Documentação de um Produto de Software

Versão 2010

Autores:

Profª. Dra. Ana Paula Gonçalves Serra

Prof. Msc. André luiz Dias Ribeiro (PMP)

Profª. Dra. Milkes Yone Alvarenga

2010

UNIVERSIDADE SÃO JUDAS TADEU
FACULDADE DE TECNOLOGIA E CIÊNCIAS EXATAS

SISTEMAS DE INFORMAÇÃO

Trabalho de Graduação

TÍTULO

SÃO PAULO

2010

UNIVERSIDADE SÃO JUDAS TADEU
FACULDADE DE TECNOLOGIA E CIÊNCIAS EXATAS

SISTEMAS DE INFORMAÇÃO

NOME_1
nome_2
nome_3

TRABALHO DE GRADUAÇÃO

TÍTULO

Trabalho de Graduação apresentado à banca examinadora do Curso de Sistemas de Informação da Faculdade de Tecnologia e Ciências Exatas da Universidade São Judas Tadeu, como exigência parcial para obtenção do grau de Bacharel em Sistemas de Informação sob orientação do Prof. Dr. Márcio de Lima.

São Paulo

2010

Avaliação do trabalho de graduação

Título	TÍTULO DO TRABALHO		
Curso	SITEMAS DE INFORMAÇÃO/CIÊNCIA DA COM...	Turma	4ASIN
Área			
Alunos	ALUNO 1 (NOME COMPLETO)		
	ALUNO 2 (NOME COMPLETO)		
	ALUNO 3 (NOME COMPLETO)		
	ALUNO 4 (NOME COMPLETO)		
	ALUNO 5 (NOME COMPLETO)		

Banca examinadora

Professores	A	B	C	D	E	F	Total	Visto
NOME PROF 1								
NOME PROF 2								
NOME PROF 3								
Média								

A - Qualidade e clareza da apresentação ..... xx pontos
B - Nível de conhecimento do assunto ..... xx pontos
C - Cumprimento dos objetivos finais..... xx pontos
D - Qualidade técnica do projeto ..... xx ponto
E - Implementação e testes ..... xx pontos
F - Documentação o apresentada..... xx pontos

Ofereço esse trabalho a meus pais que sempre se esforçaram pelo meu sucesso.

AGRADECIMENTOS

Ao Prof. Dr. Márcio de Lima pela orientação deste trabalho....

Ao Prof. Ms. Jair de Souza pelas sugestões e críticas...

.....

Aos nossos pais pelo interesse com que acompanharam...

Aos professores do curso que de alguma forma contribuíram...

ÍNDICE

1. INTRODUÇÃO AO DOCUMENTO 9

1.1. TEMA 9
1.2. OBJETIVO DO PROJETO 9
1.3. DELIMITAÇÃO DO PROBLEMA 9
1.4. JUSTIFICATIVA DA ESCOLHA DO TEMA 9
1.5. MÉTODO DE TRABALHO 9
1.6. ORGANIZAÇÃO DO TRABALHO 10
1.7. GLOSSÁRIO 10

2. REQUISITOS [G] DO SISTEMA 11

2.1. REQUISITOS FUNCIONAIS 11
2.2. REQUISITOS NÃO-FUNCIONAIS 12
2.3. PROTÓTIPO 12
2.4. MÉTRICAS E CRONOGRAMA 13
2.5. CRONOGRAMA 13

3. ANÁLISE 14

3.1. DIAGRAMA DE CLASSES DE ANÁLISE (MODELO DO DOMÍNIO) 14
3.2. DIAGRAMAS DE INTERAÇÃO 14

4. PROJETO 16

4.1. ARQUITETURA DO SISTEMA 16
4.2. DIAGRAMA DE CLASSES DE PROJETO POR CASO DE USO 16
4.3. DIAGRAMA DE COMPONENTES 18
4.4. MODELO DE DADOS 18
4.4.1. Modelo Lógico da Base de Dados 18
4.4.2. Dicionário de Dados 18

5. TESTES 19

5.1. PLANO DE TESTES 19
5.2. EXECUÇÃO DO PLANO DE TESTES 19

6. IMPLANTAÇÃO 20

6.1. MANUAL DE IMPLANTAÇÃO 20

7. MANUAL DO USUÁRIO 21

8. CONCLUSÕES E CONSIDERAÇÕES FINAIS 22

BIBLIOGRAFIA 23

PERGUNTAS FREQUNTES - COMENTÁRIOS SOBRE A DOCUMENTAÇÃO ... 24

GLOSSÁRIO 26

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 da Universidade São Judas Tadeu dos cursos de Ciência da Computação e Sistemas de Informação, apoiando o Trabalho de Graduação (TG).

Esta é a versão 2009 do documento, totalmente revisada para utilizar a notação UML e modelos do RUP (*Rational Unified Process* – Processo Unificado *Rational*). Neste documento são citados alguns modelos do RUP que podem ser utilizados e consultados na ferramenta *Rational Unified Process* (que faz parte da ferramenta *Rational Suite Enterprise*) ou o site da IBM. A escolha da orientação a objetos é devido à tendência de mercado, mas nada impede que o roteiro seja seguido no caso de opção pela Modelagem Estruturada (ver item Comentários sobre a Documentação).

No final deste documento há um glossário, os termos que constam no glossário são representados no documento pela letra [G] em azul.

Sugestões e Comentários podem ser enviados para prof.andreluiz@usjt.br.

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

Neste item deve-se apresentar o tema do projeto, de forma clara e objetiva.

1.2. Objetivo do Projeto

Neste item devem ser descritos os objetos gerais e específicos do projeto como um todo. Independente do que será implementado, este item visa o entendimento global do projeto.

1.3. Delimitação do Problema

Neste item deve ser descrita a delimitação do problema, que define o ponto central do projeto. Isso quer dizer que, dentro de uma idéia geral do projeto, deve-se ressaltar a idéia específica efetivamente a ser desenvolvida. É neste item que a amplitude do projeto tem sua delimitação perfeitamente definida.

1.4. Justificativa da Escolha do Tema

Neste item deve-se expor a motivação acadêmica para a elaboração do projeto em questão, detalhando os motivos de ordem teórica ou de ordem prática para a sua realização.

1.5. Método de Trabalho

Neste item deve-se descrever o método a ser utilizado para realização do projeto, o tipo de processo de desenvolvimento de software[1], a modelagem a ser utilizada (orientada a objeto, estruturada, outras).

[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

Neste item deve-se descrever como o documento estará organizado.

1.7. Glossário

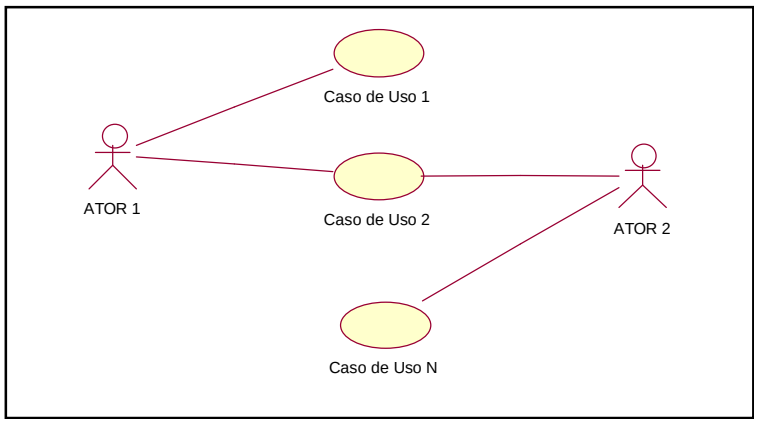
Neste item deve-se definir os termos importantes utilizados no projeto, facilitando o seu entendimento. Caso exista um número extenso de termos no projeto consultar e utilizar o modelo rup_gloss.dot – artefato do RUP.

2. Requisitos[G] do Sistema

Este capítulo tem como objetivo descrever os requisitos do sistema. No caso de sistemas que possuam usuários / solicitantes reais para o levantamento de requisitos, pode-se utilizar o modelo de documento de entrevista com usuários do RUP de Solicitações dos Principais Envolvidos (rup_stkreq.dot).

2.1. Requisitos Funcionais

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.



Notação básica do diagrama de caso de uso.

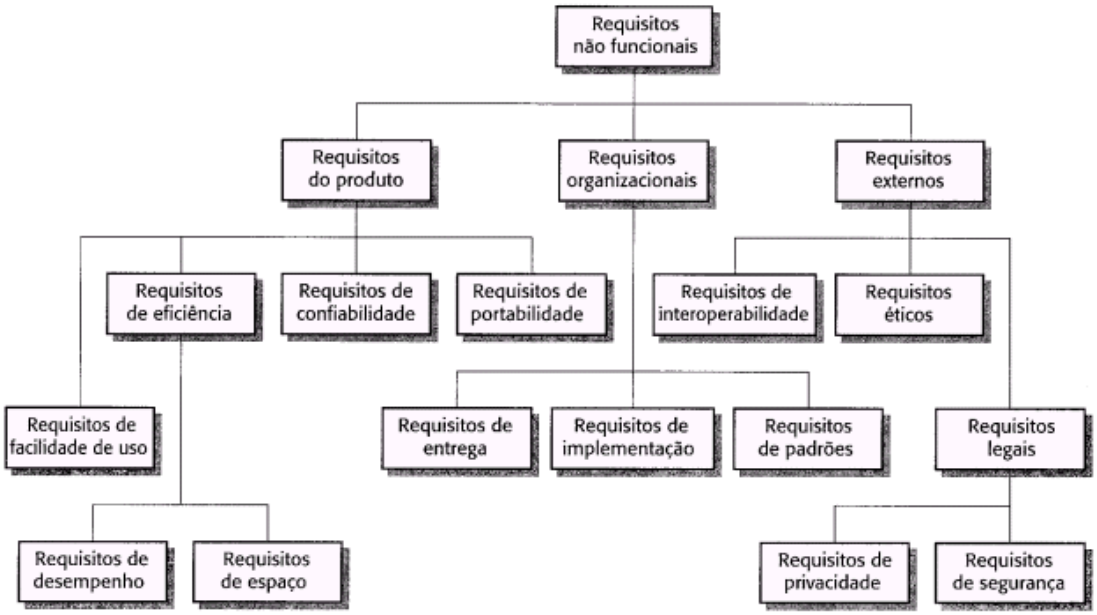
Cada ator do diagrama de caso de uso deve ser descrito de forma sucinta (2 linhas) e cada caso de uso deve ser especificado. A seguir são apresentados itens básicos para a especificação dos casos de uso do diagrama.

- ✓ Nome do Caso de Uso
- ✓ Breve descrição
- ✓ Atores envolvidos
- ✓ Fluxo Básico
- ✓ Fluxos Alternativos
- ✓ Pré-condições
- ✓ Pós-condições;

Para maiores detalhes de especificação de casos de uso consultar e utilizar o modelo rup_ucspec.dot – artefato do RUP.

2.2. Requisitos Não-Funcionais

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. Para maiores detalhes de requisitos não-funcionais consultar e utilizar o modelo de documento rup_ucspec.dot – artefato do RUP.



Sommerville, Ian. Engenharia de Software, 6ª edição.

2.3. 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 podem 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.

2.4. Métricas e Cronograma

Neste item devem ser estimados os esforços necessários em termos de recursos alocados [G] 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.

2.5. Cronograma

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 sequência 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.

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

3. Análise

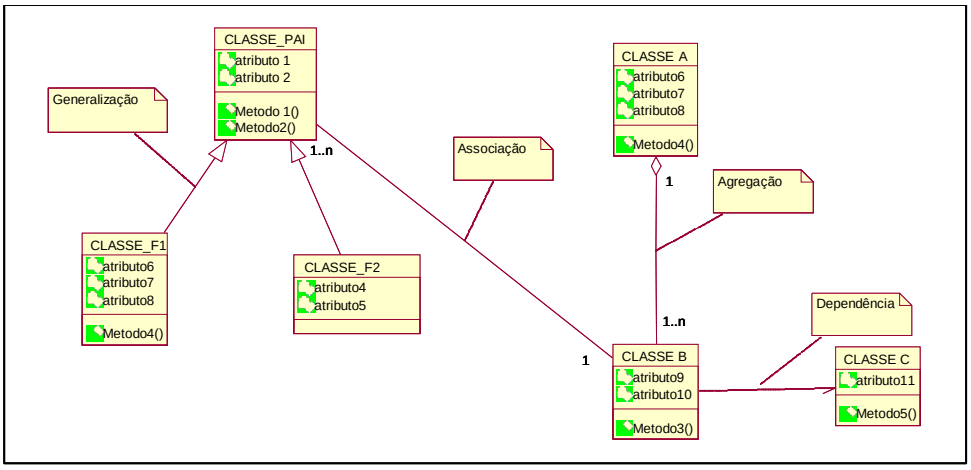
Este capítulo tem como objetivo analisar, detalhar e propor uma solução geral do sistema de acordo com os requisitos levantados e validados no capítulo 2.

3.1. Diagrama de Classes de Análise (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, refinado e complementado para compor o diagrama de classes de projeto.

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

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



Notação básica do diagrama de classes.

3.2. Diagramas de Interação

O diagrama de interação é composto pelos diagramas de sequência 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.

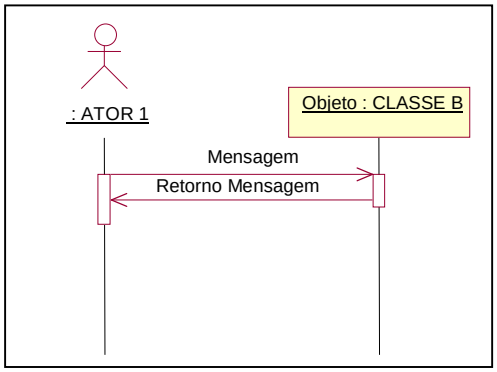
3.3.1. Diagrama de Sequência

Neste item devem ser apresentados os diagramas de sequência essenciais ao sistema, identificados através dos casos de uso, desde que o caso de uso possua mais de uma classe envolvida.

Um diagrama de sequência representa interações de objetos organizadas em uma sequência temporal, apresentando os objetos que participam da interação e a sequência das mensagens trocadas. O diagrama de sequência deve validar o diagrama de classes e vice-versa.

Obs: Para os casos de uso que representam <<crud>> básico não há necessidade de realização do caso de uso

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



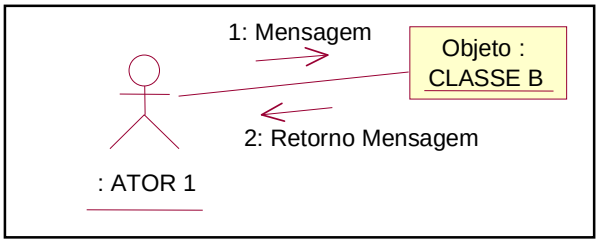
Notação básica do diagrama de sequência.

3.3.2. Diagrama de Colaboração / Comunicação

Esse diagrama é uma alternativa para o diagrama de sequência (item 3.3.1). 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 CASEs geram automaticamente o diagrama de colaboração/comunicação a partir do diagrama de sequência.

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



Notação básica do diagrama de colaboração.

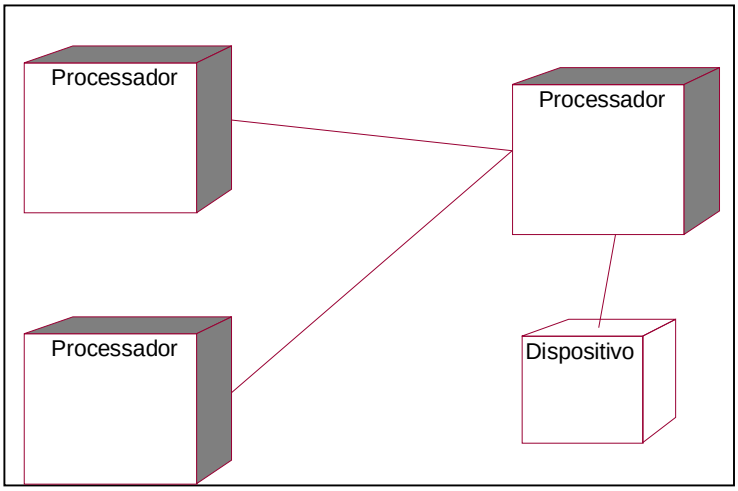
4. Projeto

Este capítulo tem como objetivo refinar a proposta de solução geral do sistema de acordo com a arquitetura e tecnologia utilizada.

4.1. Arquitetura do Sistema

Neste item deve ser apresentada a arquitetura de infraestrutura do sistema, demonstrando o tipo de arquitetura que será utilizada (por exemplo, cliente/servidor de n-camadas, MVC, ...), a configuração de hardware, de rede, de software, padrões de projeto, componentes específicos (dll, jar, ...) e componentes externos a serem utilizados, bem como o dimensionamento mínimo de conexões.

Para a representação da arquitetura de infraestrutura pode-se utilizar uma figura ilustrativa ou o diagrama de distribuição. A seguir é apresentada a notação básica de um diagrama de distribuição.



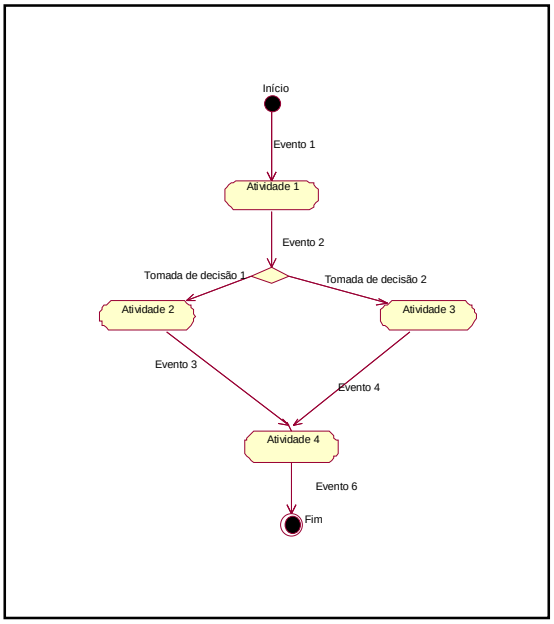
Notação básica do diagrama de distribuição.

4.2. Diagrama de Classes de Projeto por Caso de Uso

Este item tem como objetivo apresentar a realização de caso de uso para cada caso de uso, desde que o caso de uso possua mais de uma classe envolvida. Este item deve conter:

- ✓ diagrama de classes para cada caso de uso. Já descrito no item 3.1;
- ✓ diagrama de seqüência (para cada cenário do fluxo básico e fluxos alternativos). Já descrito no item 3.3.1;
- ✓ diagrama de atividades (para os métodos que contém regras de negócio).

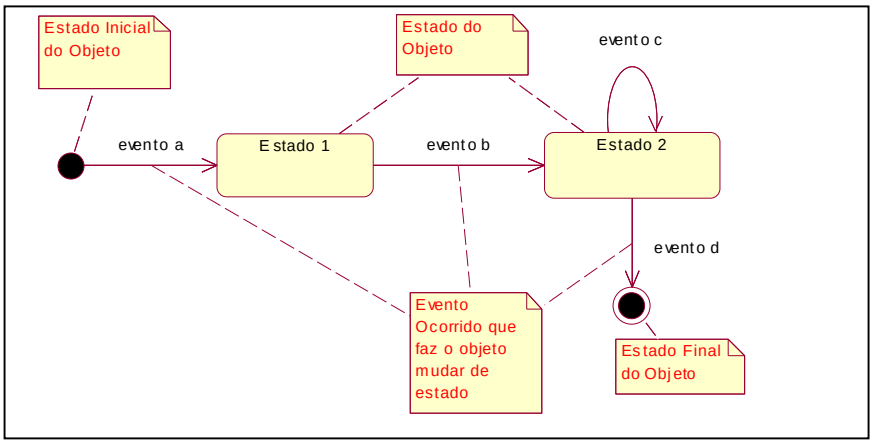
  O diagrama de atividades representa o detalhamento de tarefas e o fluxo de uma atividade para outra de um sistema. A seguir é apresentada a notação básica de um diagrama de atividades.



Notação básica do diagrama de atividades.

- ✓ diagrama de estados (se necessário) para casos de uso que possuem mais de uma classe envolvida.

  O diagrama de estados especifica as sequências de estados pelas quais o objeto pode passar durante seu ciclo de vida em resposta a eventos. A seguir é apresentada a notação básica de um diagrama de estados.



Notação básica do diagrama de estados

Obs: Para os casos de uso que representam <<crud>> básico não há necessidade de realização do caso de uso.

.

4.3. Diagrama de Componentes

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

4.4. Modelo de Dados

4.4.1. 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.

4.4.2. Dicionário de Dados

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

Tabela					
Campo	**Tipo**	**Chave Primária**	**Chave Estrangeira**	**Null**	**Observações**

Estrutura Básico do Dicionário de Dados

Tb_Cliente					
Campo	**Tipo**	**Chave Primária**	**Chave Estrangeira**	**Null**	**Observações**
Codcliente	Int	X			
Email	Caracter(30)	X		X	

Exemplo de Dicionário de Dados

5. Testes

Este capítulo tem como objetivo identificar defeitos no sistema, validar as funções do sistema, verificar se os requisitos foram implementados de forma adequada.
Para maiores detalhes pode-se consultar artefatos do RUP da fase de Testes.

5.1. 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. Para maiores detalhes ver o modelo de Plano de Testes - **Plano_de_Testes.doc.**

Por conter todos os testes do sistema, este plano poderá ser um anexo na documentação do sistema.

5.2. Execução do Plano de Testes

Neste item devem ser registrados os testes realizados no sistema tendo como base o Plano de Testes do Sistema. Para maiores detalhes ver o Roteiro de Testes - **Exemplo_Roteiro_Testes.xls.**

6. Implantação

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

6.1. 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.

Para maiores detalhes ver modelo de Plano de Implantação - **Modelo Plano Implantação.doc.**

7. 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.
Para maiores detalhes ver exemplo de manual do usuário - **Exemplo_Manual_Usuário.pdf.**

8. 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.

Perguntas Frequntes - Comentários sobre a documentação

Como utilizar o produto de documentação no TG?

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. Cada grupo deve analisar e decidir em conjunto com o seu professor orientador os itens do documento que se aplicam para o projeto escolhido e se não há necessidade de criar ou substituir itens, informações e diagramas complementares ao roteiro.

Além da aplicabilidade dos itens do documento, cada grupo deve decidir com o seu orientador o processo a ser utilizado ao longo do desenvolvimento, as etapas que devem ser cumpridas em cada reunião de orientação e o tipo de modelagem e implementação a ser utilizada. O desenvolvimento orientado a objetos é sugerido pela tendência de mercado, caso a escolha seja por um desenvolvimento estruturado ver item 2 desta seção.

No meu TG irei utilizar Modelagem Estruturada, qual é a diferença?

A diferença básica está nos diagramas e na implementação. A seguir são apresentados os itens que podem ser substituídos no roteiro e qual seria um item correspondente para a modelagem estruturada.

- ✓ **Item 2.1. Requisitos Funcionais:** apesar do diagrama de caso de uso ser da modelagem orientada a objetos, este pode ser utilizado na modelagem estruturada para identificar os usuários, os sistemas externos e as funções do sistema. Caso o diagrama de caso de uso não seja utilizado deve-se realizar uma descrição textual e detalhada de cada requisito;
- ✓ **Item 3.1 e 4.2. Diagrama de Classes:** deve-se substituir o diagrama de classes pelo diagrama de fluxo de dados (DFD) – nível 0 (diagrama de contexto), nível 1, nível 2,...nível n se necessário.
- ✓ **Item 3.2. Diagrama de Interação:** devem-se substituir os diagramas de interação pelo dicionário de dados dos diagramas de fluxos de dados;
- ✓ **Item 4.2. Diagrama de Atividades:** apesar do diagrama de atividades ser da modelagem orientada a objetos, este pode ser utilizado na modelagem estruturada para o detalhamento de funções complexas do sistema. Isso também pode ser realizado através de fluxogramas, diagramas de Nassi-Schneiderman, portugol, etc;
- ✓ **Item 5.1.. Plano de Teste:** o plano de teste deve seguir o item 6.1, mas as situações de teste podem ser extraídas dos diagramas de fluxo de dados (DFD).
- ✓ **Item 4.1. Diagrama de Distribuição:** apesar do diagrama de distribuição ser da modelagem orientada a objetos, este pode ser utilizado na modelagem estruturada para representar a parte física do sistema.

Para maiores detalhes sobre Modelagem Estruturada consultar o livro: Análise Estruturada Moderna. Edward Yourdon. Editora Campus, 3ª edição.

No meu TG haverá um estudo teórico além do sistema, qual é a diferença?

A principal diferença é criação de um capítulo intermediário entre o Capítulo 1 e 2, que deve apresentar todo o estudo teórico da pesquisa. Esse capítulo deve conter basicamente os itens a seguir, mas cabe ao professor orientador definir os itens a serem criados no estudo teórico.

- ✓ Apresentação do Trabalho Teórico;
- ✓ Justificativa do Trabalho Teórico e a relação com o projeto a ser desenvolvido;
- ✓ Dissertação do Trabalho Teórico;
- ✓ Análise dos resultados/Conclusão do Trabalho Teórico.

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)

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

- ✓ **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.

- ✓ **<<crud>>:** estereótipo estendido da UML que representa CREATE, READ, UPDATE e DELETE.

- ✓ **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.

- ✓ **MVC:** padrão de projeto de arquitetura que representa MODEL-VIEW-CONTROL.

- ✓ **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.