Proposta de Reestruturação de Trabalho de Conclusão de Curso

Análise e Desenvolvimento de Sistemas – Jogos Digitais

Prof. Fábio Lúcio Meira

Prof. Paulo Augusto Nardi

Prof. Alexandre Ponce de Oliveira

Fatec
Lins
Prof. Antonio Seabra

Sumário

Proposta de Reestruturação de Trabalho de Conclusão de Curso

Análise e Desenvolvimento de Sistemas – Jogos Digitais

Organizado por
Prof. Fábio Lúcio Meira
Prof. Paulo Augusto Nardi
Prof. Alexandre Ponce

Todo(a) aluno(a) dos cursos Superiores de Tecnologia em Análise e Desenvolvimento de Sistemas e Jogos Digitais, da Fatec Lins – Prof. Antônio Seabra, deve(m) elaborar um Trabalho de Conclusão de Curso (TCC), o qual deverá posteriormente ser aprovado em banca, para que possam colar o grau em seu respectivo curso.

A Fatec Lins – Professor Antônio Seabra receberá os créditos de Propriedade Intelectual decorrentes de projetos desenvolvidos na Instituição, em proporcionalidade definida em normativas da agência INOVA Paula Souza, ressalvando os direitos de Propriedade Industrial, conforme a legislação vigente sobre Patentes (Invenção ou Utilidade), Marcas, Programas de Computadores, Topografia de Circuitos Digitais e Direitos Autorais.

O TCC poderá ser realizado em dupla ou individualmente, seguindo uma das três modalidades a saber:

1. Relatório Técnico (para o desenvolvimento de sistemas ou jogos);
2. Análise Comparativa;
3. Provas de Conceito.

Relatório Técnico

O Relatório Técnico será utilizado para documentar TCCs onde o processo de desenvolvimento de software é o foco dos mesmos. Fica descaracterizado então, o formato de monografia em seu sentido convencional, com viés científico, e passa-se a caracterizar o TCC como um projeto técnico.

Análise Comparativa

Uma Análise Comparativa será utilizada para documentar TCCs onde o autor deseja estabelecer uma comparação técnica entre ferramentas, métodos ou técnicas. Por exemplo: "*Comparativo de performance entre os SGBD's Oracle e MySQL integrados à linguagem de programação Java utilizando diferente mecanismos de conexão*".

Provas de Conceito

Uma prova de conceito é um termo utilizado para especificar um modelo ou protótipo que possa provar o conceito (teórico) estabelecido por uma pesquisa técnica. Por exemplo: "*Aplicação de técnicas de mineração de dados em ambientes big data em nuvem*".

RELATÓRIO TÉCNICO

Análise e Desenvolvimento de Sistemas

O Relatório Técnico terá a seguinte estrutura:

1. Introdução
2. Análise de Negócios
3. Análise de Requisitos
4. Análise e Design
5. Manual do Usuário
6. Considerações Finais
7. Bibliografia

Anexos

8. Novas Tecnologias (*Revisão e Tutorial - quando se aplicar*)
9. Código Fonte Explicado (*quando se aplicar*)

RELATÓRIO TÉCNICO
1) Introdução
A introdução deve apresentar a **proposta** do projeto, os **objetivos**, a **motivação**, a **metodologia de trabalho** e os **resultados esperados.**
2) Análise de Negócios
O principal objetivo desse capítulo e efetuar a Análise de Negócios, a qual pode envolver:
2.1) Análise do Processo de Negócios
Elaboração detalhada do estudo do Processo de Negócios para o qual a solução é desenvolvida. Duas ferramentas devem ser utilizadas: 1º) Diagrama de Caso de Uso de Negócios, o qual permite ao desenvolvedor ter uma visão macro do Processo de Negócios, identificando todas as suas atividades, os papeis que as desempenha e o fluxo dos produtos de trabalho pelo processo de negócio; 2º) Diagrama BPMN (Business Process, Model and Notation), o qual permite ao desenvolvedor detalhar as tarefas de cada uma das atividades do Processos de Negócios, através de um diagrama de sintaxe simples mas semântica rica. Este item só é aplicado quando a solução é desenvolvida para um sistema *ad hoc*[1]. Caso a solução seja um *shelf system*[2], então este item pode ser descartado.
2.2) Identificação do Problema
Após o desenvolvimento do item 2.1, deverá ser desenvolvida uma análise de possíveis problemas existentes em atividades ou tarefas do Processo de Negócios analisado, ou a identificação de atividades e/ou tarefas que são passíveis de melhoria. Caso a solução seja um *shelf system*, deverá ser desenvolvida uma indicação de possíveis aplicações da solução em qualquer ambiente.
2.3) Proposta de Solução
Deverão ser apresentadas aqui as principais características, em alto nível, da solução proposta. Devem ser descritas as principais funcionalidades que atenderão às demandas definidas no item 2.2. Possíveis inovações da solução proposta podem ser destacadas neste item.

1 A tradução literal de *Ad Hoc* é “para este”. Em Engenharia de Software designa uma solução para atender uma demanda específica.
2 Um *Shelf System* é um sistema desenvolvido para ser lançado diretamente no mercado (Sistema de Prateleira).

2.4) Análise de Mercado

Neste item, o autor deverá elaborar uma análise de ferramentas similares, não descontinuadas, traçando comparativos com a solução proposta, destacando os pontos positivos da mesma em relação às concorrentes.

3) Análise de Requisitos

A Análise de Requisitos tem como principal objetivo identificar as características estruturais e comportamentais da solução proposta. Os requisitos funcionais devem obrigatoriamente ser declarados. Caso se aplique, deve-se especificar os requisitos não funcionais.

3.1) Diagrama de Casos de Uso

O Diagrama de Casos de Uso demonstra o conjunto de funcionalidades da solução e o conjunto de agentes externos (atores) que interagem diretamente com a solução proposta, além de definir o escopo do projeto.

3.2) Especificações de Casos de Uso

Uma especificação de Caso de Uso faz uma descrição textual, passo a passo, da comunicação direta entre o ator e o caso de uso. O Anexo A apresenta o *template* e um exemplo de uma especificação de caso de uso.

3.3) Especificação Complementar

A especificação complementar descreve os Requisitos Não Funcionais da solução proposta. Requisitos Não Funcionais descrevem características estruturais da solução proposta.

3.3.1) Requisitos de Usabilidade

Requisitos de Usabilidade são os requisitos inerentes aos conceitos de Interação Homem-Computador.

3.3.2) Requisitos de Confiabilidade

Requisitos de Confiabilidade são os requisitos inerentes à segurança da solução proposta. Deve-se especificar requisitos de controle de acesso à solução, backup e recuperação, e tolerância à falhas.

3.3.3) Requisitos de Performance

Qualquer requisito inerente à performance da solução deve ser especificado neste item. Podem ser tratados requisitos de performance em relação a redes de comunicação, sistemas gerenciadores de banco de dados, servidores de aplicação, etc.

3.3.4) Requisitos de Suportabilidade

Requisitos de Suportabilidade indicam restrições de escalabilidade, reusabilidade, manutenibilidade, portabilidade, etc.

4) Análise e Design

O principal objetivo da Análise e Design é demonstrar a arquitetura proposta para a solução em desenvolvimento. A arquitetura de uma solução demonstra a *estrutura* e o *comportamento* desta solução.
A arquitetura de uma solução Orientada a Objetos é descrita pela UML (Unified Modelling Language) e seus diversos diagramas. Além disso, a arquitetura também deve se preocupar com a proposta do projeto lógico do banco de dados da solução em desenvolvimento.

4.1) Diagrama Entidade Relacionamento

O diagrama Entidade Relacionamento demonstra a lógica da persistência dos dados em um banco de dados relacional. O uso desse diagrama se justifica pela alta taxa de aplicação de SGBDRs no mercado de informática. Caso a aplicação não utilize banco de dados relacional ou caso utilize um SGBDOO, o uso desse item não se faz necessário.
Caso a aplicação utilize um banco de dados NoSQL, por exemplo, representar somente a estrutura física do banco de dados ou representar logicamente através de algum diagrama que consiga demonstrar as regras lógicas do banco.

4.2) Diagrama de Classes

O Diagrama de Classes é utilizado em primeira instância para representar a lógica da persistência dos dados em tempo de execução. Caso o trabalho em questão utilize SGBDOO, este diagrama também representará a lógica da persistência em banco.

4.3) Diagrama MVC

Um Diagrama MVC (Model – View – Control), demonstra a realização da estrutura física de um componente da solução proposta. O modelo MVC deverá variar de acordo com a arquitetura utilizada (Swing, JSP, JSF, Android, etc.). O **Anexo B** apresenta o *template* e exemplos de diagramas MVC em diversas arquiteturas.

4.4) Diagrama de Atividades

Um Diagrama de Atividades demonstra a lógica do comportamento de uma funcionalidade, em alto nível. O Diagrama de Atividades não deve demonstrar regras de negócio (como um fluxograma, por exemplo), apenas o fluxo de execução da atividade. Um Diagrama de Atividades deve também demonstrar os possíveis Cenários de execução da funcionalidade, servindo como subsídio para as atividades de Desenvolvimento e Testes de Software. Funcionalidades CRUD simples não precisam ser demonstradas, a não ser que possua regras de negócio complexas. O **Anexo C** apresenta o *template* e um exemplo de um Diagrama de Atividades.

4.5) Diagrama de Sequência

Um Diagrama de Sequência demonstra a ordem cronológica de troca de mensagens entre objetos de um componente ou entre objetos da camada de visão de componentes distintos. Um Diagrama de Sequência também pode demonstrar a comunicação entre um objeto e um ator, desde que o ator represente um sistema externo à solução proposta. Funcionalidades CRUD simples não precisam ser demonstradas, a não ser que possua regras de negócio complexas. O **Anexo D** apresenta o *template* e um exemplo de um Diagrama de Sequência.

4.6) Diagrama de Estados

Um Diagrama de Estados demonstra todos os possíveis estados que um objeto de uma determinada classe pode assumir, bem como, as transições e as restrições de transição entre estes estados. O **Anexo E** apresenta o *template* e um exemplo de um Diagrama de Estados.

4.7) Diagrama de Componentes

Um Diagrama de Componentes demonstra a realização da arquitetura de uma solução em alto nível, apresentando os diversos componentes da solução e a interface entre os mesmos.
Caso a solução seja complexa e apenas uma parte da mesma seja implementada, apenas as interfaces dos componentes implementados deverão ser demonstradas. O **Anexo F** apresenta o *template* e um exemplo de um Diagrama de Componentes.

5) Manual do Usuário

Este capítulo deverá conter a descrição da solução desenvolvida pelo aluno no projeto.
Este capítulo deve ser ricamente ilustrado, apresentando os detalhes da solução apresentada e suas funcionalidades na íntegra.
Uma dica para apresentação da ferramenta é seguir a linha de raciocínio apresentada na Especificação de Caso de Uso respectiva à funcionalidade.

6) Considerações Finais

Trace aqui as considerações finais sobre o processo de produção do software escolhido.

- Exponha os pontos positivos do seu trabalho;
- Exponha as dificuldades encontradas;
- Considerações sobre o resultado final;
- Considerações sobre propostas de melhoria;
- Considerações sobre projetos futuros;
- Conclusão.

7)) Bibliografia
Devem ser listadas todas as obras (artigos, livros, etc...) que foram usados como referência na monografia. Toda obra listada deve obrigatoriamente ter sido referenciada pelo menos uma vez no texto.
8) Novas Tecnologias (Quando se aplicar)
Sempre que o trabalho apresentar o uso de alguma tecnologia ou conceito diferenciado, que não seja Estado na Arte no curso de Análise e Desenvolvimento de Sistemas, então é necessário elaborar uma breve revisão bibliográfica sobre esta tecnologia ou conceito.
9) Código Fonte Comentado
Quando a compreensão do código fonte gerado for essencial para a compreensão do trabalho, no seu todo, ou parte, então é necessário que o aluno apresente este código fonte e o explique detalhadamente.

Anexo A – Template e Exemplo de Especificação de Casos de Uso

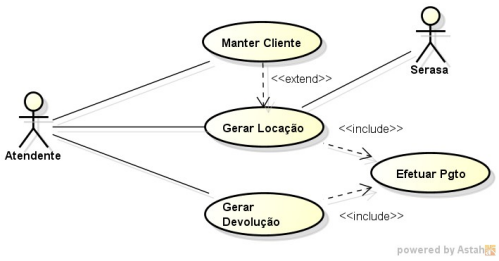
A.1) Template

A especificação de Casos de Uso deve, de forma clara, apresentar de forma textual, a sequência de operações de uma funcionalidade, bem como, as interações que ocorrem entre o ator e a mesma. Existem diversos modelos de Especificação de Caso de Uso, sendo que cada um se adéqua a uma necessidade específica. O modelo aqui apresentado é customizado a partir do modelo padrão proposto pela IBM, buscando sintetizar ao máximo seu conteúdo, sem perder a essência do mesmo.

Caso de Uso: *nome do caso de uso*
Fluxo Básico
O Fluxo Básico descreve o uso ótimo da funcionalidade, ou seja, o uso de forma que não ocorra exceções durante o fluxo de execução. Cada passo do Fluxo Básico deve ser devidamente identificado para que possa ser devidamente referenciado em outras sessões da Especificação de Caso de Uso. É importante que todo fluxo básico possua um passo específico para encerrar o Caso de Uso, garantindo que os tratamentos de exceções tenham um ponto de fuga.
Fluxos Alternativos
Um Fluxo Alternativo efetua o tratamento de uma exceção em um passo específico no caso de uso, independente de sua localização (Fluxo Básico, Fluxos Alternativos, Pontos de Inclusão/Extensão). Toda especificação de fluxo alternativo deve conter os seguintes elementos: a) **Origem**: ponto onde ocorre a exceção; b) **Exceção**: descrição da exceção ocorrida; c) **Tratamento**: qual será o tratamento que a solução oferecerá para a exceção; d) **Destino**: ponto para qual o fluxo de execução deverá retornar.
Pontos de Inclusão/Extensão
Um Ponto de Inclusão representa um conjunto de tarefas que são comuns a mais de um caso de uso. Neste caso, estas tarefas comuns podem ser desacopladas de seus casos de uso de origem e acopladas a um único ponto de inclusão. O caso de uso que teve as tarefas desacopladas acessa o ponto de inclusão para tal. Um Ponto de Extensão representa uma funcionalidade que deve ser requisitada durante a execução do caso de uso, sendo prevista esta ação. Neste caso, como a funcionalidade requisitada também representa um caso de uso, a mesma é requisitada através de um ponto de extensão e não através de um Fluxo Alternativo. A estrutura de um Ponto de Inclusão/Extensão deve conter os seguintes elementos: a) **Origem**: ponto onde ocorre a chamada ao Ponto de Inclusão/Extensão; b) **Fato**: descrição da exceção ou evento que gera a chamada ao Ponto de Inclusão/Extensão; c) **Chamada**: chamada ao Ponto de Inclusão/Extensão; d) **Destino**: ponto para qual o fluxo de execução deverá retornar.

A.2) Exemplo de Especificação de Caso de Uso

Considerando o seguinte diagrama, será especificado o caso de uso GERAR LOCAÇÃO:

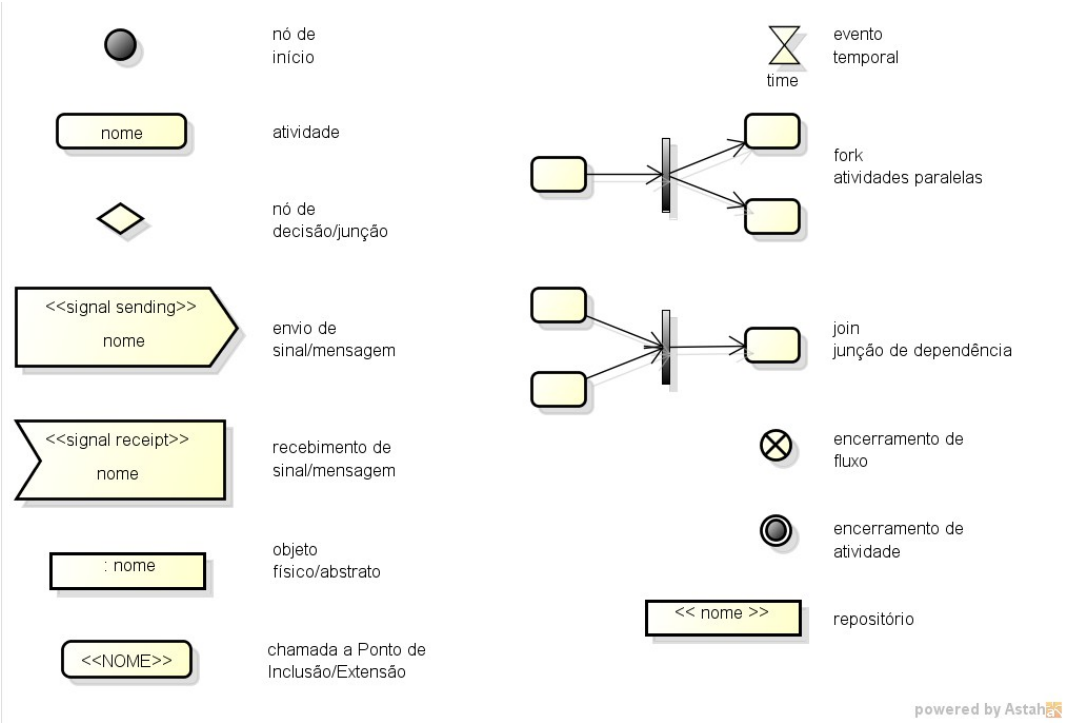


Observação: textos em itálico/azul são comentários, não devendo constar na especificação.

Caso de Uso: *Gerar Locação*
Fluxo Básico
fb1: O *Atendente* inicia o caso de uso ***O papel deste passo é identificar quem dispara o caso de uso*** fb2: O *Atendente* seleciona o cliente desejado fb3: O sistema verifica o status do cliente selecionado junto ao *Serasa* fb4: O *Atendente* seleciona o veículo desejado pelo cliente fb5: O Atendente gera o pagamento da locação fb6: O caso de uso é encerrado ***Todo fluxo básico deve ter um ponto de saída***
Fluxos Alternativos
fa1: Cliente Negativado – No fluxo básico **fb3** ***origem***, caso o cliente selecionado esteja negativado pelo Serasa ***exceção***, o sistema notificará o *Atendente* e cancelará a locação ***tratamento***. Retornar ao fluxo **fb6** ***destino***. fa2: Veículo Não Encontrado – No fluxo básico **fb4**, caso o atendente não encontre o veículo desejado pelo cliente, então deverá ser oferecida outra opção de veículo ao mesmo. Neste caso, é possível: fa2.1: se o cliente desejar escolher outro veículo, retornar ao fluxo **fb4**; fa2.2: se o cliente recusar a escolha de outro veículo, cancelar a locação e retornar ao fluxo **fb6**. fa3: Pagamento Recusado – No ponto de inclusão **pi1**, caso o pagamento do cliente tenha sido recusado, então o *Atendente* deverá notificar o cliente e cancelar a locação. Retornar ao fluxo **fb6**.
Pontos de Inclusão/Extensão
pe1: Cliente Não Cadastrado – no passo **fb2**, caso o cliente não seja cadastrado, efetuar a chamada ao ponto de extensão MANTER CLIENTE. Retornar ao fluxo **fb3**. pi1: Efetuar Pagamento – no passo **fb5**, para gerar o pagamento, efetuar a chamada ao ponto de inclusão EFETUAR PAGAMENTO. Retornar ao fluxo **fb6**.

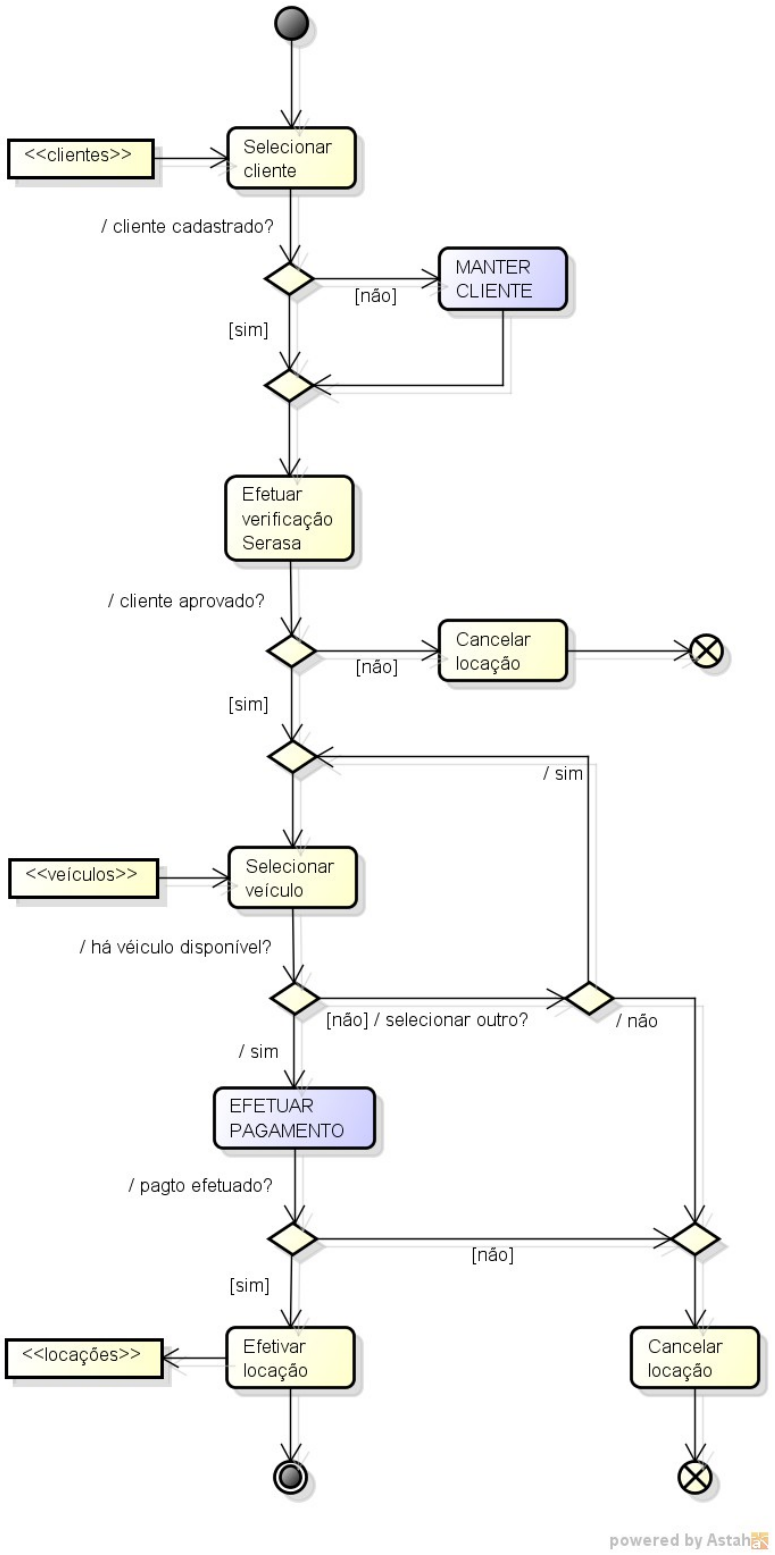
Anexo B – Template e Exemplo de Diagrama de Atividades

B.1) Template



B.2) Exemplo

O exemplo a seguir detalha a realização do caso de uso especificado no Anexo A.

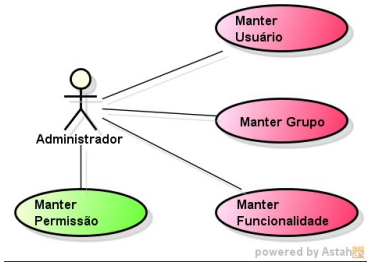


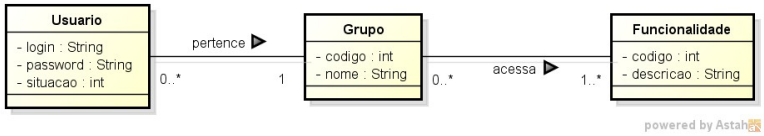
Anexo C – Template e Exemplos de Diagramas MVC

C.1) Exemplo

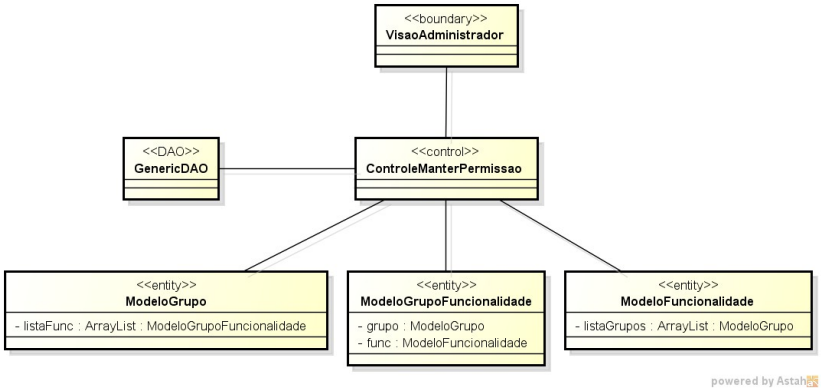
Considere o diagrama de caso de uso e o diagrama de classes, a seguir:

–





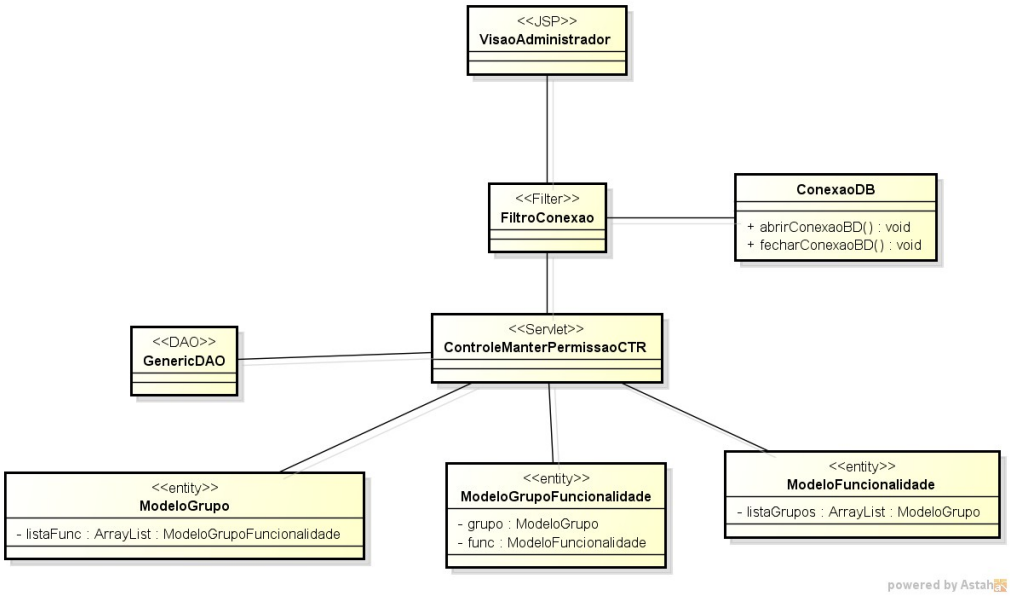
Agora considere o seguinte diagrama lógico do componente proposto, desenvolvido utilizando o conceito MVC:



Observe que o padrão MVC proposto acima segue tão somente a lógica do componente, não obedecendo padrão algum de arquitetura, como por exemplo, Swing, JSF ou Servlets.

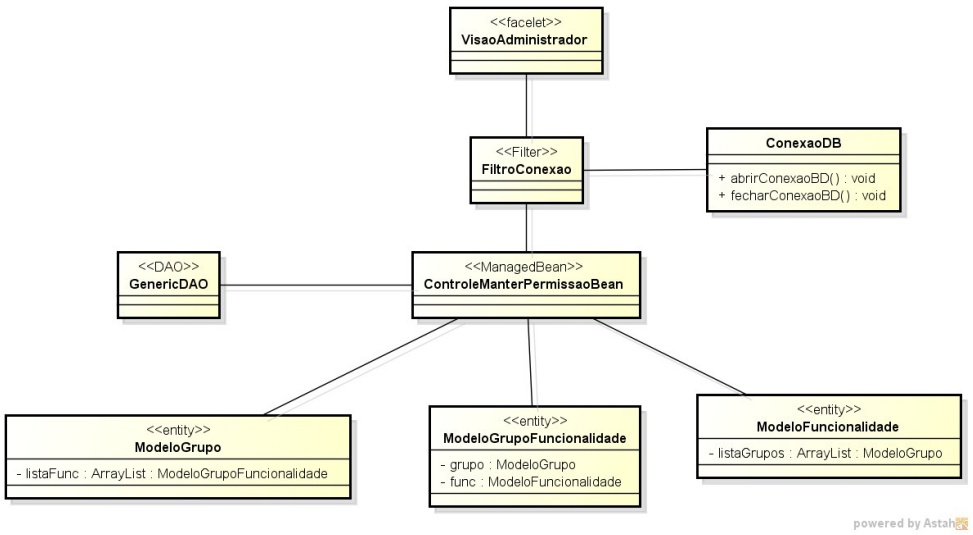
C.2) Exemplo MVC em arquitetura JSP e Servlet

Esse diagrama de classe é um exemplo do projeto físico de um componente MVC, para uma aplicação que fará uso da tecnologia **JSP** e **SERVLETS**. A classe ***VisãoAdministrador*** é um arquivo **JSP**, cujo objetivo é exibir os resultados dos processamentos realizados pelo **SERVLET** (controlador) ***ControleManterPermissaoCTR***. A conexão com o banco de dados é feita por uma classe que implementa a interface **Filter**, no diagrama representada pela classe ***FiltroConexao***. O filtro intercepta cada requisição do cliente e resposta do servidor, dessa forma o filtro abre a comunicação (sessão) com o banco de dados na requisição e encerra essa comunicação na resposta devolvida pelo servidor. A classe ***GenericDAO***, é uma classe abstrata que deve armazenar os métodos básicos de persistência. Cada classe do modelo que necessitar ser persistida deverá ter uma especialização da classe ***GenericDAO***. As classes estereotipadas como <<entity>> representa as regras de negócio da aplicação (o modelo).



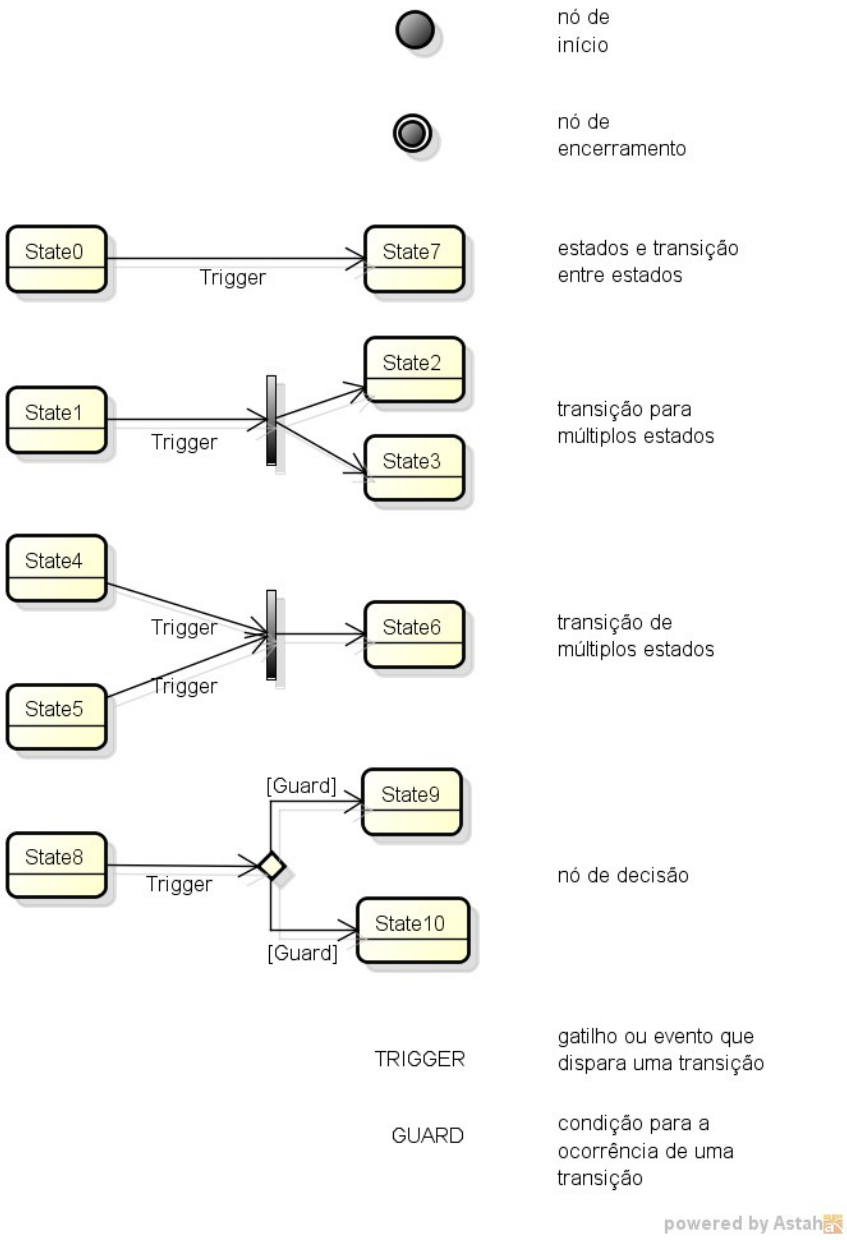
C.3) Exemplo MVC em arquitetura JSF

Esse diagrama de classe é um exemplo de um projeto físico de um componente para uma aplicação que fará uso da tecnologia **JSF**. A classe ***VisãoAdministrador*** é um arquivo **JSF**, um **facelet**, cujo objetivo é exibir os resultados dos processamentos realizados pelo **ManagedBean** (controlador) ***ControleManterPermissaoBean***. A conexão com o banco de dados é feita por uma classe que implementa a interface **Filter**, no diagrama representada pela classe ***FiltroConexao***. O filtro intercepta cada requisição do cliente e resposta do servidor, dessa forma o filtro abre a comunicação (sessão) com o banco de dados na requisição e encerra essa comunicação na resposta devolvida pelo servidor. A classe ***GenericDAO***, é uma classe abstrata que deve armazenar os métodos básicos de persistência. Cada classe do modelo que necessitar ser persistida deverá ter uma especialização da classe ***GenericDAO***. As classes estereotipadas como <<entity>> representa as regras de negócio da aplicação (o modelo).



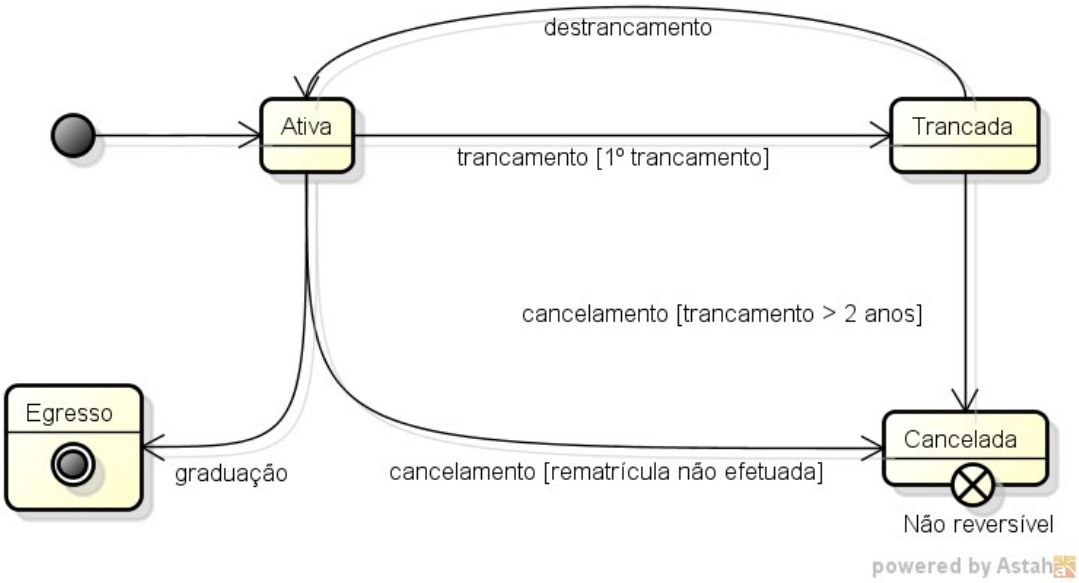
Anexo D – Template e Exemplo de Diagrama de Estados

D.1) Template

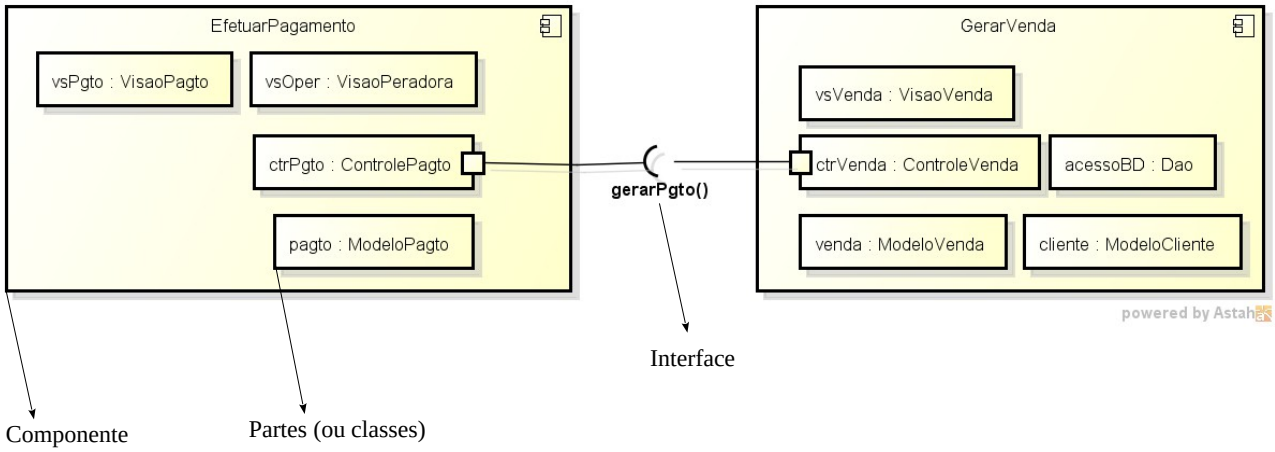


D.2) Exemplo

Neste exemplo, toma-se como objeto de estudo a classe MATRÍCULA, referente a um aluno de graduação padrão.



Anexo E – Template e Exemplo de Diagrama de Componentes



Desenvolvimento de Jogos Digitais

A documentação do Trabalho de Conclusão de Curso, no Curso de Jogos Digitais, quando for focado no desenvolvimento de um jogo, terá a seguinte estrutura:

1) Introdução
2) Documento de Conceitos
3) Documento de Proposta de Jogo
4) Roteiro (***quando aplicar***)
5) Manual do Jogo
6) Considerações Finais
7) Bibliografia

1) Introdução

A introdução deve apresentar a **proposta** do projeto, os **objetivos**, a **motivação**, a **metodologia de trabalho** e os **resultados esperados.**

2) Documento de Conceitos

A meta do Documento de Conceitos é decidir quais os objetivos do jogo, traduzidos de forma sucinta em um modelo escrito. Seus propósitos são:

1. Identificar o mercado alvo;
2. Definir os recursos do time de desenvolvimento;
3. Definir os conceitos do jogo;
4. Definir, em alto nível, detalhes de design do jogo.

É o documento que vai apresentar a proposta do jogo para uma pré-avaliação do mesmo, demonstrando se o jogo é viável e factível no tempo esperado.

O Documento de Conceito é composto pelos seguintes itens:

2.1) Premissa

A Premissa, ou Conceito Principal mantém a ideia básica do jogo. Uma premissa bem escrita consiste de somente uma ou duas sentenças, visando o jogador, descrevendo os principais atrativos e diferenciais do jogo. Pense na Premissa como aquele texto que se encaixa em um pôster ou na capa do jogo.

2.2) Motivação do Jogador

A Motivação do Jogador deve discutir de forma breve as condições de vitória ou derrota do jogo.
Pontos como:

- Como o jogador vencerá?
- Qual atrativo fará com que o jogador seja desafiado e encare o jogo até o final?

devem ser discutidos neste item.

2.3) Proposta de Individualidade

A Proposta de Individualidade é o que torna o seu jogo único. Por que os jogadores escolherão o seu jogo e não outros jogos do mercado? A Proposta de Individualidade é que fará com que o seu jogo seja ESPECIAL dentre

muitos outros.

Ela deverá responder perguntas como:

- Porque seu jogo deverá ser desenvolvido
- Porque ele é especial

e principalmente

- Porque eu vou querer jogar seu jogo?

A Proposta de Individualidade deve ter no máximo um parágrafo, contendo detalhes como as características que tornam seu jogo particularmente excepcional - itens como: gráficos, *engines* avançadas, trilha sonora, etc.

A Proposta de Individualidade pode ser pensada como o parágrafo que é incorporado à contracapa do jogo.

2.4) Público Alvo

O Público Alvo é a porção de jogadores que tem a maior probabilidade de querer jogar o seu jogo.
O Público Alvo deve incorporar questões demográficas e geográficas. Tenha certeza de pensar na característica IDADE.

2.5) Gênero

Qual gênero você escolheu para seu jogo? Se você considerar utilizar mais de um gênero no mesmo jogo, tenha cuidado para garantir que o mesmo seja compreendido tanto pela indústria de jogos quanto pelos jogadores. Muitas vezes é importante realizar uma análise de mercado para jogos do gênero que você escolheu. Buscar identificar em quais pontos as outras equipes de desenvolvimento erraram e em quais acertaram. Isso não quer dizer que você tenha que seguir exatamente o que está posto pelo mercado. Você pode acertar no que os outros erraram. Este item não deve ter mais do que um parágrafo.

2.6) Plataforma de Desenvolvimento e Requisitos de Hardware

Indique aqui qual a plataforma para qual o jogo será desenvolvido, bem como, em qual plataforma de desenvolvimento o mesmo será desenvolvido.
Ainda, busque estimar corretamente os requisitos de hardware para a execução do jogo, caso o mesmo não seja de algum console específico.

2.7) Licença

O uso de determinadas licenças ou direitos autorais pode encarecer o desenvolvimento de um jogo. Indique todas as licenças que deverão ser adquiridas para o desenvolvimento e a distribuição do jogo. Jogos esportivos com times reais, jogos de ação com super-heróis de mercado (Marvel, DC, Dark Horse), jogos baseados em filmes, devem possuir licença para ser lançados.
Lembre-se que este trabalho pode ser publicado.

OBSERVAÇÃO – Qualquer menção a um personagem com direitos autorais deverá ter anexada, a autorização por escrito e assinada do proprietário dos direitos.

2.8) Análise Competitiva

Identifique de três a cinco jogos similares ao seu e trace um comparativo entre eles, principalmente, destacando os pontos positivos de seu jogo. Porém, ao apresentar este documento, esteja preparado para identificar possíveis pontos negativos do seu jogo.

2.9) Objetivos

O que você quer causar nos jogadores? Medo, ansiedade, competitividade, diversão?
Defina em um parágrafo ou dois o que você espera causar nos jogadores que jogarão seu jogo.

3) Documento de Proposta do Jogo

A Proposta do Jogo é uma sequência do Documento de Conceito, que descreve os itens do documento anterior com maior riqueza de detalhes. O objetivo do Documento de Proposta é apresentar os detalhes do jogo.

O Documento de Proposta deve ter aproximadamente 20 páginas e também pode ser utilizado como guia para o time de desenvolvimento, antes desse começar a elaborar o Plano de Desenvolvimento do Jogo.
Um *Storyboard* pode ser anexado ao Documento de Proposta do Jogo, apresentando propostas de design e ação.
Além dos itens do Documento de Conceito, que devem ser detalhados no documento de Proposta de Jogo, devemos acrescentar ainda os seguintes itens:

3.1) Gancho

Um Gancho é um elemento que atrairá os jogadores para o jogo e o manterá interessado em jogar. "Por quê alguém compraria este jogo?" é a pergunta que deve ser respondida neste item. Escolha de 3 a 5 características do jogo que possam responder a essa pergunta. Os Ganchos podem ser baseados em itens visuais, sonoros, de jogabilidade, história. Quaisquer elementos que possam prender a atenção do jogador.

3.2) Jogabilidade

Esta seção deve listar de 10 a 20 elementos que descrevam a experiência de jogar o jogo. Algumas perguntas que devem ser respondidas são: "Quais tipos de desafios estão nesse jogo?", "Quais caminhos um jogador pode escolher em um cenário?".
O grupo deve discutir em quais atividades um jogador pode se engajar como exploração, combate, coleta de itens, resolução de puzzles, construção, gerenciamento, colaboração com outros jogadores, etc.

3.3) Características On Line

Se o jogo incluir componentes multijogadores, o time deve discutir quaisquer características pertinentes nesta seção. Podem ser discutidos elementos relacionados a times colaborativos, busca e correspondência entre jogadores, modo jogador × jogador, etc.

3.4) Tecnologia

A seção de tecnologia é opcional e deve ser incluída se existem planos de incorporar tecnologias especiais ao jogo, tanto em nível de software como em nível de hardware. Se uma tecnologia de terceiros for incorporada ao jogo, a mesma deve ser citada aqui, como uma *engine*, por exemplo.
Exemplos de tecnologias inovadoras: controles customizados para o jogo, algoritmos avançados de IA, reconhecimento de voz, reconhecimento de movimentos, etc.

3.5) Características de Arte e Áudio

É importante destacar nessa seção as características específicas de arte e áudio que são diferenciais do jogo a ser desenvolvido. Questões de licença para imagens e trilhas sonoras devem ser discutidas também nessa seção, bem como, se houver a intenção de convidar artistas para compor uma trilha sonora e/ou, uma trilha incidental.
Além disso, deve-se discutir como será trabalhada a sonoplastia do jogo.

3.6) Prólogo

O prólogo do jogo é importante por oferecer um cenário de fatos que levam até o ponto de início do jogo. Em geral, todos os fatos decorrentes do jogo dependem da história apresentada no prólogo.
Este item só deve ser incluído quando aplicável. Jogos como FIFA, por exemplo, não demanda uma história.

3.7) Sinopse

Descreve a história durante a execução do jogo. Este item não deve envolver mais do que um ou dois parágrafos e não deve detalhar todos os fatos que ocorrerão em cada uma das fases. É importante também destacar como a jogabilidade refletirá a história do jogo. Perguntas como: "O que o jogador fará no jogo?", "Quais os tipos de cenários o jogador encontrará?" ajudam a definir a jogabilidade em relação à sinopse.
Este item só deve ser incluído quando aplicável.

3.8) Descrição dos Personagens

Apresente neste item uma descrição de todos os principais personagens do jogo. Descreva pontos como nome, descrição física, caráter, e sua relevância para a história do jogo.

3.9) Conceitos de Arte

Inclua os conceitos de arte do jogo proposto, incluindo *sketchs* de personagens e cenários.
Storyboards também são bem-vindos.

4) Roteiro

Um roteiro é uma orientação, um guia e esquema das situações, cenas, ações e decisões do personagem numa história. É o material que vai apontar qual situação deve vir primeiro, qual deve vir depois, como e onde o personagem deve agir, em quais cenários, etc.
O roteiro age como um guia para a equipe de desenvolvimento.

4.1) Storyline

O conflito central, também chamada de espinha dorsal dramática. É a condensação do nosso conflito básico. Contém, em síntese, em uma frase, toda a história. Uma *storyline* deve ser breve, concisa e eficaz. Não deve ultrapassar poucas linhas e através dele devemos ficar com a noção daquilo que vamos contar.

4.2) Argumento ou Sinopse

Desenvolve o desenvolvimento dos fatos e do caráter dos personagens e as características ambientais. Aqui devemos localizar a história no tempo (*quando*) e no espaço (*onde*).

4.3) Personagens

O detalhamento das características dos participantes da história. Mostrar cada personagem com um conjunto de caraterísticas definidas. Qual sua história e seu envolvimento com os outros personagens e com o enredo geral.

4.4) Enredo ou Ação Dramática

De que maneira vamos contar essa história. A organização e estruturação das cenas sequenciadas pelas alterações do espaço e do tempo determinadas no argumento. É o momento em que se inicia o trabalho de definir o fluxograma, isto é, a estrutura de conteúdos e links que comporão a obra. É o momento de descrever as cenas e os diálogos dos personagens.

4.5) Roteiro Final

É difícil dizer que um roteiro chegou ao final porque ele é constantemente reescrito. No entanto, feita as etapas anteriores, junta-se tudo neste chamado “roteiro final”.
É importante salientar que não existem modelos pré-definidos de roteiros, muitas empresas têm um modo próprio para eles. No entanto, as etapas acima, são básicas para qualquer roteiro,

5) Manual do Jogador

O texto do manual do jogador deve instruí-lo de forma que o mesmo seja capaz de compreender suas principais nuances no que tange aos seus principais elementos, como por exemplo:

1. Cenários;
2. Personagens;
3. Recursos;
4. Equipamentos e/ou Ferramentas;
5. Armamentos;
6. Objetivos do Jogo;
7. Condições de Vitória;
8. Condições de Derrota;
9. Bônus;
10. Controle dos Personagens;
11. Configurações do Ambiente de Jogo;
12. Configurações de Controle;
13. Mapas;
14. Editores (de personagens e mapas);

15. Outros.

É importante que o manual seja ricamente ilustrado.

6) Considerações Finais

Trace aqui as considerações finais sobre o processo de produção do jogo escolhido.

- Exponha os pontos positivos do seu trabalho;
- Exponha as dificuldades encontradas;
- Considerações sobre o resultado final;
- Considerações sobre propostas de melhoria;
- Considerações sobre projetos futuros;
- Conclusão.

7) Bibliografia

Devem ser listadas todas as obras (artigos, livros, etc...) que foram usados como referência na monografia. Toda obra listada deve obrigatoriamente ter sido referenciada pelo menos uma vez no texto.

ANÁLISE COMPARATIVA

Considere que em uma comparação existem dois ou mais objetos de estudo que serão comparados dentro de um ambiente controlado, seguindo-se um método e critérios de comparação. A análise resultante de uma comparação só pode ser afirmada como verdadeira se puder ser reproduzida por outra pessoa, portanto os critérios devem ser bem estabelecidos e o ambiente controlado deve ser bem descrito. Do contrário, outras pessoas não poderão reproduzir os resultados e não poderão confirmar a veracidade de suas informações.
Outro ponto importante é que quaisquer resultados são verdadeiros dentro do ambiente controlado. Generalização de resultados só pode ser feitos com base em estudos aprofundados, usando-se predição probabilística ou outro modo de generalização (o que normalmente foge ao escopo de um TCC).
Com isso em mente, o aluno deve definir: (i) os objetos de estudo (o que será comparado); (ii) o ambiente em que serão comparados; (iii) o método de comparação e análise; (iv) os critérios de comparação.

A Análise Comparativa terá a seguinte estrutura:

1. Introdução
2. Conceitos Básicos
3. Metodologia
4. Levantamento de Dados e Análise de Resultados
5. Considerações Finais
6. Bibliografia
7. Anexos (Quando se aplicar)

ANÁLISE COMPARATIVA

1) Introdução

A introdução deve conter: (1) uma breve descrição do contexto do trabalho; (2) quais problemas existem na falta dos objetos de comparação; (3) como os objetos de comparação resolvem os problemas; (4) a motivação/justificativa do trabalho; (5) o objetivo do trabalho.

1. A descrição do contexto tem como objetivo ajudar o leitor a ter uma visão geral do trabalho. Por exemplo, se o trabalho envolve a comparação entre dois motores de jogo, o leitor deve saber que o desenvolvimento de jogos é multidisciplinar e complexo (por envolver programação, modelagem, animação, efeitos sonoros, etc...), do tempo que se leva para o desenvolvimento de um jogo e da importância do reuso no processo de desenvolvimento. E, deve saber o que é um motor de jogo. Outro exemplo, se o trabalho envolve a comparação entre dois servidores de e-mail, é necessário descrever a importância do serviço de e-mail e a complexidade em manter o serviço no ar dado à quantidade de spams recebidos.

2. A definição do problema ajuda o leitor a entender a importância dos objetos de comparação, caso não sejam bem aplicados ou não sejam usados. Como exemplo, o aluno pode descrever as consequências de se desenvolver um jogo sem um motor. Pode ser descrito também, as consequências em implementar um servidor de e-mail sem um anti-spam.

3. Este item também ajuda o leitor a entender a importância dos objetos de comparação. A diferença para o item anterior é que, neste ponto, deve ficar claro como os objetos de comparação resolvem o problema.

Como exemplo, pode ser descrito como um motor de jogo promove o reuso, e quais os seus benefícios (geralmente um motor de jogo envolve uma IDE que integra funcionalidades como compilação, depuração, deployment, facilidades de HCI para desenvolvimento, etc.).

4. Uma vez que o contexto esteja claro para o leitor, ele deve entender porque é relevante o seu trabalho. A motivação e justificativa estão intimamente relacionadas com o problema e os objetos de comparação. Por exemplo, existem diversos motores que permitem o desenvolvimento do jogo, mas não se conhece um estudo que mostre as características de cada um e a performance de um mesmo jogo feito em diferentes motores. Outro exemplo, os servidores de e-mail permitem a integração com softwares de antispam, nesse caso é interessante mostrar as características desses servidores e a comparação entre eles baseado em uma simulação de ataque (por exemplo, DoS).
5. Este item é dividido em objetivo geral e objetivos específicos. Em objetivo geral, deve ser apresentada a finalidade do projeto. Objetivos específicos descrevem os objetivos parciais do projeto. Como exemplo, um estudo não é um objetivo, mas a identificação de características levantadas por um estudo pode ser um objetivo específico. A comparação não é um objetivo, mas a identificação da diferença de performance entre dois objetos de estudo (que foi descoberta pela comparação) pode ser considerada um objetivo.

2) Conceitos Básicos

Neste capítulo, devem ser descritos todos os conceitos relacionados com o projeto. O objetivo é fornecer ao leitor todo o conteúdo teórico necessário para que ele entenda os demais capítulos do TCC. Por exemplo, se uma comparação visa a identificar o impacto de diferentes algoritmos de *anti-aliasing* na performance de um jogo, então o leitor deve saber o que é *anti-aliasing*, quais são os algoritmos existentes e como funcionam. Mas, antes disso, como o *anti-aliasing* é aplicado durante ou após o processo de renderização, então leitor deve saber o que é renderização, quais as principais técnicas de renderização, em qual delas o *anti-aliasing* é aplicado (rasterização) e como funciona essa técnica. Caso o objetivo seja avaliar o desempenho entre banco de dados, o leitor precisa saber o que é um banco de dados e quais as formas para se medir desempenho em um banco de dados.

2.1) Definição dos conceitos

Nesta seção, devem ser descritos todos os conceitos relacionados com o objeto de comparação. Como no exemplo anterior, se os objetos de comparação são diferentes algoritmos de *anti-aliasin*g, então aqui deve ser descrito o que é renderização, rasterização e *anti-aliasing*. Caso os objetos de comparação são diferentes banco de dados, então deve-se ser descrito o que é SGBD e banco de dados.

2.2) Objetos analisados

Nesta seção, devem ser descritos os objetos de comparação. Continuando o exemplo anterior, aqui devem ser detalhados como funcionam os algoritmos a serem comparados, se pertencem a alguma classificação, quais os impactos esperados sob a performance, qualidades e desvantagens. Assim como no exemplo anterior, é necessário especificar os gerenciadores de bancos de dados que serão comparados, e também suas características, qualidades, vantagens e desvantagens.

3) Metodologia

Neste capítulo, deve ser descrita, em detalhes, como o aluno realizará a comparação e como realizará a análise dos dados da comparação. Por exemplo: quais os critérios de comparação (performance, qualidade, segurança, etc.) e como serão aplicados, qual o método de análise dos dados (análise quantitativa, qualitativa, etc.), em que ambiente de hardware e em que configuração será feita a comparação, em que ambiente de software e quais configurações e em que domínio (por exemplo, a análise usará um jogo de FPS específico, durante uma fase específica).

3.1) Definição dos critérios e métodos de análise

Nesta subseção, devem ser listados e explicados os critérios e métodos de análise.
Toda comparação deve ser realizada com base em um grupo de critérios. Um critério serve para que a comparação siga uma regra (ou padrão) que definirá como extrair os dados necessários. Por exemplo, suponha que o aluno deseja identificar qual o algoritmo de *anti-aliasing* mais complexo. Para que obtenha uma resposta,

o aluno precisar definir como classificará um algoritmo de acordo com a complexidade. Isto é, deve saber responder quando um algoritmo é complexo ou não e quais os níveis de complexidade. Para isso, o aluno pode definir o seguinte critério: a complexidade será calculada com base em análise de complexidade assintótica (que é uma análise conhecida para classificação de nível de complexidade de um algoritmo). Outro exemplo seria caso o aluno deseja identificar qual banco de dados tem melhor desempenho. Para isso, é necessário definir como serão feitas as consultas e como extrair os dados para analisar os resultados.
Toda comparação gera resultados. Os resultados devem ser compilados com base em algum método conhecido, para que o aluno possa gerar suas conclusões. Por exemplo, para definir se o resultado obtido por um algoritmo de *anti-aliasing* é melhor ou pior que o de outro algoritmo, o aluno deve primeiro definir o nível de significância (que define a partir de que ponto a diferença de resultados será considerada significante). Neste caso, se a diferença de resultado for abaixo do nível de significância, considera-se que o resultado foi o mesmo para ambos.
Quando a comparação envolver uma amostragem de dados, o aluno pode usar teste de hipótese, como teste de duas ou outro método de análise de resultados.

3.2) Definição do ambiente

Nesta seção, deve ser descrito o ambiente de hardware e software em que a comparação será realizada, assim como as suas delimitações. A descrição de hardware geralmente envolve a configuração de máquina e rede (caso aplicável). A descrição de software envolve o sistema operacional, máquina virtual, configurações e outros aplicativos necessários para a comparação ser executada. Também é importante descrever a aplicação (e configuração) que será usada como alvo para que a comparação seja executada.
Por exemplo, se os algoritmos de *anti-aliasing* forem comparados, é importante descrever os componentes de hardware da máquina, configuração da máquina, sistema operacional, em que jogo/fase/configuração os algoritmos foram comparados e em que ambiente de desenvolvimento/game engine, o jogo foi desenvolvido.

3.3) Planejamento de execução

Nesta seção, deve ser descrito como será feita a execução das comparações. Se há fatores externos que possam influenciar no resultado da comparação, eles devem ser isolados ou minimizados ou controlados a medida do possível. Por exemplo, *anti-aliasing* não é o único recurso que influencia na performance do jogo. Portanto, é importante que o jogo seja configurado para que a influência do *anti-aliasing* seja perceptível e outros recursos não mascarem o seu impacto.

4) Levantamento dos dados e análise dos resultados

Neste capítulo, devem ser apresentados os dados da comparação e a análise dos resultados, seguindo-se os critérios, métodos e plano de execução estabelecidos no capítulo anterior.
Dados podem ser expressos em tabelas e gráficos, mas devem sempre ser seguidos de explicação. Uma tabela ou um gráfico não têm significado se não houver uma descrição sobre eles. A descrição dos resultados deve ajudar o leitor a entender os números expressos.
A análise dos resultados deve seguir o método de comparação estabelecido no capítulo anterior. É importante entender que o objetivo da comparação é a análise. Não basta apresentar os dados e apontar diferenças de valores. É preciso concluir o que os resultados significam.

5) Considerações finais

Neste capítulo, devem ser descritas as considerações finais. Uma estrutura muito utilizada é: retomar o assunto para que o leitor relembre o contexto do trabalho, qual era o problema e o objetivo da comparação; um resumo do que foi comparado, em que ambiente e com que critérios e métodos a comparação foi realizada; resumo da análise dos resultados, enfatizando nas descobertas obtidas pela comparação; possíveis fatores que ameaçam a validação dos resultados; trabalhos futuros.

6) Bibliografia

Devem ser listadas todas as obras (artigos, livros, etc...) que foram usados como referência na monografia. Toda obra listada deve obrigatoriamente ter sido referenciada pelo menos uma vez no texto.

Anexos (quando se aplicar)

PROVAS DE CONCEITO

Uma prova de conceito é um termo utilizado para especificar um modelo ou protótipo que possa provar o conceito (teórico) estabelecido por uma pesquisa técnica. A prova de conceito permite demonstrar na prática uma metodologia e tecnologias envolvidas para implementar um protótipo a partir de um método ou de uma ideia, com o propósito de verificar sua viabilidade.

Em Tecnologia da Informação (TI), a prova de conceito é caracterizada pelo desenvolvimento de um protótipo para provar a viabilidade de um projeto de Sistemas, Rede de Computadores, Banco de Dados ou Infraestrutura.

A prova de conceito terá a seguinte estrutura:

1. Introdução
2. Conceitos Básicos
   2.1 Definição dos conceitos
   2.2 Objeto analisado
3. Metodologia
   3.1) Definição dos critérios e métodos de análise
   3.2) Levantamento de hipóteses (quando aplicável)
   3.3) Definição do ambiente
   3.4) Protótipo e Planejamento de execução
4. Levantamento dos dados e análise dos resultados
5. Considerações finais
6. Bibliografia

Anexos (quando se aplicar)

PROVA DE CONCEITO
1) Introdução
A introdução deve conter: (1) uma breve descrição do contexto do trabalho; (2) a motivação/justificativa do trabalho; (3) o objetivo do trabalho; (4) a metodologia de trabalho; e (5) estrutura do trabalho. 1. A descrição do contexto tem como objetivo ajudar o leitor a ter uma visão geral do trabalho. Por exemplo, se o trabalho envolve a avaliação de um algoritmo de criptografia na transmissão de dados em um cluster HPC, deve-se descrever a importância da criptografia para um ambiente em rede e também qual seria o impacto no desempenho do cluster. Outro exemplo, caso o trabalho envolve a implementação de um jogo utilizando o kinect, é necessário descrever que o desenvolvimento de jogos é multidisciplinar e complexo (por envolver programação, modelagem, animação, integração com equipamentos, etc...), do tempo que se leva para o desenvolvimento de um jogo e da importância da interação humano-computador (IHC). 2. Para o desenvolvimento é necessário descrever quais foram as motivações que o levou para o desenvolvimento do trabalho, pode envolver a formulação do problema. A justificativa responde a pergunta do porque fazer o trabalho. Pode-se incluir também as possíveis contribuições esperadas. 3. Este item é dividido em objetivo geral e objetivos específicos. Em objetivo geral, deve ser apresentada a finalidade do projeto. Objetivos específicos descrevem os objetivos parciais do projeto.

4. Este item descreve todos os passos necessários para o desenvolvimento do projeto, ou seja, os procedimentos necessários para atingir os objetivos propostos.

5. A estrutura descreve como será composto o trabalho, ou seja, descreve brevemente o conteúdo dos capítulos do trabalho.

2) Conceitos Básicos

O principal objetivo desse capítulo é fazer uma revisão bibliográfica dos conceitos que estão relacionados com o objeto principal de estudo do trabalho e fornecer ao leitor todo o conteúdo teórico necessário para que ele entenda os demais capítulos do TCC.

2.1) Definição dos conceitos

Nesta seção, devem ser descritos todos os conceitos relacionados com o método ou com a ideia em questão. Por exemplo, se o objetivo é avaliar um algoritmo de criptografia em um cluster HPC, deve-se descrever o que é um algoritmo de criptografia, cluster de computadores e computação de alto desempenho. Caso o objetivo seja o desenvolvimento de um jogo com Kinect, deve-se decrever o kinect, suas interfaces, suas leituras, restrições, descrever a conexão do Kinect com o PC.

2.2) Objetos analisados

Nesta seção, devem ser descritos os objetos de comparação. Continuando o exemplo anterior, aqui devem ser detalhado como funciona o algoritmo, se pertence a alguma classificação, quais os impactos esperados em relação ao desempenho do cluster, qualidades e desvantagens. No outro exemplo, devem ser detalhado como funciona a API de conexão entre o computador e o Kinect.

3) Metodologia

Este capítulo tem como objetivo descrever os detalhes de como será realizado o desenvolvimento do projeto, por exemplo, o ambiente necessário para execução do teste e todo o planejamento para sua execução ou também os procedimentos para desenvolvimento do jogo.

3.1) Definição dos critérios e métodos de análise

Uma prova de conceito deve ser realizada com base em um grupo de critérios. Um critério serve para que a prova de conceito siga uma regra (ou padrão) que definirá como extrair os dados necessários para análise.
Por exemplo, se você deseja avaliar um algoritmo de criptografia em um cluster HPC é necessário definir quais os critérios que serão utilizados e a partir disso, como serão analisados os resultados. No caso de um cluster HPC um bom critério seria o desempenho baseado na transmissão dos dados. Neste contexto, o desempenho pode ser analisado pela cifragem e decifragem dos dados, pode ser baseado na qualidade de serviço (QoS) ou também baseado no tamanho e quantidade de pacotes.

3.2) Levantamento de hipóteses

Nesta seção, deve ser descritos o que pode acontecer (hipóteses) em relação aos critérios e métodos definidos na etapa anterior. Ou seja, a escolha de um critério X pode te levar a uma situação, já a escolha do critério Y leva a outra situação. Essas hipóteses precisam ser consideradas para a análise dos resultados.
Por exemplo, para avaliar o desempenho em uma transmissão em rede, o critério baseados em QoS tem como resultado diminuir o tempo de transmissão. Já a avaliação baseada no tamanho dos pacotes pode refletir em um aumento do tempo de transmissão.

3.3) Definição do ambiente

Nesta seção, deve ser descrito o ambiente de hardware e software em que a avaliação de desempenho será realizada, assim como as suas delimitações. A descrição de hardware geralmente envolve a configuração de máquina e rede. A descrição de software envolve o sistema operacional, máquina virtual, configurações e outros aplicativos necessários para a avaliação de desempenho ser executada.

3.4) Protótipo e Planejamento de execução

Nesta seção, deve ser descrito como será feita a execução da avaliação de desempenho. Se há fatores externos

que possam influenciar no resultado da avaliação de desempenho, eles devem ser isolados ou minimizados ou controlados a medida do possível. Por exemplo, *throughput* não é o único fator que influencia no desempenho da rede, existem outros, por exemplo, *delay*.

4) Levantamento dos dados e análise dos resultados

Neste capítulo, devem ser apresentados os dados da avaliação de desempenho e a análise dos resultados, seguindo-se os critérios, métodos e plano de execução estabelecidos no capítulo anterior.
Dados podem ser expressos em tabelas e gráficos, mas devem sempre ser seguidos de explicação. Uma tabela ou um gráfico não têm significado se não houver uma descrição sobre eles. A descrição dos resultados deve ajudar o leitor a entender os números expressos.
A análise dos resultados deve seguir o método de avaliação de desempenho estabelecido no capítulo anterior. É importante entender que o objetivo da avaliação de desempenho é a análise. Não basta apresentar os dados e apontar diferenças de valores. É preciso concluir o que os resultados significam.

5) Considerações finais

Neste capítulo, devem ser descritas as considerações finais. Uma estrutura muito utilizada é: retomar o assunto para que o leitor relembre o contexto do trabalho, qual era o problema e o objetivo; em que ambiente e com que critérios e métodos foi realizado; resumo da análise dos resultados, enfatizando nas descobertas obtidas; possíveis fatores que ameaçam a validação dos resultados; trabalhos futuros.

6) Bibliografia

Devem ser listadas todas as obras (artigos, livros, etc...) que foram usados como referência na monografia. Toda obra listada deve obrigatoriamente ter sido referenciada pelo menos uma vez no texto.

Anexos (quando se aplicar)