Dados do diário** Depois de registrado as informações não poderá ser modificadas.
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

CASO DE USO: Ver Horário Professor
ID: UC13
Atores: Professor
Descrição: Este caso de uso tem como finalidade mostra uma tela com o horário das aulas individual por professor.
Precondições: O professor deve estar logado no sistema.
Fluxo principal: **Consultar dados da Avaliação** O sistema ira mostra o horário do aluno. **Imprimir Dados** Depois poderá ser feita uma impressão dos resultados.
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

CASO DE USO: Emitir histórico escolar
ID: UC14
Atores: Funcionário
Descrição: Este caso de uso tem como finalidade a impressão do Histórico escolar do aluno
Precondições: O funcionário deve estar logado no sistema. O aluno deve ter um cadastro como usuário. As notas de todo o período que o aluno estudou devem estar cadastradas no sistema.
Fluxo principal: **Imprimir histórico** Imprimir todos os dados registrado do aluno
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

CASO DE USO: Emitir confirmação de matrícula
ID: UC15
Atores: Funcionário
Descrição: Este caso de uso tem como finalidade a impressão da confirmação de matricula
Precondições: O funcionário deve estar logado no sistema. O caso de uso matricular (UC04) ter sido realizado
Fluxo principal: **Imprimir** Após a confirmação da matricula o sistema ira emitir um comprovante.
Pós-condições: Nenhuma
Fluxo alternativo: Nenhuma

APÊNDICE D - Documento de Arquitetura de Software

Sistema de Gestão Escolar da Escolar (SGE Arco-Íris)

Documento de Arquitetura de Software

Versão 1.1

Histórico da Revisão

Data	Versão	Descrição	Autor
23/02/2015	1.0	Versão Inicial do Documento de Arquitetura	Damião Rodrigues
06/05/2015	1.1	Versão Final do Documento de Arquitetura – Atualização.	Damião Rodrigues, Rodrigo Alves Costa

1 INTRODUÇÃO

Neste documento iremos detalhar as principais partes da arquitetura proposta para o desenvolvimento do Sistema de Gestão Escolar. A arquitetura é formada pelo padrão de projetos de arquitetura: MVC e Facade, seguindo o paradigma de Orientação a Objetos. Neste documento será destacada cada parte da arquitetura o motivo da sua criação e qual a influência ela traz para a instituição.

1.1 Finalidade

1.2

Este documento oferece uma visão geral arquitetural abrangente do sistema, usando as visões arquiteturais para representar diferentes aspectos do sistema. Serve como um meio de comunicação entre o Arquiteto de Software e outros membros da equipe de projeto, para capturar e comunicar as decisões arquiteturais significativas que foram tomadas em relação ao Sistema.

1.2 Escopo

O documento propõe a apresentação da arquitetura de software do Sistema de Gestão Escolar da Escola Arco-Íris para toda a equipe de desenvolvimento.

2 REPRESENTAÇÃO DA ARQUITETURA

Este documento apresenta a arquitetura como uma série de visualizações; visualização caso de uso, visualização lógica e visualização da implementação.

Essas visões são apresentadas como RUP e utiliza a UML (Linguagem de Modelagem Unificada) subjacente desenvolvido utilizando a ferramenta de modelagem em UML, a StarUML.

A arquitetura de software é a definição e a representação de uma estrutura para a composição do sistema, em termos de seus componentes, propriedades e interações. A arquitetura deste sistema está baseada na arquitetura de referência J2EE. O modelo de arquitetura de software do Sistema de Gestão Escolar segue o padrão do MVC juntamente um Facade, pois o aumento da complexidade das aplicações desenvolvidas torna-se fundamental a separação destes dados.

3 OBJETIVOS E RESTRIÇÕES DA ARQUITETURA

A meta principal da arquitetura deste software é proporcionar entendimento comum entre os desenvolvedores. Para a proposta da arquitetura foram considerados fatores como: finalidade do sistema, tipo de usuários e ambiente de execução. Sendo assim, a arquitetura a ser utilizada neste projeto precisa atender às seguintes características:

- O sistema deve ser desenvolvido em camadas com um bom acoplamento entre elas.
- O modelo de interface deverá ser WEB de forma a suportar os seguintes navegadores: Mozilla Firefox, Google Chrome.
- A arquitetura deve seguir o padrão J2EE.
- Utilização de componentes *opensource*.
    - TomCat como servidor WEB (contêiner WEB)
- Persistência de tipagem (ex: formato de CPF, Data) devem ser feitos no cliente via JavaScript.
- Persistência de obrigatoriedade, como verificar se todos os campos obrigatórios de um formulário foram preenchidos pode ser feito no cliente via JavaScript.

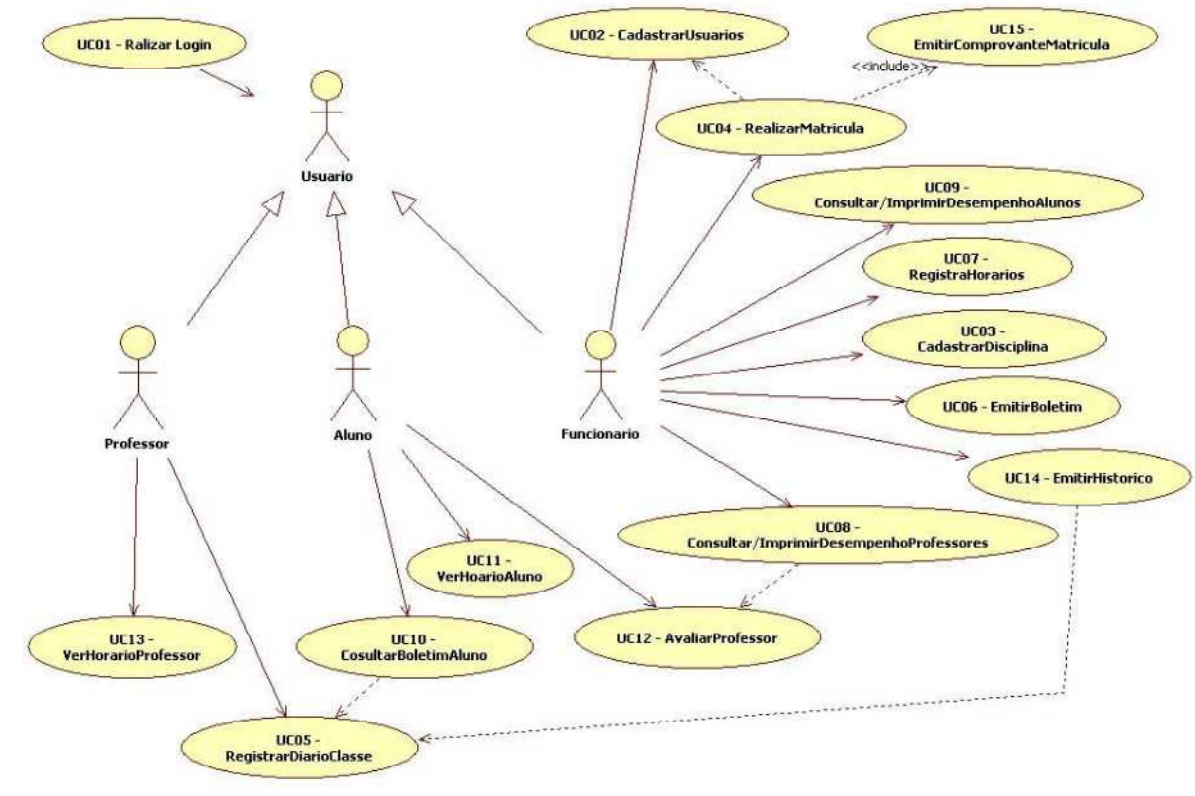
As restrições são:

- Para o desenvolvimento só serão utilizadas ferramentas open source.
- Os prazos de entrega devem ser respeitados.
- O sistema deve seguir as regras de negócio previamente analisadas e discutidas, sem alterar o funcionamento atual do negócio.

4 VISÃO DE CASOS DE USO

Uma descrição da visualização de casos de uso da arquitetura de software. A Visualização de Caso de Uso é uma entrada importante para a seleção do conjunto de cenários e/ou casos de uso que são o foco de uma iteração. Ela descreve o conjunto de cenários e/ou os casos de uso que representam alguma funcionalidade do sistema. Abaixo é apresentado os casos de uso que representam a funcionalidade do sistema.

ID	Nome
UC01	Realizar *login*
UC02	Cadastrar Usuários
UC03	Cadastrar Disciplinas
UC04	Realizar Matricular
UC05	Registrar diário de Classe
UC06	Emitir Boletim
UC07	Registrar Horários
UC08	Consultar/Imprimir desempenho dos professores
UC09	Consultar/Imprimir desempenho dos alunos
UC10	Consultar Boletim do aluno
UC11	Ver Horário Aluno
UC12	Avaliar Professor
UC13	Ver Horário Professor
UC14	Emitir Histórico
UC15	Emitir comprovante de matrícula



4.1 Atores

Aluno: Caracteriza-se como aluno o usuário que está ou já esteve matriculado em turmas, este terá acesso as suas respectivas funções

Professor: Caracteriza-se como professor o usuário ao qual esteja lecionando alguma disciplina - este terá acesso as suas respectivas funções.

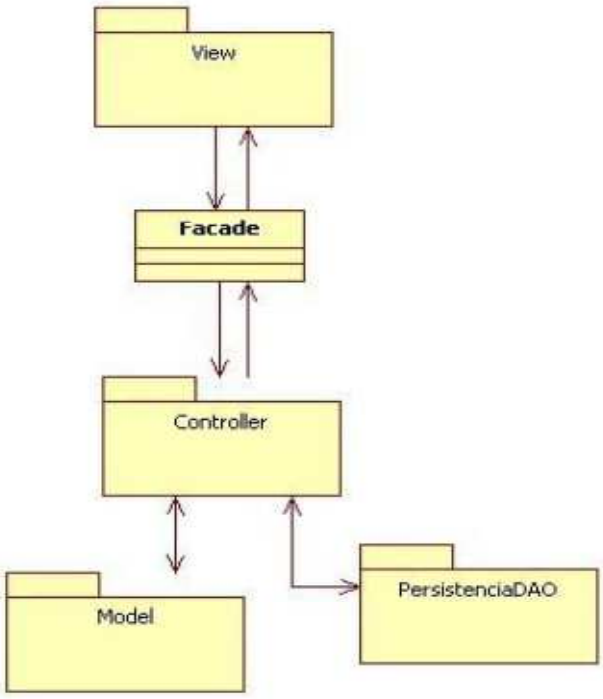
Funcionário: Caracteriza-se como funcionário o usuário responsável pelo cadastro em geral e expedição de relatórios tem acesso a suas respectivas funções.

5 VISÃO LÓGICA

Uma descrição da visualização lógica da arquitetura. Descreve as classes mais importantes, sua organização em pacotes de serviço e subsistemas e a organização desses subsistemas em camadas. Diagramas de classe podem ser incluídos para ilustrar os relacionamentos entre as classes, subsistemas, pacotes e camadas significativos da arquitetura.

5.1 Visão Geral

Na elaboração uma arquitetura para o desenvolvimento de sistemas, principalmente orientado a objetos, é necessário se preocupar com a separação real das camadas pertencentes à arquitetura. É nesse contexto que começamos a discutir os principais elementos dela, que é composta por elementos que valorizam o encapsulamento e a manutenção, utilizando de camadas e pacotes específicos.



View - Na camada view está presente a interface gráfica, que torna a apresentação das informações ao usuário coerentes com o objetivo do projeto.

Facade - Ela é a entrada única, tanto das interfaces gráficas do usuário como de outros sistemas, para o acesso as regras de negócio. Com a fachada, conseguimos então ter uma independência de interface gráfica, como também podemos proteger e controlar o acesso às regras de negócio.

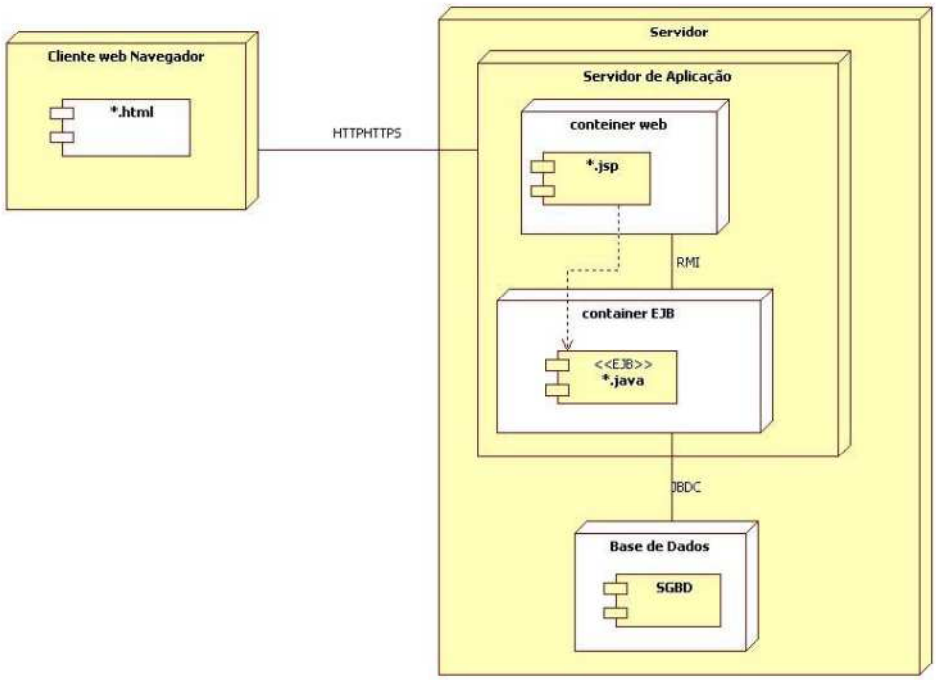
Controller - Um controlador é responsável pela execução de um ou mais fluxos de execução que são modeladas em um caso de uso, ou seja, podemos dizer que o controlador é em si a implementação da regra de negócio.

Model - O modelo extende a funcionalidade por trás da aplicação web. As classes que representam a Abstração da solução do problema encontram-se nesta camada. Os relacionamentos, cardinalidade, interfaces que especificam a solução são contidos neste pacote.

PersistenciaDAO - Para consultas ao banco de dados, existe uma camada especifica para trabalhar com o banco de dados, evitando alto acoplamento. A DAO contém classes abstratas que implementam funcionalidades de conexão e outros controles que guiam os tipos de acesso a banco de dados.

6 VISÃO DA IMPLANTAÇÃO

Uma descrição da visualização da implantação da arquitetura que descreve os diversos nós físicos para as mais típicas configurações de plataforma. Esta seção é organizada por nós e uma representação de seus componentes



Nome do Diagrama: Visualização da Implantação

Cliente web - computador com navegador

São os computadores da rede que fazem conexão com o servidor para realizar alguma tarefa

Servidor

Computador central, onde estão localizados o servidor de aplicações Tomcat e o de banco de dados SGBD - ele e responsável pelas respostas das solicitações dos clientes web.