UNIP INTERATIVA

Projeto Integrado Multidisciplinar – PIM VII

Cursos Superiores de Tecnologia

Sistema para realizar o controle de matrículas de cursos livres

UNIP Jundiaí

2018

UNIP INTERATIVA

Projeto Integrado Multidisciplinar – PIM VII

Cursos Superiores de Tecnologia

Sistema para realizar o controle de matrículas de cursos livres

Nome completo do aluno: Pedro Bueno da Silva

RA: 177789-1

Curso: Análise e Desenvolvimento de Sistemas

Semestre: 2º semestre 2018

UNIP Jundiaí

2018

Resumo**:**

Neste projeto é desenvolvido a fase de design para um sistema de controle de matriculas de cursos livres, seguindo as premissas que o projeto do sistema de controle de matricula já foi iniciado por um profissional que não o continua. Assim sendo a continuidade do projeto da fase de análise para a de design é de responsabilidade dos integrantes do projeto integrador multidisciplinar.

**Palavras chaves**: Sistema, controle, matricula, análise e design.

Abstract**:**

In this project, the design phase was developed for a free course registration control system, following the assumptions that the design of the registration control system has already been initiated by a professional who does not continue. Therefore, the continuity of the project from the analysis phase to the design phase is the responsibility of the members of the multidisciplinary integrating project.

**Keywords**: System, control, registration, analysis and design.

Sumário

[Resumo 3](#_Toc526676216)

[Abstract 4](#_Toc526676217)

[Introdução 6](#_Toc526676218)

[Fase de análise 8](#_Toc526676219)

[Descrição dos requisitos 9](#_Toc526676220)

[Diagrama de casos de uso 10](#_Toc526676221)

[Diagrama de classes 11](#_Toc526676222)

[Fase de design 12](#_Toc526676223)

[Arquitetura MVC 13](#_Toc526676224)

[Diagrama de classe de implementação 14](#_Toc526676225)

[Diagrama de sequência de implementação 14](#_Toc526676226)

[Diagrama de atividades 14](#_Toc526676227)

[Diagrama de distribuição 14](#_Toc526676228)

[Diagrama de classe de implementação – Manter Curso 15](#_Toc526676229)

[Diagrama de classe de implementação – Manter Aluno 16](#_Toc526676230)

[Diagrama de classe de implementação – Efetuar matricula 17](#_Toc526676231)

[Diagrama de classe de implementação – Relatório matricula 18](#_Toc526676232)

[Diagrama de classe de implementação – Login Aluno 19](#_Toc526676233)

[Diagrama de classe de implementação – Consultar Curso 20](#_Toc526676234)

[Diagrama de classe de implementação – Consultar Matrícula 21](#_Toc526676235)

[Diagrama de atividade - Valor Curso 22](#_Toc526676236)

[Diagrama de distribuição 23](#_Toc526676237)

[Conclusão: 24](#_Toc526676238)

[Referências bibliográficas 25](#_Toc526676239)

Introdução**:**

Para compreendermos quais as finalidades da fase de design, é necessário fazermos uma comparação com os objetivos da fase de análise. A fase de análise enfatiza a compreensão dos requisitos em seu caráter detalhado. Ou seja, busca-se detalhar os conceitos e operações relacionadas ao sistema. Em outras palavras, na fase de análise o que se busca é saber QUAIS os processos, conceitos, etc, relacionados ao software em desenvolvimento. Na fase de design, iniciamos o desenvolvimento de uma solução lógica baseada no paradigma da orientação a objetos. Ou seja, não mais nos satisfaz saber quais os processos, conceitos, etc. … relacionados com o software, mas saber COMO os processos, conceitos, etc… são implementados. Assim, diversos objetivos são esperados na fase de design. Dentre eles, temos:

• A aquisição de uma compreensão profunda dos fatores relacionados a requisitos não funcionais e restrições relacionadas a diversas características do sistema, tais como: linguagens de programação, reutilização de componentes de software, sistemas operacionais, tecnologias de distribuição e concorrência, tecnologias de bancos de dados, tecnologias de interface com o usuário, tecnologias de gerenciamento de transações, etc…

• A criação dos elementos lógicos necessários para as atividades de implementação subsequentes, por meio da definição de subsistemas, interfaces e classes

• O planejamento do trabalho de implementação decomposto em pedaços que possam ser trabalhados por diferentes equipes de trabalho, possivelmente ao mesmo tempo

• Captura das principais interfaces entre subsistemas, úteis no projeto da arquitetura do software, principalmente para a sincronização entre diferentes equipes de trabalho

• Permitir a especificação de elementos lógicos a serem implementados por meio de uma notação uniforme

• Criar uma abstração objetiva da implementação do sistema, de tal forma que a implementação seja um mero refinamento do design, permitindo e.g. a geração automática de código.

Vemos, portanto, que a etapa de design corresponde a uma mudança de postura em relação à fase de análise. Enquanto na fase de análise a nossa preocupação era compreender bem o que o sistema deveria fazer, na fase de design já se começa um planejamento estratégico para a implementação do sistema que será desenvolvido. Analisando agora a etapa de design dentro do ciclo de vida de um software, ou seja, tendo em perspectiva a maturidade do desenvolvimento do mesmo em termos de iterações, veremos que o design é enfatizado nas iterações finais da fase de elaboração e o começo da fase de construção. Lembrando que as fases de especificação, análise, design, implementação e testes se repetem em cada iteração, com diferentes ênfases, veremos que somente a partir de um certo número de iterações a ênfase no design será maior. Entretanto, veremos que essa ênfase maior dura somente até o meio da fase de construção.

# Fase de análise

Fase de análise ou também chamada de análise dos requisitos, é a fase que especifica os requisitos aplicando as investigações realizadas pelo analista. Assim resultando no refinamento e detalhamento dos requisitos a serem atendidos, os artefatos gerados nesta fase são de suma importância para a continuação do projeto de software.

Dentre as premissas propostas para este projeto acadêmico a fase de análise já está terminada, contudo a apresentação dos artefatos já existentes, se veem necessários por questões de nexo ao projeto de software representado por este documento acadêmico.

# Descrição dos requisitos

Dentre os artefatos da fase de análise temos a descrição dos requisitos conforme citados abaixo conforme manual proposto para este projeto.

O sistema será utilizado por atendentes e alunos matriculados. Todo acesso ao sistema é feito em terminais na escola por meio de login e senha.

O atendente poderá realizar as seguintes ações:

* Manter cursos: o atendente cadastra os cursos que abrangem 2 áreas diferentes: informática e artes. Todos os tipos de cursos possuem código, nome, data de início, data de término, horário, número de vagas e valor.
* Para os cursos de informática há também o número do laboratório e o registro dos softwares que serão utilizados, e, para o curso de artes, a descrição do material e os nomes dos livros que serão utilizados.
* Manter alunos: o atendente cadastra os alunos, informando: nome, endereço, telefone, e-mail, RG, CPF, login e senha do aluno.
* Cadastrar matrícula: o atendente cadastra a matrícula do aluno em um ou mais cursos. É gerado um código de matrícula, a data da matrícula, o valor da matrícula, o status de pagamento e o status da matrícula. Após o cadastro da matrícula, os dados da matrícula (código matrícula) são enviados para o Sistema Financeiro.
* Gerar relatórios de matrículas: o atendente emite relatórios em tela da quantidade de matrículas por curso em um determinado período.

O aluno poderá realizar as seguintes ações:

* Consultar cursos: o aluno consulta informações dos cursos disponíveis.
* Consultar matrículas: o aluno consulta matrículas de cursos que já realizou ou está realizando.

Também temos a descrição de regras especificas que segue abaixo.

Regras:

1. Caso o aluno já tenha realizado outros cursos, terá desconto progressivo. Um curso, desconto de 5%; dois cursos, desconto de 10%; mais de dois cursos, desconto de 15%.
2. Toda parte de cobrança do curso é controlada pelo Sistema Financeiro, o qual somente recebe as informações do cadastro de matrícula.
3. Um aluno matriculado pode realizar vários cursos.
4. Devem ser exibidas mensagens de advertência para todas as ações malsucedidas.

# Diagrama de casos de uso

O diagrama de caso de uso auxilia a comunicação entre o analista e os envolvidos ao projeto de software, abaixo o diagrama de caso de uso proposto.

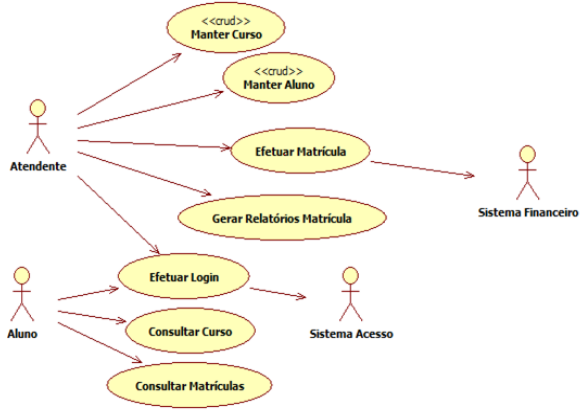


Figura - Diagrama de Casos de Uso (Manual PIM VII, 2018, p. 10)

# Diagrama de classes

Diagrama de classe representa o objeto e informação dos fundamentos usados pela aplicação**,** suas classes e relações podem ser implementadas para diferentes fins**.** Dentro da proposta deste projeto foi fornecido o diagrama a seguir.

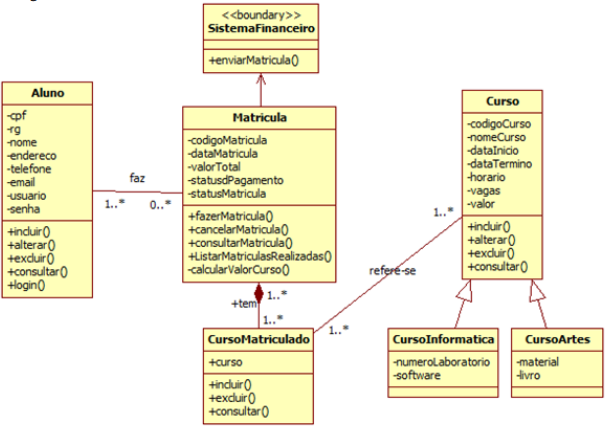
****

Figura - Diagrama de Classes (Manual PIM VII, 2018, p. 11)

# Fase de design

Fase de design ou também chamado de fase de projeto, nesta se considera como base as especificações de requisitos elaborados na fase de análise para se desenhar/projetar como serão realizadas as funcionalidades e interatividade do sistema.

Em design existem resoluções importantes que fogem do proposito das especificações de requisitos, exemplo, “o sistema deve realizar o cálculo de total de vendas e do custo variável”, ao desenhar ou projetar se decide se os cálculos serão realizados em duas funções independente ou por uma única função. Deve também se decide se a entrada de dados será realizada toda e apenas no final será calculado ou se os dados conforme os dados são imputados os cálculos são realizados.

# Arquitetura MVC

MVC é um padrão de arquitetura de software que separa a ação do usuário com o sistema da apresentação da informação, a sigla MVC representa model, view e controller em inglês, em português seria modelo, visão e controlador. O modelo contém logica, regras de negócios, dados da aplicação e funções. A visão pode conter qualquer espécie de saída de apresentação de dados, como um formulário ou uma lista, se necessário pode se ter várias visões de um mesmo dado. O controlador faz a intermediação da entrada e comutando em comandos para a visão ou modelo.

A seguir podemos visualizar o diagrama de classes para a arquitetura de referência de MVC usada para o sistema de matriculas. Para que o processo de implementação seja ágil a utilização de frameworks se vê como necessária, no caso podemos visualizar que a arquitetura utiliza o framework Laravel para auxiliar o uso do MVC, o ORM Eloquent para persistência de dados na base de dados, e Blade para facilitar a implementação das visões.

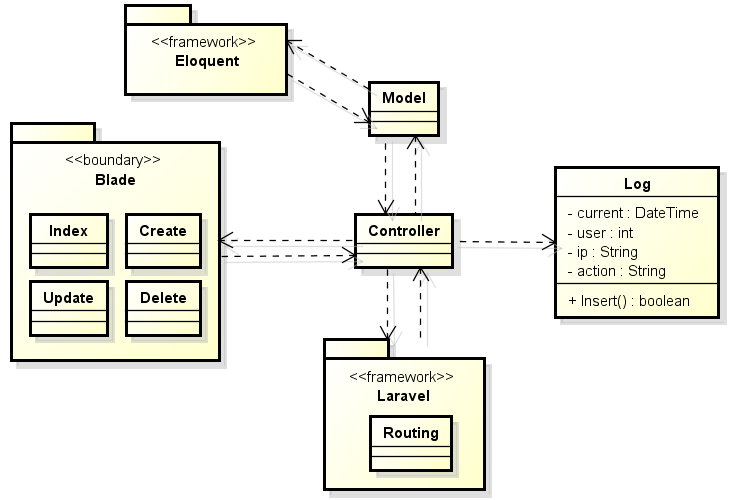


Figura - Arquitetura referência MVC

# Diagrama de classe de implementação

Dividido entre três perspectivas fundamentais no desenvolvimento de sistema, também expõe a relação entre os objetos. Seguindo os preceitos da arquitetura MVC se tem a forma de organizar a codificação, as classes são criadas e separadas pelas camadas do MVC.

# Diagrama de sequência de implementação

Tem como objetivo mostrar a troca de mensagens entre os objetos dentro no decorrer de tempo para realização de operações utilizando, no caso deste projeto de sistema é detalhada para cada caso de uso e segue o padrão da arquitetura MVC.

# Diagrama de atividades

O diagrama de atividades tem o propósito de salientar o fluxo de atividades em um único processo. O diagrama mostra como uma atividade depende de outra. As dependências dentre as atividades são representadas por transições que as conectam.

# Diagrama de distribuição

Diagrama de distribuição ou também chamado de diagrama de implantação, detalha os componentes de software e hardware e a interação entre eles. Os componentes são apresentados em “nós” de processamento, as dependências e comunicação entre os “nós” é a visão que o diagrama representa.

# Diagrama de classe de implementação – Manter Curso

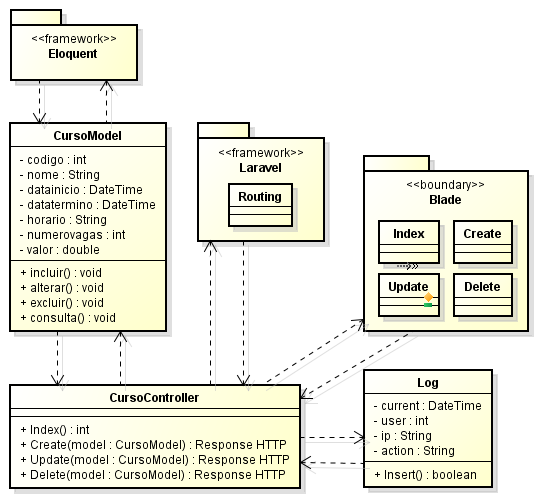


Figura - Diagrama de classe de implementação - Manter Curso

# Diagrama de classe de implementação – Manter Aluno

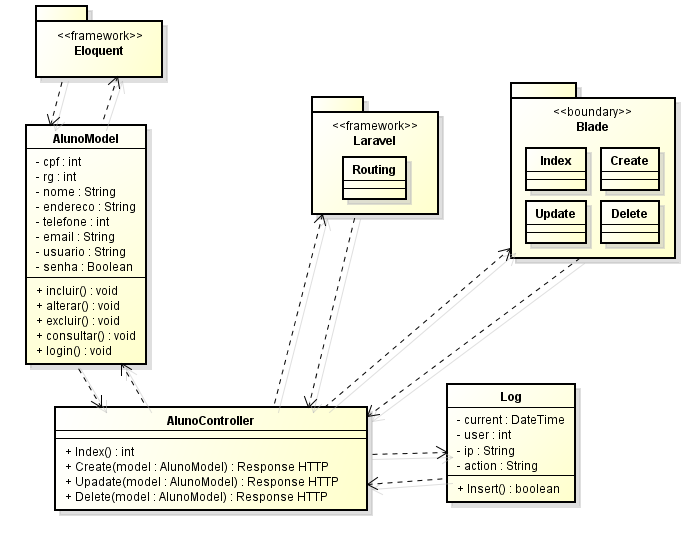


Figura - Diagrama de classe de implementação - Manter Aluno

# Diagrama de classe de implementação – Efetuar matricula

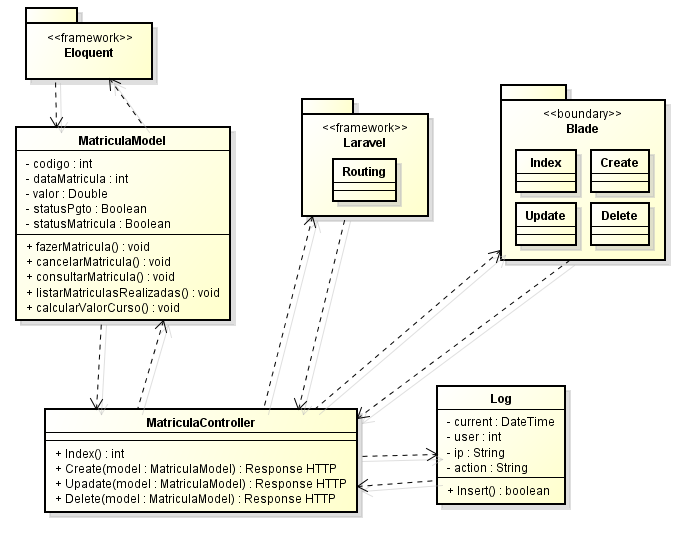


Figura - Diagrama de classe de implementação - Efetuar matricula

# Diagrama de classe de implementação – Relatório matricula

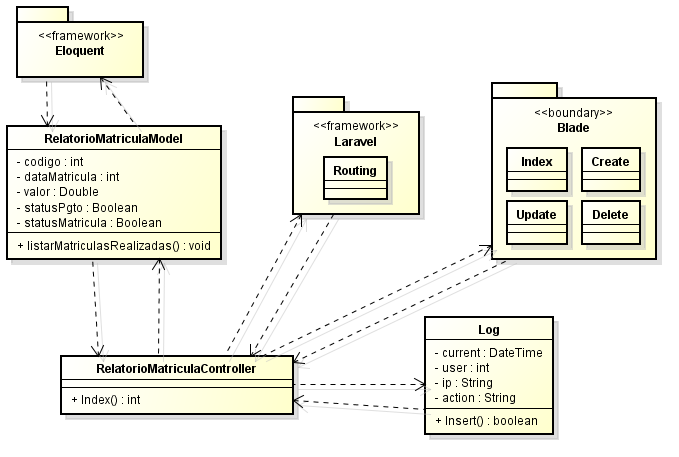


Figura - Diagrama de classe de implementação - Relatório matricula

# Diagrama de classe de implementação – Login Aluno

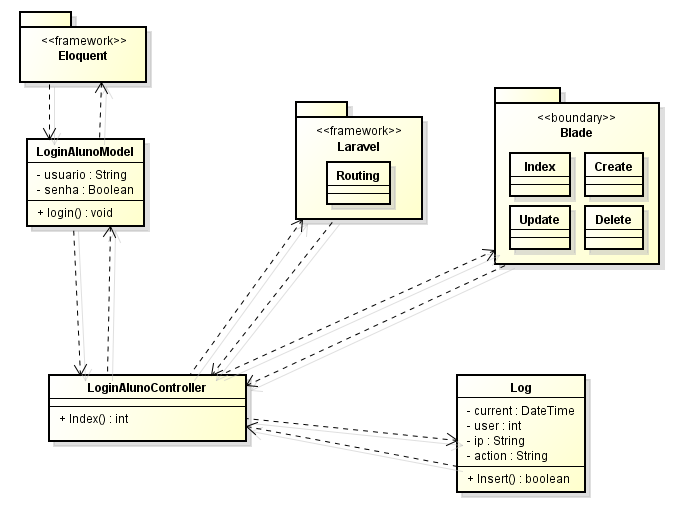


Figura - Diagrama de classe de implementação - Login Aluno

# Diagrama de classe de implementação – Consultar Curso

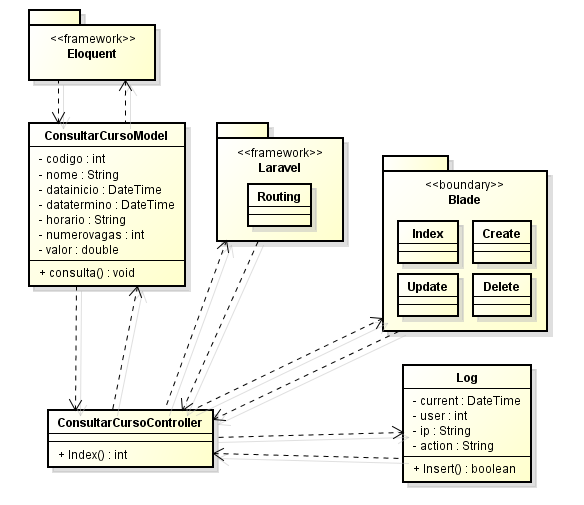


Figura - Diagrama de classe de implementação - Consultar Curso

# Diagrama de classe de implementação – Consultar Matrícula

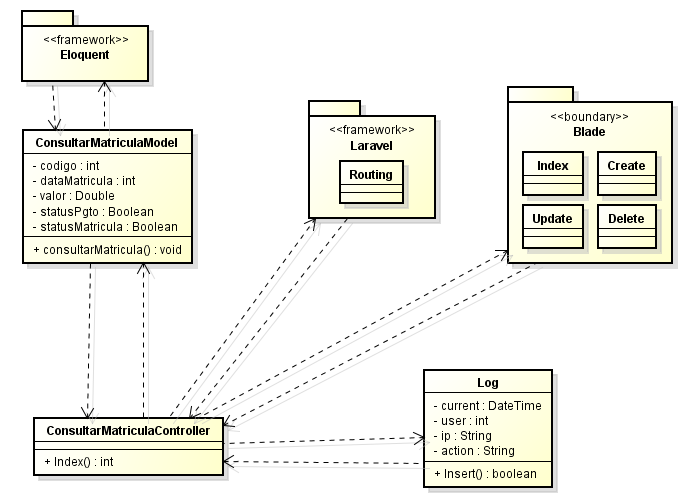


Figura - Diagrama de classe de implementação - Consultar Matricula

# Diagrama de atividade - Valor Curso

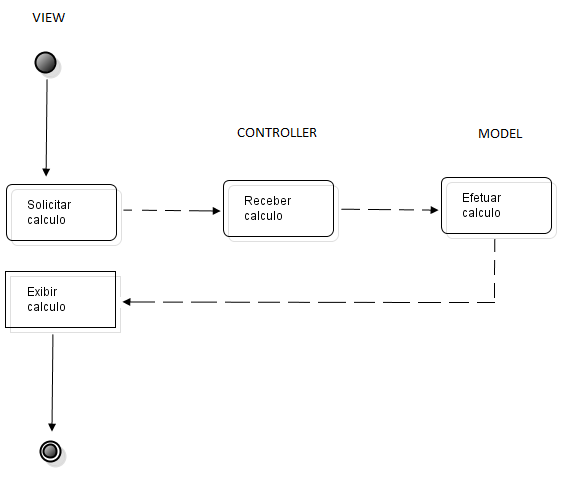


Figura - Digrama de atividade - Valor Curso

# Diagrama de distribuição

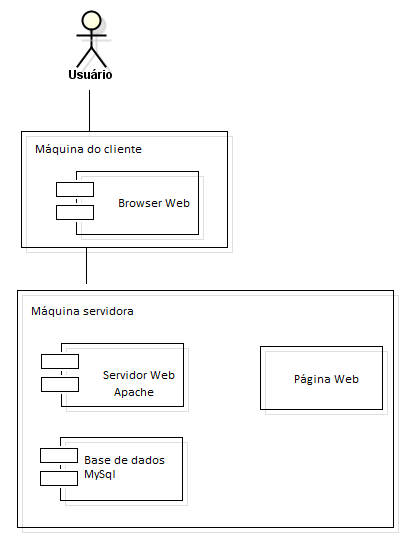


Figura - Digrama de distribuição

# Conclusão:

Com a execução desse projeto conseguimos entender a real importância dos diagramas e das técnicas que aprendemos nesse semestre. Outro ponto importante foi o planejamento para execução de um projeto, notamos que seguindo todos os passos temos um melhor resultado e diminuem os erros e dúvidas e assim podemos passar para ao usuário todos os aspectos do projeto e se caso ocorrer algum erro saberemos soluciona lós com maior facilidade.

Referências bibliográficas**:**

**UNIP INTERATIVA.** **Manual PIM VII** – Disponível em: https://ava.ead.unip.br/bbcswebdav/pid-360907-dt-content-rid-2419929\_1/institution/Conte%C3%BAdos%20das%20Disciplinas%20Espec%C3%ADficas/PIM%20-%20REGULAR/GEST%C3%83O%20ANALISE%20DE%20DESENVOLVIMENTO%20DE%20SISTEMAS/Projeto%20Integrado%20Multidisciplinar%20VII/Manual%20PIM%20VII.pdf. Acesso em: 04 de Setembro de 2018.

**A Fase de Análise no Processo Unificado** –Disponível em: http://www.dca.fee.unicamp.br/~gudwin/ftp/ea976/Analise.pdf. Acesso em: 07 de Setembro de 2018.

**MVC Xerox Parc** - Disponível em http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html. Acesso em 13 de setembro de 2018

PENDER, Tom. **UML, a Bíblia**, Tradução Daniel Vieira - Rio de Janeiro: Editora Campus, 2004.

FLOWER, Martin. **UML Essencial: Um breve guia para linguagem-padrão de modelagem de objetos**. São Paulo: Editora Bookman, 2007.

**Diagramas de classe UML: referência**: Disponível em https://msdn.microsoft.com/pt-br/library/dd409437.aspx. Acesso em 11 de Setembro de 2018