ES3 - Engenharia de Software III - Manhã

TEMA: METODOLOGIA CUSTOMIZADA

Grupo AViS

Pedro Bernardo de SOUSA 0030481711006

Weuller Júnior Souza Bessa 0030481621040

Vítor Andrade Marques da Silva 0030481511040

Sorocaba

Primeiro Semestre de 2019

**RESUMO**

A Alloy City Linguistics desenvolveu e mantém uma plataforma de ensino e aprendizado de francês baseada em tecnologias web. O objetivo do grupo AViS até o final do curso ADS dos participantes é desenvolver um cliente para essa plataforma existente. O cliente vai se chamar AViS - Alloy Virtual Space, e vai se apoiar na RESTful API disponibilizada pela Alloy para o projeto. O cliente AViS será capaz de simular um ambiente 3D onde professor e aluno poderão interagir de maneira mais próxima de interações humanas presenciais, isto é, com a impressão de compartilhar o lugar, e não apenas o momento.

**Palavras-chave:** AViS. *Plataforma*. EAD. Ambiente.Tri-dimensional.

**LISTA DE figuras**

Figura 1 Símbolos xx

Figura 2 Ciclo de Vida xx

Figura 3 Processos xx

**LISTA DE TABELAS**

Tabela 1 Padrão xx

Tabela 2 Ciclo xx

Tabela 3 Grupos xx

**SUMÁRIO**

**1 INTRODUÇÃO** 5

1.1 QUESTÃO DE PESQUISA xx

1.2 OBJETIVOS DA PESQUISA xx

**2 FUNDAMENTAÇÃO TEÓRICA** xx

**2.1** PROCESSOS ORGANIZACIONAIS................................................................ xx

**2.2** PROCESSO UNIFICADO RATIONAL (RUP) xx

RUP - Rational Unified Process - é um framework iterativo para desenvolvimento de software criado pela Rational Software, que é uma divisão da IBM. Não se trata de um método prescritivo concreto, mas um conjunto adaptável de conceitos, que podem ser seletivamente adotados por uma equipe de desenvolvimento, conforme suas necessidades. RUP é uma implementação de marca registrada do Processo Unificado proposto, que é mais genérico. UP, *Unified Process*, pode, ocasionalmente, ser usado no lugar de RUP para de evitar a infração de direitos autorais da IBM.

Ao longo dos anos 90, a Rational Software incorporou diversos elementos ao RUP, a medida que foi adquirindo outras empresas. Entre 2000 e 2003, mudanças foram introduzidas para permitir a adaptação do método com base em necessidades práticas, encontradas pelas equipes durante o uso da metodologia. Por exemplo, foram introduzidos conceitos e técnicas de metodologias como XP (Extreme Programming), entre outras, que coletivamente seriam chamadas de metodologias Ágil. Entre as novas ideias que foram acrescentadas ao RUP, figura o chamado TDD, ou Test Driven Development, que dá mais destaque ao teste, não apenas para verificar a implementação correta de uma unidade de código, mas também para ajudar a equipe a definir com precisão determinística o que se espera de cada unidade.

**2.3** PROCESSO UNIFICADO ÁGIL (AUP) xx

AUP – Agile Unified Process – é uma versão simplificada do RUP desenvolvida por Scott Ambler, considerando técnicas e conceitos ágeis utilizando técnicas de desenvolvimento orientado a testes (TDD) onde o ciclo de desenvolvimento é mais curto e os requisitos são transformados em casos de teste, sendo assim, em vez de entregar o software de uma vez, são liberadas pequenas partes, o AUP utiliza também de modelagem ágil (AM) já que o ciclo é mais curto, a modelagem ágil permite um ambiente com possíveis rápidas mudanças. No AUP as etapas de (implementação, teste e implantação) ocorrem partindo da segunda etapa, já na AUP estas são as três últimas etapas, sendo assim, RUP é uma metodologia mais estática (sequêncial), já que a documentação precede o sistema, não havendo muitas alterações nas etapas finais.

* 1. DIAGRAMAS UML xx

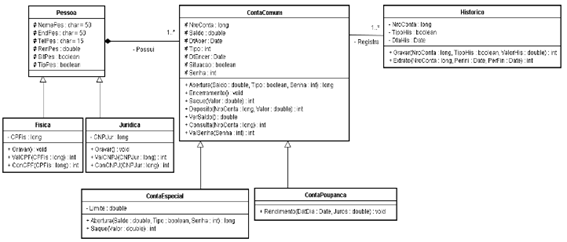
A sigla UML significa Linguagem de Modelagem Unificada utilizada para representar um sistema de forma padronizada. Não sendo uma metodologia, a UML tem como objetivos a especificação, documentação e estruturação por meio de diagramas representando o sistema como um todo.

Estes diagramas são classificados como, estruturais e comportamentais

# Diagramas estruturais

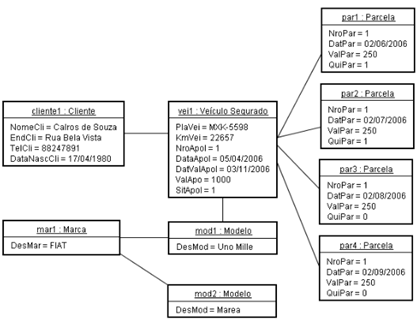
Tratam do ponto de vista do sistema quanto às classes, representando as estruturas estáveis

* **Diagramas de classe**



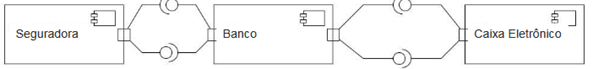
O Diagrama de Classes é utilizado para fazer a representação de estruturas de classes de negócio, interfaces e outros sistemas e classes de controle.

* **Diagramas de objetos**



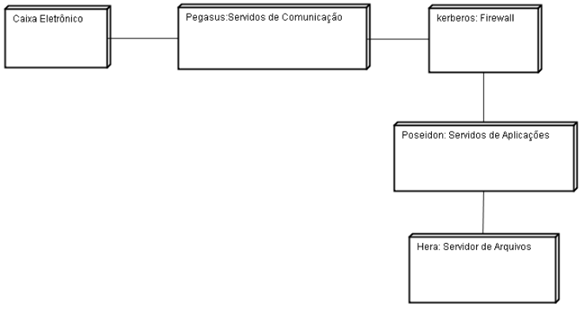
O diagrama de objetos representa os objetos de um diagrama de classes em um determinado instante de tempo, representando suas instâncias e seus relacionamentos, conforme definidos no diagrama de classes.

* **Diagramas de componentes**



Este diagrama mostra os artefatos de que os componentes são feitos, como arquivos de código fonte, bibliotecas de programação ou tabelas de bancos de dados.

* **Diagramas de implementação ou instalação**



O diagrama de implantação, consiste na organização do conjunto de elementos de um sistema para a sua execução.

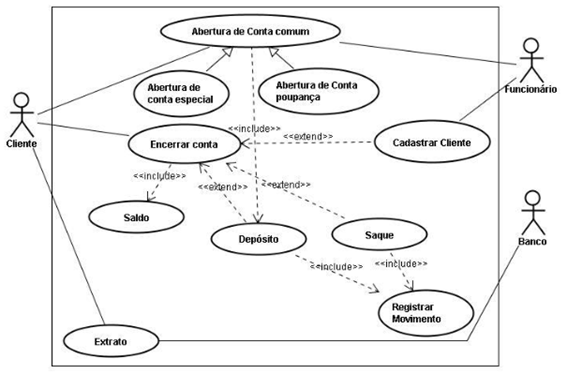
* **Diagramas de pacotes**

O diagrama de pacotes tem o objetivo de transformar as classes em pacotes. O critério para definir os pacotes é subjetivo e depende da visão e das necessidades do projetista. Este deve definir uma certa semântica e colocar os elementos similares e que tendem a serem modificados em conjunto num mesmo pacote. Como também, pode-se usar os pacotes para mostrar a arquitetura do sistema.

# Diagramas comportamentais

Utilizado para visualizar, especificar, construir e documentar aspectos dinâmicos, ou seja, que se altera, como por exemplo, itens como o fluxo de mensagens.

* **Diagrama de caso de uso**



Um diagrama de caso de uso mostra um conjunto de casos de uso e atores (um tipo especial de classe) e seus relacionamentos.

* **Diagramas de sequência**

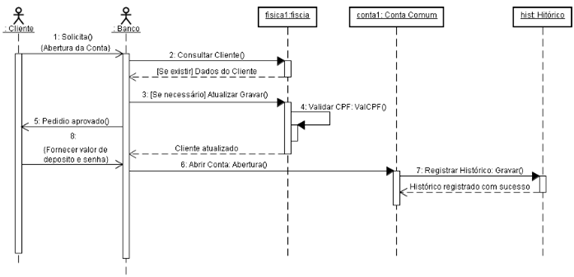
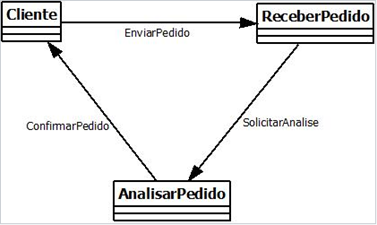


Diagrama de interação que dá ênfase à ordenação temporal de mensagens.

* **Diagramas de colaboração**



Um diagrama de comunicação é um diagrama de interação que dá ênfase a organização estrutural dos objetos que enviam e recebem mensagens.

* **Diagramas de transição de estados**

Mostra uma máquina de estados, que consiste de estados, transições, eventos e atividades. Use o diagrama de estados para ilustrar a visão dinâmica de um sistema. Esses diagramas são importantes principalmente para fazer a modelagem do comportamento de uma interface, classe ou colaboração.

Referências:

<https://homepages.dcc.ufmg.br/~figueiredo/disciplinas/aulas/uml-diagramas_v01-1.pdf>

https://pt.wikipedia.org/wiki/UML

**3 ANÁLISE DE NEGÓCIOS (INÍCIO)** ................................................................ xx

**3.1** SITUAÇÃO ATUAL .......................................................................................... xx

**3.2** PROBLEMAS ................................................................................................... xx

**3.3** SOLUÇÕES ..................................................................................................... xx

* 1. SITUAÇÃO PROPOSTA ...................................................................................xx

**4 SISTEMA DE INFORMAÇÕES (ELABORAÇÃO)** ........................................... xx

**4.1** CONTEXTO ..................................................................................................... xx

**4.2** REQUISITOS NÃO FUNCIONAIS ................................................................... xx

**4.3** REQUISITOS FUNCIONAIS xx

**4.4** DIAGRAMA DE CASOS DE USO xx

**4.5** CASOS DE USO DETALHADOS xx

**4.6** OUTROS DIAGRAMAS xx

**5 CONSIDERAÇÕES FINAIS**

**REFERÊNCIAS** XX

**1 INTRODUÇÃO**

Arial 12 com n parágrafos.

* 1. QUESTÃO DE PESQUISA**:**

*Definir e especificar uma metodologia customizada para projetos de software.*

Aqui, pretende-se definir a metodologia customizada que se pretende usar para desenvolver o cliente AViS.

A metodologia base será a AUP. Com isso, pretende-se manter e aplicar a recomendação de basear o desenvolvimento em ciclos curtos, com resultados concretos e tangíveis. Pretende-se também manter em mente os **12 Princípios do Manifesto Ágil**.

## Nossa maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado.

## Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente.

## Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo.

## Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.

## Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho.

## O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face.

## Software funcionando é a medida primária de progresso.

## Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.

## Contínua atenção à excelência técnica e bom design aumenta a agilidade.

## Simplicidade – a arte de maximizar a quantidade de trabalho não realizado – é essencial.

## As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizáveis.

## Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

Tendo esses 12 princípios em vista e utilizando como base a metodologia AUP, nós decidimos que o nosso ciclo de desenvolvimento terá 2 semanas.

Nós pretendemos também incorporar na nossa metodologia alguns aspectos da TDD, Test Driven Development. Em TDD, o teste é frequentemente escrito antes daquilo que ele testa. Ao escrever o teste, a equipe frequentemente atravessa perguntas sobre o comportamento da unidade. É útil que essas perguntas sejam respondidas antes de implementar a unidade. O teste é equivalente a redundância desejada, por exemplo, em engenharia mecânica: se uma bomba de fluido do sistema hidráulico é muito importante e a segurança do operador seria ameaçada em caso de falha daquela peça, acrescenta-se uma segunda bomba. Em engenharia de software, esse tipo de redundância (duplicar o código) é absurda, visto que bugs no código de origem estarão presentes na cópia. Um teste, por sua vez, contempla apenas a interface da unidade lógica (função, classe, etc.), mas mantém total independência dos algoritmos usados dentro dela. Assim, avaliamos automaticamente o comportamento da unidade.

Assim, vamos progressivamente expandir o projeto, em ciclos de 2 semanas, sempre dando ênfase ao 3 pilares do projeto: a modelagem, o teste e o código em si.

Isso significa que, ao final de cada ciclo, nós pretendemos poder apresentar novos diagramas, testes e unidades lógicas, assim como refinamentos nos diagramas, testes e unidades lógicas existentes.

O cronograma deverá ser elaborado contemplando as 4 fases AUP (concepção, elaboração, construção e transição) 3 vezes. A primeira iteração do conjunto de fases será realizada ao longo do semestre corrente, e deverá ser concluída em meados de junho de 2019. A segunda iteração das 4 fases AUP dirá respeito à disciplina de Laboratório de Engenharia de Software, e deverá ser concluída em meados de dezembro de 2019. Naquele período do projeto, uma ênfase especial será dada à fase de construção, dada a natureza da disciplina. Finalmente, ao longo do último semestre do curso, nós entraremos em uma período final do projeto, ao final do qual estima-se que haverá um produto devidamente apresentável ao mercado, que poderá ser colocado à disposição dos usuários.

Neste primeiro período, que já começou, nós pretendemos enfatizar a arquitetura do sistema, a integração com o backend existente, através de pequenas unidades de código capazes de demonstrar o uso das tecnologias que deverão ser usadas.

Em particular, deverá ser implementado até junho de 2019:

## Um programa capaz de trocar mensagens remotamente, através do protocolo UDP, usando endereços IPv6;

## Um programa capaz de renderizar um ambiente em 3D;

## Um processo capaz de gerar código a partir de diagramas UML e vice-versa;

## Um programa capaz de efetuar o login na RESTful API da Alloy;

## Um programa capaz de baixar conteúdo disponibilizado pela RESTful API da Alloy;

## Um programa capaz de ingerir o fluxo de áudio de um microfone;

## Um programa capaz de ingerir o fluxo de vídeo de uma webcam;

## Um programa capaz de identificar as coordenadas de um rosto em um fluxo de imagens;

## Um programa capaz de processar um fluxo de imagens (recortar, transformar em monocromático, etc);

Obviamente, esses programas deverão ser acompanhados de uma breve documentação que registra o porque de cada demonstrativo, no contexto do cliente AViS. Isto é, deverá ser devidamente explorado, conceitualmente, porque o cliente AViS só será possível se essas tecnologias forem demonstradas.

**1.2** OBJETIVOS DA PESQUISA:

**1.2.1** **Objetivo Geral**

Conhecer e aplicar padrões ao processo de software. Mapear modelos de representação. Integrar abordagens organizacionais e tecnológicas.

**1.2.2 Objetivos Específicos**

Abordagem de Requisitos Organizacionais com a opção de Análise de Processos Gerenciais (Gestão por Processos e Gestão de Processos).

Considerar opções para projetos de software no contexto dos processos, que podem ser: Software Autônomo, Orientado por Processo e Automatizado por Processo.

No projeto do Software, utilizar Diagramas UML aprendidos em ES1 e ES2, completando e ampliando essas representações e aplicando o Processo Unificado (RUP) com abordagem Ágil (Agile Unified Process).

1. **FUNDAMENTAÇÃO TEÓRICA**

**É o que já existe, deveremos estudar e descrever os modelos atuais, antes de apresentar a nossa solução (quanto ao processo de desenvolvimento).**

**2.1** PROCESSOS ORGANIZACIONAIS

**2.2** PROCESSO UNIFICADO RATIONAL (RUP)

**2.3** PROCESSO UNIFICADO ÁGIL (AUP)

**2.4** DIAGRAMAS UML

1. **ANÁLISE DE NEGÓCIOS (INÍCIO)**

**3.1** SITUAÇÃO ATUAL

**3.2** PROBLEMAS

**3.3** SOLUÇÕES

**3.4** SITUAÇÃO PROPOSTA

1. **SISTEMA DE INFORMAÇÕES (ELABORAÇÃO)**

**4.1** CONTEXTO

**4.2** REQUISITOS NÃO FUNCIONAIS

**4.3** REQUISITOS FUNCIONAIS

**4.4** DIAGRAMA DE CASO DE USO

**4.5** CASOS DE USO DETALHADOS

**4.6** OUTROS DIAGRAMAS

1. **CONSIDERAÇÕES FINAIS**

**REFERÊNCIAS**

BIZAGI. ***Bizagi Modeler***. Disponível em:

<<http://www.bizagi.com/pt/produtos/bpm-suite/modeler>>

Acesso em: 16 de julho de 2018

NONAKA, Ikujiro e TAKEUCHI, Hirotaka.

**Criação do Conhecimento na Empresa:**

como as empresas geram a dinâmica da inovação.

Rio de Janeiro: Campus,1997.

OMG. **“Business Process Model and Notation:** BPMN.”

Versão 2.0 2011

PMI. **“Guia do Conhecimento em Gerenciamento de Projetos: Guia PMBOK”.**

5 ed.

São Paulo: Saraiva, 2013.

PROBST, G.; RAUB, S.; ROMHARDT, K.

“**Gestão do Conhecimento** - Os Elementos Construtivos do Sucesso”

Porto Alegre: Bookman, 2002

WANKES, Leandro**. Ciclo de Vida de um Projeto**. Disponível em:

<<http://www.wankesleandro.com>>

Acesso em: 16 de julho de 2018

*Tema:*

O projeto de minha equipe visa aprimorar a experiência do aluno de educação à distância, através de uma ferramenta que maximize a imersão num ambiente de aula. Estimulando sensorialmente a percepção de sutis aspectos da comunicação que, de fato, são significativos no caso do ensino de idiomas, em especial na educação à distância. A plataforma oferecerá a possibilidade de o aluno ouvir as falas de outros agentes no processo educacional, buscando fazer com que a percepção se dê como se todos estivessem num mesmo ambiente, com as variações audíveis e visíveis em relação ao espaço e posicionamento dos agentes envolvidos.

A plataforma simulará uma sala de aula no ambiente virtual, com características similares às existentes em ambientes de jogos, de forma que o deslocamento dos avatares influencie na intensidade do som recebido pelos usuários, os próprios avatares deverão ser capazes de demonstrar expressões faciais da forma mais fidedigna possível.

Para tanto, serão utilizadas câmeras e microfones pelos usuários, para que possam ser captadas suas expressões faciais e sua capacidade de pronúncia e é aí que está o diferencial do projeto, o foco do projeto está ligado em como se dará a comunicação entre os agentes, abordando questões ligadas à compressão e à um ambiente em três dimensões.

Motivação:

A equipe está motivada com a ideia de projeto apresentada pelo Pedro Bernardo, tal iniciativa se deu por uma necessidade constatada em seu ambiente de trabalho. Acreditamos que essa plataforma, além de viável, pode se tornar um produto diferenciado em relação aos modelos existentes no mercado como o AVA (usado na Uninter) ou o BlackBoard (usado na Universidade Cruzeiro do Sul e na Unicid). Tal produto além de vendável, pode oferecer uma experiência mais intensa, verdadeiramente imersiva sob a ótica dos alunos usuários e se tornar uma ferramenta mais abrangente no processo de avaliação dos professores tutores.

Aptidões:

O escopo do projeto é muito abrangente, ao longo do desenvolvimento deveremos estipular diversas etapas de conquistas e nos tornaremos analistas capazes de mensurar os desafios à serem enfrentados.

Dada a abrangência, cada membro buscará oferecer suporte aos demais, porém, cada qual buscará agir conforme seus diferenciais. Temos no Bernardo, um dos programadores da equipe, Weuller a capacidade e experiência no desenvolvimento de interfaces, eu, Vítor, buscarei atuar com a documentação de cada etapa do projeto.

<https://www.devmedia.com.br/modelos-de-processos-ageis-conceitos-e-principios/30059>

<https://en.wikipedia.org/wiki/Rational_Unified_Process>

<http://www.ambysoft.com/unifiedprocess/agileUP.html>

<https://www.infoescola.com/engenharia-de-software/rup/>