Documentação de um Produto de Software

Versão 3.0

Autor: João Victor da Silva, Luiz Guilherme Zanella Lopes.

2024

ÍNDICE DETALHADO

Prefácio

O objetivo deste documento é fornecer um roteiro para o desenvolvimento de sistemas de software utilizando os princípios da engenharia de software orientada a objetos com notação UML (*Unified Modeling Language*). É destinado a todos os alunos dos cursos de Ciência da Computação, Sistemas de Informação e Processamento de Dados, apoiando as disciplinas de Metodologia de Desenvolvimento de Sistemas, Engenharia de Software I, Engenharia de Software II, entre outras, além do Trabalho de Graduação (TG).

Modelo da Documentação

Esta é a parte mais importante do texto pois apresenta um roteiro de documentação orientado a objetos de sistemas de software utilizando notação UML, desde a fase inicial do projeto de software até a sua implantação.

1. Introdução ao Documento

O objetivo deste capítulo é apresentar o projeto. Para tal, deve-se desenvolver um texto, com as seguintes características: impessoalidade, objetividade, clareza, precisão, coerência e concisão. A introdução deve abrange os itens a seguir.

1.1. Tema

Plataforma de streaming especializado para exibir aulas gravadas da UFFS e de outras Universidades Federais eventualmente.

1.2. Objetivo do Projeto

O objetivo geral deste projeto é desenvolver uma plataforma de streaming educacional que facilite o acesso à educação de qualidade na Universidade Federal da Fronteira Sul e outras instituições de ensino. A plataforma visa atender às necessidades de aulas a distância e de reposição, além de oferecer recursos para o ensino híbrido e complementar.

1.3. Delimitação do Problema

A UFFS segue atrasada na implementação de um sistema de ensino híbrido, que combina aulas presenciais e remotas de forma integrada e eficiente. Atualmente, a universidade não aplica esse sistema e nem dispõe de uma plataforma unificada que permita aos professores e alunos uma transição entre os ambientes de aprendizagem online e presencial. Esse problema é agravado pela falta de ferramentas adequadas, o que resulta em uma experiência de ensino e aprendizagem ineficiente.

1.4. Justificativa da Escolha do Tema

O tema foi escolhido devido à uma dificuldade exibida pelos professores e alunas em fazer reposição de aulas que foram perdidas ou de revisar temas que foram abordados anteriormente. Com uma plataforma que permita o armazenamento e exibição das aulas gravadas, facilitaria na hora de revisar conteúdos passados anteriormente em uma matéria específica além de poder relembrar conteúdos de matérias que são pré-requisitos de outras. Outrossim é a possibilidade de recuperar aulas que por algum motivo tiveram que ser canceladas ou no qual um aluno fora impossibilitado de participar por motivo pessoal.

1.5. Método de Trabalho

O método que será usado para desenvolvimento da plataforma será o de cascata, no qual serão

elencados requisitos, realizado a descrição dos casos de uso e feita a análise e validação com os stakeholders. Após isso será entrado em um cenário de implementação, no qual será construído o system design do projeto com informações técnicas, e codificação das funcionalidades necessárias para atender aos casos de uso. Por fim serão feitos os testes e se concluídos será feita a implementação da plataforma para uso do cliente final que são os professores e estudantes da faculdade.

[1] Para maiores detalhes dos tipos de processos de desenvolvimento de software consultar o livro Engenharia de Software – Roger Pressman – 5ª edição - Capítulo 2.

1.6. Organização do Trabalho

O documento estará organizado de forma a seguir o modelo cascata que será utilizado durante o desenvolvimento do projeto, logo seguirão as mesmas etapas: Descrição do problema e elencar requisitos, system design e codificação, testes, implantação e manutenção.

1.7. Glossário

Plataforma de Streaming: Plataforma de exibição de vídeos ou livestreams, como exemplo a Netflix, Disney Plus, etc.
Stakeholders: Envolvidos no projeto, principalmente clientes e interessados;
System Design: Detalhes técnicos de implementação e design de infraestruturas e requisitos que serão implementados;

2. Descrição Geral do Sistema

Este capítulo tem como objetivo descrever de forma geral o sistema, o escopo e as principais funções. A descrição geral do sistema deve abrange os itens a seguir. Como referência pode-se consultar e utilizar o modelo rup_vision_sp.dot – artefato do RUP.

2.1. Descrição do Problema

O problema a ser resolvido é a dificuldade na recapitulação e reforço do estudo feito presencialmente na faculdade de forma remota. Os principais afetados pelo sistema serão os estudantes e professores, que serão impactados de forma que facilitará o entendimento de conteúdos complexos passados em aula, permitindo rever a explicação quantas vezes forem necessárias para melhor entendimento das partes. A solução se constitui em uma plataforma que permita acesso de vídeos em separados por componentes curriculares e módulos internos dentro desses componentes no qual serão definidos pelos professores, que serão responsáveis por fazer upload dessas aulas no sistema.

2.2. Principais Envolvidos e suas Características

2.1.1. Usuários do Sistema

- Os principais usuários do sistema serão os professores que ministram aulas na UFFS.
- Os estudantes matriculados na UFFS.
- A equipe técnica responsável pela manutenção e operação do sistema na UFFS.
- Os coordenadores e diretores responsáveis pela gestão acadêmica dos cursos.

2.1.2. Desenvolvedores do Sistema

O desenvolvimento do sistema será realizado por uma equipe diversificada que inclui estudantes do curso de Ciências da Computação, professores e membros da administração da UFFS.

2.3. Regras de Negócio

Na plataforma é necessário que o estudante possa acessar, após uma forma de identificação por login, um catálogo de componentes curriculares, nos quais serão listados as vídeo-aulas e aulas gravadas feitas em sala, dessa forma como regras de negócio podem se citar:

1. Autenticação

É necessário que o usuário que irá acessar o sistema forneça suas informações de usuário e senha que utiliza hoje nas plataformas da UFFS, com isso será feita a identificação de aluno ou professor e liberadas as permissões de acesso às outras partes da plataforma.

2. Catálogo de Vídeos

Após acesso do usuário, ele deve ter permissão de acessar páginas de componentes curriculares nos quais ele está matriculado ou que já cursou anteriormente, dentro de cada componente curricular

serão listadas as vídeo-aulas e aulas gravadas pelos professores para acesso do usuário.

3. Exibição de Vídeo

É necessário que após a seleção da aula desejada, seja exibido um player de vídeo que permita o usuário reproduzi-lo, pausar, avançar e recuar no tempo e manipular o volume.

4. Upload de Vídeos

É necessário que usuários com cargo de administrador ou professor tenham a possibilidade de fazer upload das vídeo-aulas ou aulas gravadas nos componentes curriculares nos quais tenham acesso.

5. Descrição de Vídeos

É necessário que para cada vídeo que o professor colocar na plataforma, seja dada uma descrição mínima sobre o que se trata a aula, e caso for preferência, anexado conteúdos extras como materiais para leitura, sumários, resumos, etc.

3. Requisitos do Sistema

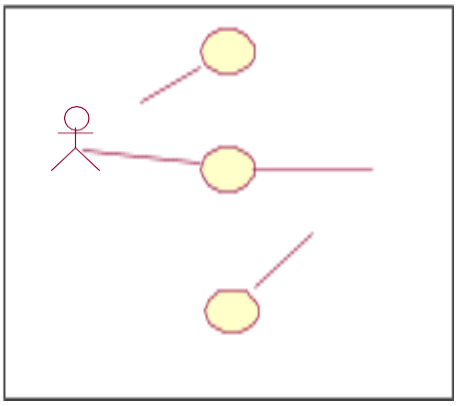
3.1. Requisitos Funcionais

IDENTIFICADOR	DESCRIÇÃO	CLASSIFICAÇÃO
RF01	O sistema deve proporcionar aos alunos uma forma eficiente de acessar conteúdos educacionais, gravações de aulas, materiais de leitura e recursos complementares. O acesso aos conteúdos deve ser simplificado através de uma **interface de usuário clara e organizada**, que permita aos alunos localizar e interagir com os materiais de forma rápida.	Essencial
RF02	Os alunos poderão **pausar, retroceder, avançar, ajustar o volume e selecionar a qualidade de reprodução** das aulas gravadas para uma experiência personalizada e adaptada às suas necessidades.	Essencial
RF03	Os vídeos postados na plataforma deverão possuir as seguintes informações: **Título (string), Descrição (string), Data (temporal), Tempo de duração: (temporal) e Componente Curricular (associação)**	Importante
RF04	Os professores têm a capacidade de acessar e visualizar estatísticas detalhadas sobre o uso dos materiais educacionais postados em seus cursos.	Desejável
RF05	O sistema deve fornecer um mecanismo para enviar notificações e alertas aos **usuários**, mantendo-os informados sobre eventos importantes relacionados às suas atividades acadêmicas e interações na plataforma.	Importante
RF06	O usuário deve poder realizar autenticação com seu **usuário** (string) e **senha** (string) fornecidos pelos professores ou pela própria universidade	Essencial
RF07	Os usuários da plataforma serão divididos em três níveis de acesso, esses serão: **Estudante, Professor, Administrador**.	Importante
RF08	O usuário **Estudante** poderá fazer comentários nas vídeo-aulas postadas na plataforma, os quais poderão ser administrados pelos professores e administradores	Desejável
RF09	Os usuários do tipo **Professor e Administrador** poderão adicionar uma descrição personalizada às	Importante

	aulas postadas, contendo anexos de materiais, resumos e sumários.	
RF10	Os usuários devem podem se desconectar da plataforma quando desejar a partir de um botão de fácil acesso	Essencial

- Postar uma aula gravada:
    - Facilitar o acesso dos alunos a conteúdos educacionais.

Neste item devem ser apresentados os requisitos funcionais que especificam ações que um sistema deve ser capaz de executar, ou seja, as funções do sistema. Os requisitos funcionais geralmente são melhor descritos em diagramas de caso de uso, juntamente com o detalhamento dos atores e de cada caso de uso.
A seguir é apresentada a notação básica de um diagrama de caso de uso.



Caso de Uso 1
ATOR 1
Caso de Uso 2
ATOR 2
Caso de Uso N

Notação básica do diagrama de caso de uso.

Cada ator do diagrama de caso de uso deve ser descrito de forma sucinta (2 linhas) e cada caso de uso deve ser especificado. A seguir são apresentados itens básicos para a especificação dos casos de uso do diagrama.

- ✔ Nome do Caso de Uso
- ✔ Breve descrição
- ✔ Atores envolvidos
- ✔ Pré-condições
- ✔ Seqüência de Eventos ou Fluxo Principal de Eventos
- ✔ Pós-condições;
- ✔ Exceções ou Fluxo Secundário de Eventos
- ✔ Observações

3.2. Requisitos Não-Funcionais

IDENTIFICAÇÃO	DESCRIÇÃO	CLASSIFICAÇÃO
RNF01	O sistema deve garantir a segurança dos dados dos **professores/estudantes** e a privacidade das informações. Isso inclui a proteção de informações pessoais e educacionais, bem como a segurança dos conteúdos gravados e postados. O sistema deve ser projetado para resistir a tentativas de acesso não autorizado, roubo de dados e outras ameaças cibernéticas.	Essencial
RNF02	O usuário deve ter uma exibição clara dos vídeos postados em uma qualidade superior a **480p** em um tempo de carregamento que seja **aceitável (preferencialmente não mais que 1 minuto de espera)**	Importante
RNF03	A plataforma deverá ser acessada pelo navegador do usuário de forma prática e sem tempo de carregamento excessivo, sendo compatível com, pelo menos, o **Firefox, Google Chrome, Microsoft Edge e Safári**, além de ser possível acessar por esses mesmos pelo dispositivo móvel.	Importante
RNF04	A plataforma deve possuir responsividade adaptável a qualidades dos seguintes dispositivos: celular (**<460px**), tablets (**>480px <1024px**) e laptop ou computador pessoal (**>1024px**)	Importante
RNF05	O usuário do tipo **Professor** deverá ter permissão de realizar upload e exclusão de vídeos, reprodução desses e deverá ser restrito a postar somente aos componentes curriculares ministrados pelo mesmo, além de não poder acessar áreas administrativas da plataforma	Essencial
RNF06	O usuário de tipo **Estudante** deverá ter acesso restrito a assistir as aulas postadas somente nos componentes curriculares nos quais **está matriculado**, ou nos quais **já cursou em algum semestre anterior**	Essencial
RNF07	O usuário do tipo **Administrador** deve ter acesso a todas as áreas da plataforma, incluindo as administrativas para gerênciais os usuários do sistema	Essencial

Neste item devem ser apresentados os requisitos não funcionais, que especificam restrições sobre os serviços ou funções providas pelo sistema. A seguir são apresentados alguns tipos de requisitos não funcionais.

3.3. Protótipo

Neste item deve ser apresentado o protótipo do sistema que consiste na interface preliminar contendo um subconjunto de funcionalidades e telas. O protótipo deve ser incrementalmente evoluído até a concordância completa dos requisitos previstos para o sistema, de comum acordo com o usuário. O protótipo é um recurso que deve ser adotado como estratégia para levantamento, detalhamento, validação de requisitos e modelagem de interface com o usuário (usabilidade).

As telas do sistema podem ser criadas na própria linguagem de desenvolvimento ou em qualquer outra ferramenta de desenho. Cada tela deve possuir uma descrição detalhada do seu funcionamento. Alguns itens importantes na descrição são:

- Objetivo da tela;
- De onde é chamada e que outras telas pode chamar;
- Regras:
  - ✔ Domínio (tamanho de campo, tipo de dados que aceita valor default);
  - ✔ Tipo de usuários que podem acessar;

- ✔ Lógica de negócio (campos obrigatórios, validade entre datas, preenchimento anterior de um campo para efetuar uma operação, etc).

A descrição detalhada das telas deve registrar informações que possam ser consultadas na implementação do sistema, facilitando, agilizando e minimizando erros de implementação e na execução de testes.

3.3.1. Diagrama de Navegação

Neste item deve ser apresentada a seqüência de navegação das telas.

3.4. Métricas e Cronograma

Neste item devem ser estimados os esforços necessários em termos de recursos alocados e tempo para a obtenção do sistema. Para realizar a estimativa, indicam-se o uso de alguma técnica de métrica, como Pontos de Função ou Pontos de Caso de Uso.

Após os cálculos de métricas deve-se elaborar o cronograma detalhado do sistema, que contempla todas as tarefas descritas e os recursos alocados para cada tarefa, com datas para início e término de cada atividade. A seqüência das tarefas e a divisão entre os recursos devem ser realizadas de acordo com o processo de desenvolvimento de software escolhido para o desenvolvimento do sistema, descrito no item 1.5.

Para elaboração do cronograma pode-se utilizar uma ferramenta como o Microsoft Project.

4. Análise e *Design*

Este capítulo tem como objetivo analisar e detalhar a solução do sistema de acordo com os requisitos levantados e validados no capítulo 3. Para isso, deve-se ter uma visão geral da arquitetura do sistema e a modelagem da solução do sistema através de diagramas.

4.1. Arquitetura do Sistema

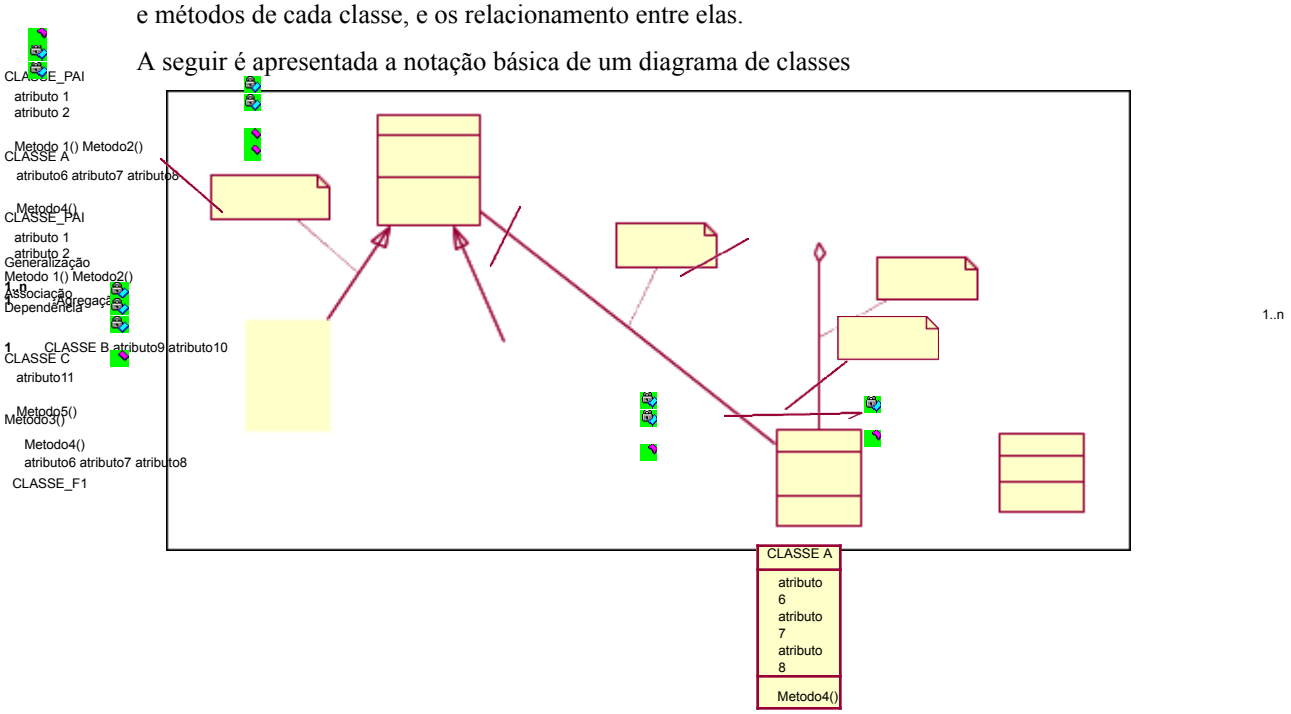
Neste item deve ser apresentada a arquitetura de infra-estrutura do sistema, demonstrando o tipo de arquitetura que será utilizada (por exemplo, cliente/servidor de n-camadas), a configuração de hardware, de rede e de software a serem utilizados, bem como o dimensionamento mínimo de conexões.

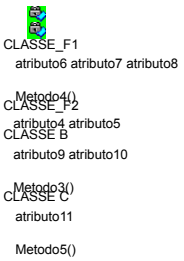
4.2. Modelo do Domínio

Neste item deve ser apresentado o modelo do domínio, que representa um primeiro modelo conceitual do diagrama de classes. Posteriormente, esse diagrama deve ser validado e complementado para compor o diagrama de classes final.

O diagrama de classes deve possuir todas as classes identificadas do sistema, deve conter os atributos e métodos de cada classe, e os relacionamento entre elas.

A seguir é apresentada a notação básica de um diagrama de classes





Notação básica do diagrama de classes.

4.3. Diagramas de Interação

O diagrama de interação é composto pelos diagramas de seqüência e colaboração (comunicação, versão 2.0 UML) e modela os aspectos dinâmicos do sistema, mostrando a interação formada por um conjunto de objetos permitindo identificar mensagens que poderão ser enviadas entre eles.

4.3.1. Diagrama de Seqüência

Neste item devem ser apresentados os diagramas de seqüência essenciais ao sistema. Um diagrama de seqüência representa interações de objetos organizadas em uma seqüência temporal, apresentando os objetos que participam da interação e a seqüência das mensagens trocadas. O diagrama de seqüência deve validar o diagrama de classes e vice-versa.

A seguir é apresentada a notação básica de um diagrama de seqüência.

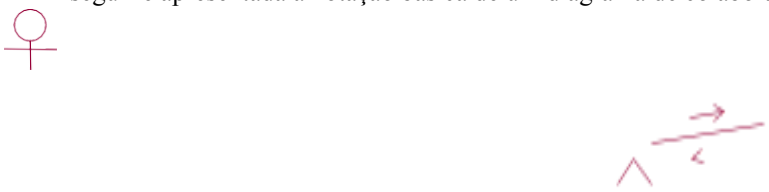
Notação básica do diagrama de seqüência.

4.3.2. Diagrama de Colaboração / Comunicação

Esse diagrama é uma alternativa para o diagrama de seqüência (item 4.3.1). Neste item devem ser apresentados os diagramas de colaboração/comunicação essenciais ao sistema. Um diagrama de colaboração descreve um padrão de interação entre objetos, apresentando os objetos que participam da interação bem como os seus links e mensagens trocadas.

Geralmente as ferramentas CASE geram automaticamente o diagrama de colaboração/comunicação a partir do diagrama de seqüência.

A seguir é apresentada a notação básica de um diagrama de colaboração/comunicação.



Notação básica do diagrama de colaboração.

4.4. Diagrama de Classes

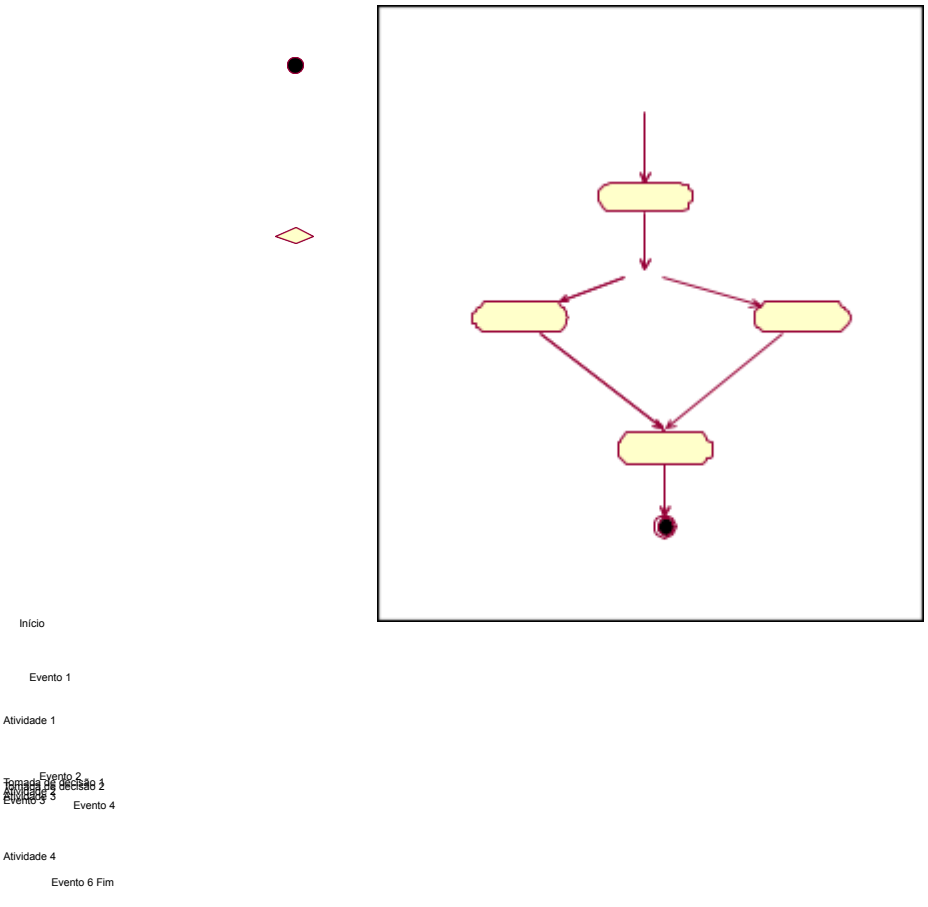
Neste item deve ser apresentado o diagrama de classes completo e validado.

4.5. Diagrama de Atividades

Neste item deve ser apresentado o diagrama de atividades, que representa o detalhamento de tarefas e o fluxo de uma atividade para outra de um sistema.

Nem todos os sistemas necessitam da elaboração do diagrama de atividades, pois nem todas as tarefas do sistema necessitam de um detalhamento. Com isso, deve-se analisar a real necessidade e no que este diagrama irá auxiliar na implementação do sistema, como: detalhamento de *workflow*, de métodos, entre outros.

A seguir é apresentada a notação básica de um diagrama de atividades.



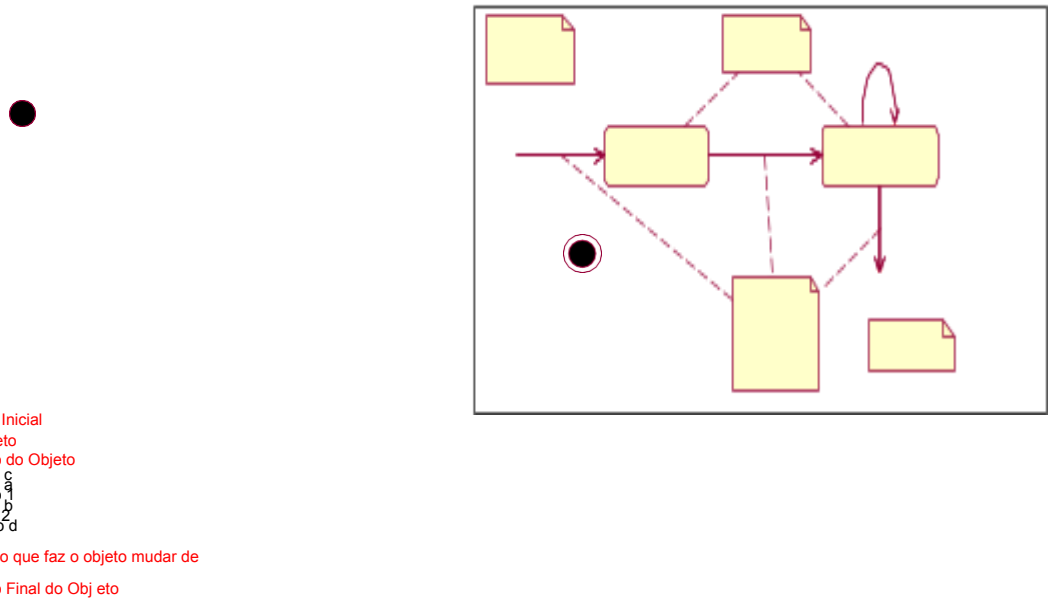
Notação básica do diagrama de atividades.

4.6. Diagrama de Estados

Neste item deve ser apresentado o diagrama de estados, que especifica as seqüências de estados pelas quais o objeto pode passar durante seu ciclo de vida em resposta a eventos.

Nem todos as classes necessitam da elaboração do diagrama de estados, pois nem todas as classes mudam muito de estado no seu ciclo de vida. Com isso, deve-se analisar a real necessidade desse diagrama para o desenvolvimento do sistema.

A seguir é apresentada a notação básica de um diagrama de estados.

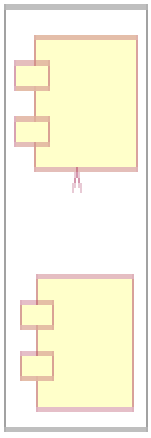


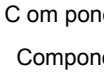
Notação básica do diagrama de estados.

4.7. Diagrama de Componentes

Neste item deve ser apresentado o diagrama de componentes que apresenta a organização e as dependências entre os componentes [G].

A seguir é apresentada a notação básica de um diagrama de componentes.





Notação básica do diagrama de componentes.

4.8. Modelo de Dados

4.8.1. Modelo Lógico da Base de Dados

Neste item deve ser apresentado o modelo lógico da base de dados, que pode ser o modelo entidade-relacionamento ou objeto da base de dados. No caso do modelo entidade-relacionamento o modelo lógico deve passar por todas as regras de normalização.

Como base para geração do modelo lógico pode-se utilizar o diagrama de classes. Geralmente ferramentas CASE geram automaticamente o modelo lógico da base de dados a partir do diagrama de classes.

4.8.2. Criação Física do Modelo de Dados

Neste item deve ser realizada a criação física do banco de dados, ou seja, a criação de *scripts*.

4.8.3. Dicionário de Dados

Neste item deve ser criado o dicionário de dados do banco de dados, com o obejtivo de documentar todas as tabelas, atributos, *stored procedures* [G].

4.9. Ambiente de Desenvolvimento

Neste item devem ser apresentados os softwares de desenvolvimento (linguagem de programação, banco de dados, ferramentas, etc.), equipamentos de hardware e redes que sejam essenciais para o desenvolvimento do sistema.

4.10. Sistemas e componentes externos utilizados

Neste item devem ser descritos os sistemas e componentes externos que serão utilizados no sistema. Por exemplo, sistemas que serão integrados ao sistema desenvolvido, componentes comprados ou livre que estão sendo utilizados para facilitar ou complementar o desenvolvimento do sistema.

5. Implementação

Este capítulo tem como objetivo a implementação das classes em termos de componentes, ou seja, toda a implementação deve ser realizada de acordo com as definições das fases anteriores e todos os recursos da programação orientada a objetos que a linguagem escolhida oferece.

Geralmente ferramentas CASE geram automaticamente pseudocódigos fontes (dependendo da linguagem utilizada) baseados no diagrama de classes.

Algumas boas práticas de programação devem ser seguidas para um maior entendimento do código. Algumas delas são:

- Cabeçalho de funções contendo campos como descrição, data de criação, autor, etc;
- Comentários no código;
- Padronização de nomes de variáveis, parâmetros, funções, tabelas, *stored procedures*, etc;
- Verificação de declaração das variáveis;
- Tratamento de erros;
- Criação de *stored procedures* para acesso aos dados da base;
- Utilização de padrões de projeto [G];
- Otimização do código, utilizando os “melhores” algoritmos e funções de recursividade.

6. Testes

Este capítulo tem como objetivo identificar defeitos no sistema, validar as funções do sistema, verificar se os requisitos foram implementados de forma adequada e avaliar a qualidade do software.

6.1. Plano de Testes

Neste item deve ser criado o plano de testes do sistema, permitindo a validação do sistema por parte do desenvolvedor, através da verificação dos requisitos do sistema desenvolvido. Inicialmente, identificam-se os requisitos técnicos e funcionais do sistema, e listam-se todas as situações que podem ocorrer com o sistema (essas situações podem ser elaboradas através do diagrama de caso de uso e dos diagramas de seqüência). Deve-se realizar testes de consistência de campos, funcionalidades, desempenho, etc. O Plano de Testes do Sistema deverá conter, no mínimo.

- ✔ Nº do Teste;
- ✔ Descrição do Teste;
- ✔ Resultado Esperado**.**

Por conter todos os testes do sistema, este plano poderá ser um anexo na documentação do sistema. Alguns tipos de testes a serem realizados são: teste de funcionalidades, teste de usabilidade, teste de desempenho, teste de carga, teste de *stress*, teste de volume, teste de segurança e controle de acesso, teste de tolerância a falhas e recuperação, teste de configuração, teste de instalação, etc..

Para maiores detalhes consultar o modelo de documento de plano de testes do RUP rup_tstpln.dot.

6.2. Execução do Plano de Testes

Neste item devem ser registrados os testes realizados no sistema tendo como base o Plano de Testes do Sistema.

O registro dos testes deve conter a identificação do sistema, o nome do realizador dos testes e a configuração do ambiente onde foi realizado o teste. Além disso, para cada teste, deve-se ter os seguintes dados:

- ✔ Nº do teste;
- ✔ Resultado Obtido; e
- ✔ Comentários (se necessário).

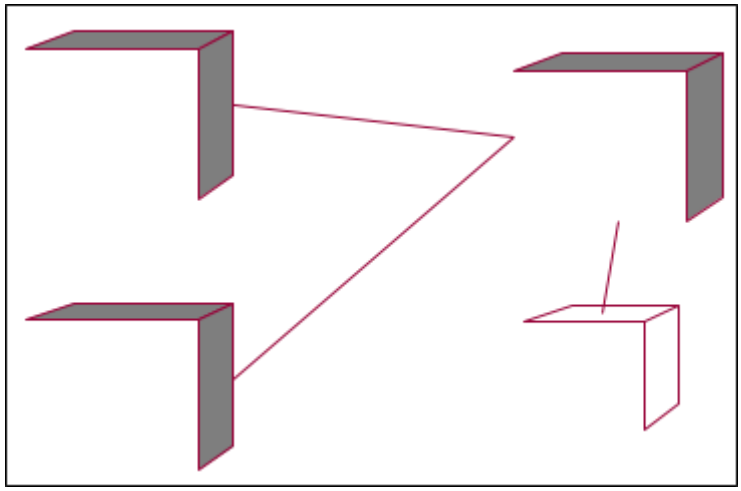
7. Implantação

Este capítulo tem como objetivo apresentar informações relevantes para a implantação e funcionamento do sistema.

7.1. Diagrama de Implantação

Neste item deve ser apresentado o diagrama de implantação que representa a parte física do sistema, exibindo os dispositivos, as máquinas de processamento em tempo de execução e os componentes que nelas serão instalados.

A seguir é apresentada a notação básica de um diagrama de implantação.



Notação básica do diagrama de implantação.

7.2. Manual de Implantação

Neste item deve ser elaborado o manual de instalação. Este manual deve conter a descrição passo a passo de como deve ser realizada a instalação do sistema.

8. Manual do Usuário

Este capítulo tem como objetivo a elaboração de um manual do usuário. Este manual deve conter a descrição passo a passo de como utilizar o sistema.

9. Conclusões e Considerações Finais

Este capítulo tem como objetivo apresentar e demonstrar a aplicabilidade dos resultados obtidos, suas limitações, inovação, possíveis integrações com outros projetos e continuação do sistema em trabalhos futuros.

Bibliografia

Neste item devem-se apresentar todas as obras (livros, artigos, Internet, revistas, etc...) utilizadas na elaboração da documentação e na implementação do projeto.

Comentários sobre a Documentação

Esta é uma parte que responde a algumas dúvidas freqüentes referente à documentação.

Comentários sobre a documentação

1. Como utilizar o produto de documentação ?

Esse documento tem como objetivo fornecer um roteiro que auxilie os alunos no desenvolvimento de software orientado a objetos, utilizando notação UML. Com isso, o roteiro deve ser algo flexível e adaptável a cada projeto.

2. Qual é o material que posso consultar caso tenha dúvidas?

- ✔ FOWLER, M.; SCOTT.K. **UML Essencial**. Editora Bookman, 3ª edição.
- ✔ BOOCH, G.; RUMBAUGH, J.; JACOBSON. I. **UML – Guia do Usuário**. Editora Campus, 2000.
- ✔ BOGGS, W.; BOGGS, M. **Mastering UML com Rational Rose 2002**. Editora Alta Books, 2002.
- ✔ QUATRANI. T. **Modelagem Visual com Rational Rose 2000 e UML**. Editora Ciência Moderna, 2001.
- ✔ PRESSMAN, R. Software Engineering. A Practitioner´s Approach. 5ª edição, 2003. McGrawHill.
- ✔ SOMERVILLE, I. **Engenharia de Software**. Addison Wesley, 6ª edição.
- ✔ YOURDON, E **Análise Estruturada Moderna**. Editora Campus, 3ª edição.
- ✔ Ferramenta *Rational Unified Process*.
- ✔ Site: http://prof.usjt.br/anapaula (disciplinas de Metodologia de Desenvolvimento de Software e Engenharia de Software)

3. Quais são as Ferramentas CASE que podem ser utilizadas?

- ✔ *Rational Suite Enterprise*
  - *Rational Rose* para modelagem de sistemas orientados a objetos;
  - *Rational Unified Process* - RUP para utilização do processo de desenvolvimento e para consulta de artefatos, definições, etc;
  - *Requisite Pro* para criação e gerenciamento de requisitos.
  - Além de outras ferramentas da Suíte...

- ✔ *System Architect* para modelagem de sistemas orientados a objetos e estruturados.
- ✔ *Microsoft Project* para elaboração de cronograma, planejamento e gerenciamento do projeto.
- ✔ *Erwin* para elaboração do modelo lógico e físico do banco de dados;
- ✔ *ArgoUML* para modelagem de sistemas orientados a objetos.
- ✔ *Dome* para modelagem de sistemas orientados a objetos.
- ✔ Outras....

Glossário

Glossário

- ✔ **Componente:** representa uma parte física da implementação de um sistema, que inclui código de software, com o objetivo de criar código de software coeso para sua reutilização e facilidade de manutenção.

- ✔ **Ferramenta CASE (*Computer Aided Software Engineering*):** é uma ferramenta que auxilia no processo de desenvolvimento de software, ajudando a garantir a qualidade do projeto e facilitando a criação de modelos, documentos.

- ✔ **Padrões de Projeto (*design patterns*):** são soluções simples para problemas específicos no projeto de software orientado a objetos. Padrões de projeto capturam soluções que foram desenvolvidas e aperfeiçoadas ao longo do tempo.

- ✔ **Recursos alocados:** pessoas que irão trabalhar no projeto.

- ✔ **Regras de negócio**: declarações e regras da política ou condição que deve ser satisfeita no âmbito do negócio.

- ✔ **Requisito:** um requisito descreve uma condição ou capacidade à qual um sistema deve se adaptar, sejam necessidades dos usuários, um padrão ou uma especificação.

- ✔ ***Stored Procedures*:** é uma rotina escrita através de comandos SQL, que tem como objetivo encapsular o processo de negócio e sua reutilização. As *stored procedures* ficam armazenadas no gerenciador de banco de dados.

- ✔ **Usabilidade:** é a qualidade da interface homem-máquina, que permite que o usuário realize com eficiência e conforto as atividades a que o sistema se destina.