INSTITUTO FEDERAL
CEARÁ
Campus Aracati

{Introdução}

Análise e Projetos de Sistemas

Prof. Fábio José.
Prof.fabiojose@gmail.com
Fabio.jose@ifce.edu.br

Processo de Desenvolvimento de Software

- Atividade complexa;
- Envolve:
  - Tratamento das complexidades dos componentes de desenvolvimento de um software (hardware, software, peopleware, procedimentos, etc);

Processo de Desenvolvimento de Software

- A partir do ano de 1994, o *Standish Group International* publica (a cada 2 anos), um estudo intitulado de *Chaos Research;*
- Objetivo: promover a demonstração dos resultados obtidos nos projetos de software que foram executados durante o período de tempo estudado;

Processo de Desenvolvimento de Software

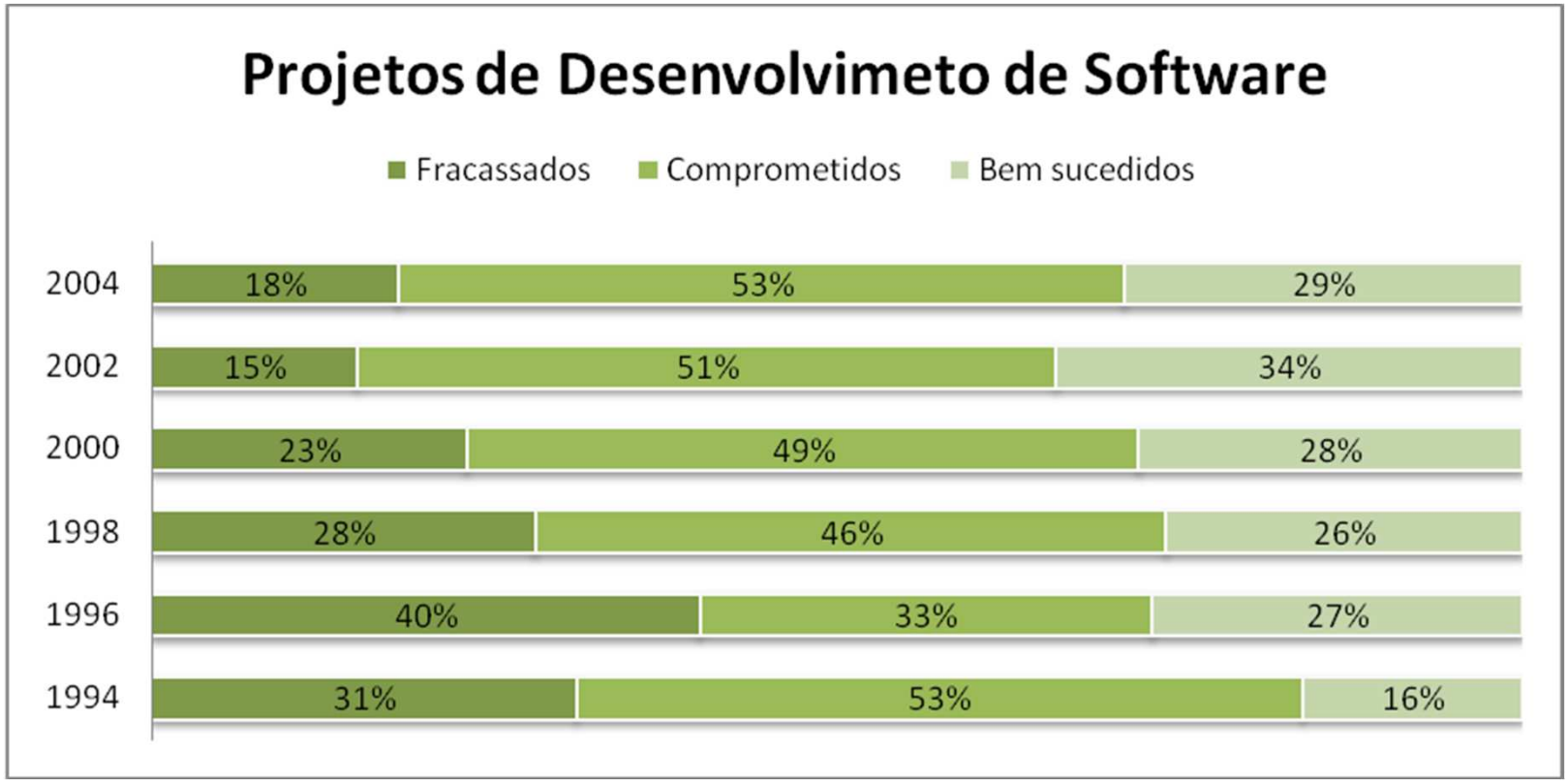


Figura 1 - Resultado dos estudos do Chaos Research
Fonte: Chaos Research 1994-2004

/FabioJose

Processo de Desenvolvimento de Software

- Conclusão:
  - Necessidade de definir processos de desenvolvimento de software!

Processo de Desenvolvimento de Software

- Compreende todas as atividades necessárias para definir, desenvolver, testar e manter um produto de software.
- Objetivos:
  - Definir quais atividades devem ser executadas ao longo de um projeto;
  - Quando, como e por quem será executadas;
  - Promover pontos de controles para acompanhar o andamento do desenvolvimento;
  - Padronizar a forma de desenvolvimento de software em uma organização;

Processo de Desenvolvimento de Software

- Classifica em atividades as tarefas realizadas durante a construção de um sistema de software.

  Ex: RUP, XP, MSF, etc.

- Não existe o melhor processo de desenvolvimento, mas o que se aplica melhor para um determinado cenário.
- Cada processo determina o modo como arranjar e encadear as atividades de desenvolvimento.

Processo de Desenvolvimento de Software

- Atividades típicas de um Processo de Desenvolvimento:
  - Levantamento de Requisitos
  - Análise de Sistemas
  - Projeto de Sistemas
  - Implementação
  - Testes
  - Implantação

Levantamento de Requisitos

- Fase da etapa do ciclo de vida cuja atividade é a compreensão do problema aplicado a um contexto para o desenvolvimento de um sistema.
- Permitir que usuários e desenvolvedores tenham a mesma visão;
- Busca levantar e definir as necessidades dos futuros usuários do sistema a ser desenvolvido;

Levantamento de Requisitos

- Necessidades dos usuários = Requisitos;
- São identificados a partir de um domínio.
- Domínio = Área do conhecimento ou atividade específica
- Caracteristicas:
  - Conjunto de conceitos e terminologias compreendidas por especialistas da área;

Análise de Requisitos

- Fundamental para o desenvolvimento de sistemas.
- Descobrir o que o cliente quer com o sistema.
- Processo de descobrir que operações o sistema deve realizar e quais as restrições que existem sobre estas operações.

Tipos de Requisitos

- Funcionais
  – o que o sistema deve fazer
  Ex: Registrar a locação de um veículo,
  Emitir nota fiscal de serviço.

- Não-funcionais
  – restrições sobre como o sistema deve desempenhar suas funções
  Ex: Tempo de reposta do sistema deve ser inferior a 10 segundos,
  Utilizará o S.Operacional Windows 2000.

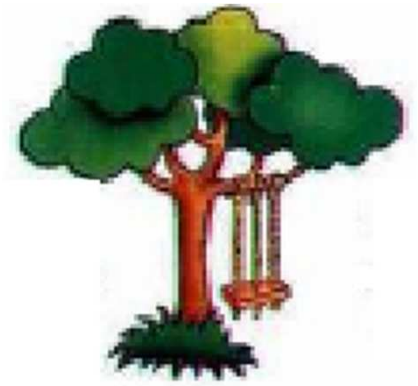
Erro no Entendimento Correto dos Requisitos

• O analista determina que requisitos são coisas que ele *planejou* , e não o que o cliente ou usuário *solicitam*.

/FabioJose

Erro no Gerenciamento de Requisitos

Como o sistema foi descrito no levantamento inicial



Como foi definido pelo Analista

/FabioJose

Erro no Gerenciamento de Requisitos

Como foi especificado no Projeto

Como foi Implementado

Erro no Gerenciamento de Requisitos

Como foi corrigido pela Manutenção

O que o Usuário queria

Técnicas de Levantamento de Requisitos

- **Entrevistas:**
- - Planeje a entrevista
- - Conduza a entrevista
- - Escreva o relatório
- Método interessante para coletar informações
- Revela informações sobre:
  - As opiniões do entrevistado
  - O estado do sistema atual (se existir)
  - Objetivos pessoais e organizacionais
  - Procedimentos informais

– Ouvir é essencial!

Planejamento da Entrevista

- Ler o material de suporte
- Estabelecer os objetivos da entrevista
- Decidir quem entrevista
- Preparar o entrevistado
- Decidir os tipos de questões e sua estrutura

Tipos de Questões

- **Questões Abertas**

- Não existe uma agenda pré-definida

- **Questões Fechadas**

- Limita o número de possíveis respostas

- Gera dados precisos

Questões Abertas

•**Benefícios**

-Entrevistado fica mais a vontade

-Usar o vocabulário do entrevistado

-Maior detalhe

-Gera novas questões

-Mais interessante para o entrevistado

-Mais expontaneidade

-Exige pouca preparação

•**Desvantagens**

- Pode-se obter muitos detalhes sem importância

-É possível perder o controle da entrevista

-As respostas podem demorar demasiado tempo

-Parecer pouco preparado

Questões Fechadas

•**Benefícios**

-Poupam Tempo

-É fácil comparar entrevistas

-Vai direto ao que é importante

-Mantém o controle sobre a entrevista

-É possível cobrir muitos assuntos

-Focado apenas nos dados relevantes

•**Desvantagens**

-Penoso para o entrevistado

-Falhas de detalhes importantes

-Pode não abordar as idéias essênciais

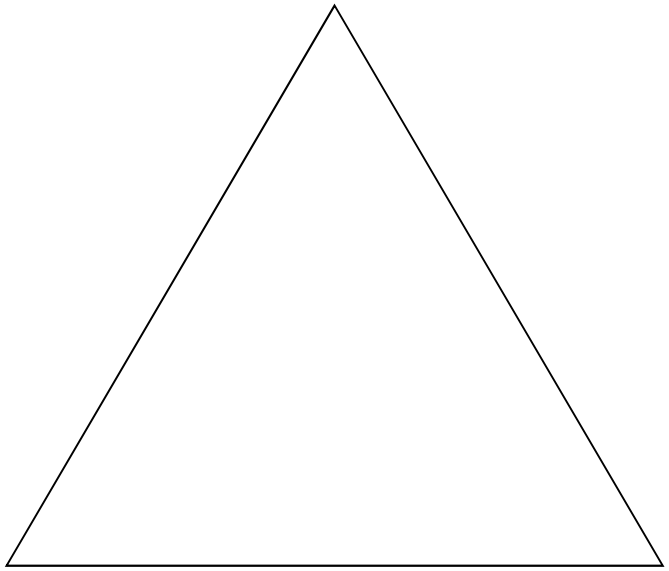
-Não cria empatia com o entrevistado

Estruturas de Entrevistas

- **Pirâmide**

- Iniciar com questões específicas, fechar com questões genéricas

- Usar com entrevistados relutantes



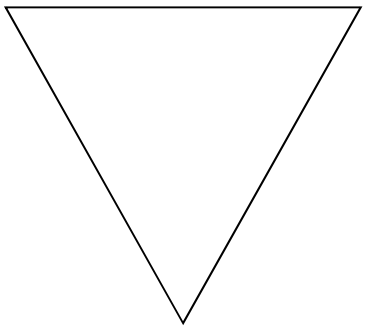
Estruturas de Entrevistas

- **Funíl**

- Iniciar com questões genéricas, fechar com uma questão específica

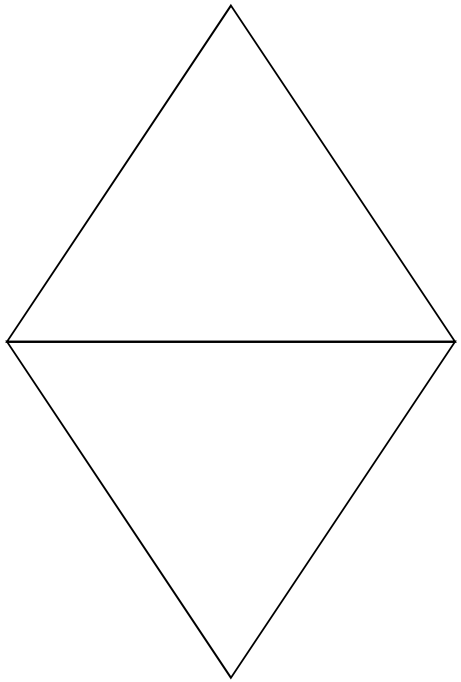
- Forma amigável de começar a entrevista

- Usar quando os entrevistados tem uma relação efetiva com o assunto



Estruturas de Entrevistas

- **Diamante**
  - Combina as aproximações anteriores,
  - Mais demorado
  - Mantém o entrevistado interessado usando questões variadas



Registrar a Entrevista

- Podem ser registradas num gravador digital ou num bloco de notas

- O registro em gravador deve ser feito com permissão e conhecimento do entrevistado

Conduzir a Entrevista

- **Antes da Entrevista**
  - Confirmar o encontro
  - Vestir-se de forma apropriada
  - Chegar cedo
- **Início da Entrevista**
  - Cumprimentar o entrevistado
  - Relembrar o objetivo da entrevista
  - Informar a forma de registro da entrevista
- **Perguntas Iniciais**
  - Começar com conversa agradável, perguntas abertas
  - Ouvir atentamente as respostas
  - Procurar metáforas

Conduzir a Entrevista

- **Durante a Entrevista**
  - Não deve exeder 1 hora
  - Ter certeza de perceber o que está sendo dito
  - Pedir definições
  - Ser sensível

Conduzir a Entrevista

- **Encerrar a Entrevista**
  - Refletir sobre o entrevistado
  - Existe algo que não foi referido, mas que o entrevistado deseja adicionar
  - Fazer um sumário
  - Quem deverá ser entrevistado a seguir
  - Marcar próxima reunião
  - Agradecer pelo tempo dispendido

Relatório da Entrevista

- Escrever o mais cedo possível a seguir à reunião
- Incluir um breve sumário inicial e uma descrição mais detalhada
- Rever o relatório com o entrevistado

Documentar Requisitos

Finalidade

Identificar, detalhar e documentar o conjunto de requisitos funcionais.

Freqüência

Os requisitos funcionais devem ser elaborados na fase de Iniciação e podem ser refinados de modo incremental durante as fases de Elaboração e Construção. Devem ser revistos e/ou alterados quando ocorrer uma mudança de requisitos.

Documentar Requisitos

Passos

1. Identificar os requisitos funcionais
2. Descrever os requisitos funcionais
3. Documentar os requisitos funcionais
4. Obter aprovação dos requisitos

Documentar Requisitos

1. Identificar Os Requisitos Funcionais

Os requisitos devem ser extraídos dos artefatos de proposta e levantamento com o usuário. Um requisito é definido como uma condição ou uma capacidade com a qual o sistema deve estar de acordo. Os requisitos funcionais especificam ações que um sistema deve ser capaz de executar, sem levar em consideração restrições físicas. Os requisitos funcionais especificam, portanto, o comportamento de entrada e saída de um sistema.

Documentar Requisitos

2. Descrever os requisitos funcionais

É importante que os requisitos funcionais tenham além de seu identificador uma descrição detalhada que mencione as entradas e saídas (“O sistema emite notas fiscais a partir de pedidos de venda aprovada, gerando log, baixa de estoque e emissão de cupom fiscal”).

Documentar Requisitos

3. Documentar os requisitos funcionais

Os requisitos funcionais podem ser documentados através do documento de Especificação de Requisitos Funcionais ou então é inserido no repositório de requisitos (por exemplo: Rational RequisitePro). Para obter maiores informações onde como e onde devem estar documentados os requisitos funcionais, consulte Procedimento – Elaborar Plano de Gestão de Requisitos.

Documentar Requisitos

4. Obter aprovação dos requisitos

Após documentar os requisitos funcionais, estes devem ser aprovados pelo cliente. Após esta aprovação estes requisitos devem ser congelados em baseline. Os requisitos estando documentados e aprovados serão utilizados pela equipe do projeto como base para os planos, produtos e atividades do projeto.

Análise de Sistemas

- Fase da etapa do ciclo de vida cuja atividade é a construção de modelos do sistema.
  - Trata dos aspectos lógicos e independentes de implementação.
- Quebra um sistema em componentes;
- Estudar como esses componentes interagem;
- Entender como esse sistema funciona.

Análise de Sistemas

- Estudo detalhado dos requisitos levantados;
- Permitir a construção de modelos (Plantas) para representar o sistema a ser construído;
- Não se preocupa em “como” será realizado sua implementação;
  - Obter a melhor solução sem se preocupar com os detalhes tecnológicos.
  - Foco no “o que” o sistema deve fazer e depois definir “como”fazê-lo.

Análise de Sistemas

- Não são considerados nessa fase:
  - Ambiente tecnológico;
- “Paralisia da Análise”:
  - Analistas passam muito tempo construíndo modelos;
  - Na Prática: Desenvolvedores construíndo o sistema sem antes terem definido completamente o problema;
  - Os modelos devem ser validados e verificados na fase de análise.

Análise de Sistemas

- Validação:
  - Assegurar que os requisitos serão atendidos;
  - Permite aos analistas assegurar que a especificação está correta.
  - Os modelos são validados junto aos usuários;
  - “Será que o software correto está sendo construído?”

Análise de Sistemas

- Usuário validou!
  - Entendeu o modelo;
  - O modelo reflete suas necessidades.

- Usuário NÃO validou!
  - Não entendeu o modelo;
  - Existe diferentes interpretações (ambiguidades e contradições) entre analistas e usuários;
  - Pode gerar grande impacto em prazos e custos para um projeto de software se identificado em etapas posteriores.

Análise de Sistemas

- Fator de sucesso de um sistema:
  - Envolvimento de especialistas do domínio do problema durante o desenvolvimento;
  - Validação do modelo pelos clientes.

Análise de Sistemas

- Verificação:
  - Analisar se os modelos construídos estão em conformidade com os requisitos definidos;
  - “O software está sendo construído corretamente?”

Análise de Sistemas

- O que é verificado nos modelos?
  - Analise da exatidão de cada modelo;
  - A consistência com os demais modelos;
  - Atividade típica da fase de projeto.

Análise de Sistemas

- Resultados:
  - Um modelo de classes e objetos do sistema;
  - Um modelo funcional do sistema a ser desenvolvido.

Análise de Sistemas

- Subfases:
  - Análise de domínio (do negócio);
  - Análise da aplicação (dos componentes do sistema);

Análise de Sistemas

- Análise de domínio (do negócio);
  - Identificar e modelar os objetos do mundo real que são processados pela aplicação;

  Ex: O sistema: Controle Acadêmico;

  Objeto de domínio: O aluno;

    - Os objetos de domínio fazem sentido aos especialistas de domínio;
    - Correspondem a conceitos do dia-a-dia do trabalho desses profissionais;
    - Os analistas de domínio devem interagir com os especialistas do domínio.

Análise de Sistemas

- Análise de domínio (do negócio);
  - Identificar as regras de negócio e os processos do negócio realizados na organização;

  Ex: O sistema: Controle Acadêmico;

  Regras do domínio:

  - Matricula acadêmica de um aluno;
  - Matrícula financeira de um aluno;

Análise de Sistemas

- Análise de domínio também conhecida como modelagem de negócio ou modelagem de processos do negócio;
- Normalmente é seguida pela Análise da Aplicação.

Análise de Sistemas

- Análise da Aplicação (do sistema);
  - Identificar os objetos de análise que :
    - não fazem sentido para os especialistas de domínio ;
    - Necessário para suprir necessidades funcionais do sistema;
  - Objetos referente a aspectos computacionais de alto nível;

    Ex: Tela de inscrição em disciplinas;

  - Qualquer objetos que seja necessário para que o sistema possa se comunicar com o seu ambiente;

Análise de Sistemas

- Análise da Aplicação (do sistema);
  - Objetos da aplicação só tem sentido no contexto de um sistema de software;
  - Só identifica os objetos, não como serão implementados.

Análise de Sistemas

- Por que dividir a Análise em subfases?
  - Reuso;
    - Criação de componentes com potencialidade para reuso dentro do mesmo domínio.

Análise de Sistemas

- Ferramentas:
  - UML (*Unified Modeling Language ou Linguagem de Modelagem Unificada*);
  - Principais diagramas da Análise:
    - Diagrama de casos de uso
    - Diagrama de Classes
  - Demais diagramas da Análise:
    - Diagrama de interação
    - Diagrama de estado
    - Diagrama de Atividades

Projeto de Sistemas

- Fase da etapa do ciclo de vida cuja atividade é o planejamento do uso da tecnologia para construír o sistema.
- Se preocupa em “como” o sistema deverá funcionar para atender aos requisitos;
  - Considera detalhes tecnológicos (aspectos físicos e tecnológicos).
  - Foco no “como” o sistema deve fazer.

Projeto de Sistemas

- Aspectos considerados:
  - Arquitetura do sistema;
  - Padrão da GUI (*Graphic User Interface*);
  - Linguagem de programação;
  - SGBD – Sistema Gerenciados de Banco de Dados;
  - Etc.

Projeto de Sistemas

- Produz a descrição computacional do "como" o software deve fazer para ser coerente com a descrição da análise.
- É direcionada pelos modelos elaborados na fase de análise;

Projeto de Sistemas

- Atividades principais do projeto:
  - Projeto da arquitetura (projeto de alto nível);
  - Projeto detalhado(projeto de baixo nível).

Projeto de Sistemas

- Projeto da arquitetura :
  - Distribuir as classes de objetos em subsistemas e seus componentes;
  - Distribuir os componentes fisicamente pelos recursos de hardware disponíveis.
  - Principais diagramas:
    - Diagrama de Implementação
  - Realizado pelo arquiteto de software;

Projeto de Sistemas

- Projeto Detalhado :
  - Modelagem das colaborações entre objetos de cada módulo;
  - Objetiva realizar as funcionalidades do módulo;
  - Realização dos projetos da GUI;
  - Realização dos projetos de banco de dados;
  - Considerados aspectos de concorrência e distribuição do sistema;
  - Mapeamento dos modelos de análise para artefatos de software;
  - Projeto dos algoritmos a serem utilizados no sistema;

Projeto de Sistemas

- Projeto Detalhado :
  - Principais diagramas utilizados:
    - Diagrama de Casos de uso
    - Diagrama de Classes
    - Diagrama de Interação
    - Diagrama de Estados
    - Diagrama de Atividades

Projeto de Sistemas

- As atividades de análise e projeto são descritas separadamente, mas durante o desenvolvimento elas se misturam.

Implementação

- Realiza a atividade de codificação do projeto do sistema;
- Utiliza diversas tecnologias (linguagens de programação, framework, banco de dados, etc.)
- Pode reutilizar componentes;
- Corresponde a 60 % da atividade de desenvolvimento.

Testes

- Permite a verificação do sistema segundo as especificação realizadas na fase de projetos;
- Os módulos são testados e depois integrados;
- Principal produto: Relatório de testes

Implantação

- Sistema é empacotado;
- Instalado no ambiente do usuário;
- Elaboração dos manuais do sistema;
- Carga de dados;
- Treinamento dos usuários;
- Migração de sistema / dados (eventualmente);

Equipe de Desenvolvimento de Software

Componente Humano

- Desenvolvimento de Software é uma tarefa cooperativa;
- Demanda especialistas em áreas específicas:
  - Técnica de TI;
  - Especialistas do domínio a ser desenvolvido.

Equipe Típica de TI

- Gerente;
- Analistas;
- Projetistas;
- Programadores;
- Clientes;
- Grupo de Testes;
- Grupo de Qualidade.

Gerente de Projetos

- Gerencia e coordena as atividades necessárias a construção de um sistema;
- Realizar o orçamento do projeto;
  - Estimar tempo e custos do projeto;
- Definir qual processo de desenvolvimento adotar;
- Cronograma de execução;
- Alocação de Recursos;
- etc.

Analistas

- Analista de sistema – Conhecimento do domínio;
- Entender os problemas do domínio;
- Definir requisitos do sistema;
- Comunicar-se com especialistas do domínio;
- Conhecer sem ser especialista (vocabulário, termos do negócio, etc);

Analistas

- Enteder as necessidades do cliente;
- Repassar esse entendimento aos desenvolvedores de software;
- É ponte de comunicação entre a equipe de negócios e a equipe de TI;
- Dominar a modelagem de sistemas;

Analista de Negócio x Analista de Sistemas

- Analista de Negócio
  - O que o cliente faz?
  - Por que o cliente faz?
  - As práticas atuais fazem sentindo na organização?

- Analista de Sistemas
  - Traduz as necessidades dos clientes em características de um produto de software?

Analistas - Características

- Capacidade de comunicação;
  - Escrita e Fala;
  - Agente facilitador da comunicação entre cliente e equipe técnica;
  - Bom relacionamento interpessoal;
  - Conhecimento técnico;
  - Ética profissional;
    - Tem acesso a informações estratégicas e sigilosas da organização e do projeto.

Projetistas

- Avaliar as alternativas de solução do problema modelado;
- Gerar especificações técnicas e detalhadas da solução;
- A atividade é também chamado de projeto físico;
- Trabalham nos modelos resultantes da análise;

Projetistas

- Adicionam aspectos tecnológicos aos modelos gerados pela análise;
- Tipos de Projetistas:
  - de Interface Gráfica;
  - de Redes;
  - de Banco de dados;
  - etc.

Arquitetos de Software

- Desenvolvimento de sistemas complexos;
- Elaborar a arquitetura do sistema;
- Definir os subsistemas;
- Definir as interfaces entre os sistemas;
- Tomar decisões técnicas detalhadas;
- Trabalha em conjunto com o gerente para priorizar e organizar o plano de projeto.

Programadores

- Implementação do sistemas;
- Uma equipe é composta por vários programadores;
- Conhecimento profundos em lógica e linguagem de programação;
- Conhecimentos em banco de dados (ler modelos de dados e utilizar instruções SQL);

Programadores

- Conhecimentos das principais ferramentas de desenvolvimento (frameworks, geradores de códigos, ferramentas de testes, etc);
- Nem sempre um bom programador poderá ser um bom analista.
- Um programador não muito talentoso pode se tornar um bom analista.

Especialistas do Domínio

- Também conhecido como especialista do negócio (ou cliente);
- Profundo conhecedor da área de negócio em que o sistema será construído;
- Tipos de Clientes:
  - Usuário;
  - Contratante.

Cliente Usuário x Cliente Contratante

- Cliente Usuário:
  - Quem utiliza o sistema;
  - É um especialista do domínio;
  - Interage com o analista de sistemas;
  - Auxilia no Levantamento de requisitos;

- Cliente Contratante:
  - Quem encomenda / patrocina o desenvolvimento do sistema;

Clientes Internos

- Para produtos de software encomedados por departamentos estratégicos de uma organização (marketing);
- Produtos voltados para o mercado de massa:
  - Processadores de Textos;
  - Editores Gráficos;
  - Jogos Eletrônicos;
  - etc.

Participação do cliente

- Fator determinante do sucesso de um projeto de desenvolvimento de software;
- Seu distanciamento está associado a projetos fracassados(prazos e custos);
- Um usuário satisfeito é um legítimo participante no desenvolvimento do sistema.

Avaliadores da Qualidade

- Características de um Sistema de Software de Qualidade:
  - Desempenho;
  - Confiabilidade;
- Asseguram a adequação do processo de desenvolvimento e do produto de software a padrões de qualidade estabelecidos pela organização.

Modelos de Ciclo de Vida

Modelos de ciclo de vida

- Desenvolver software envolve diversas fases;
- Encadeamento específico das fases para a construção do sistema;
- Diferenças entre os modelos se deve a maneira como as fases são encadeadas.

O modelo em Cascata

- Também chamado de Clássico ou linear;
- Característica: Progressão sequencial entre as fases;
- Eventualmente pode ocorrer retroalimentação de uma fase com a anterior.
- Muito utilizado em conjunto com a metodologia estruturada;

O modelo em Cascata

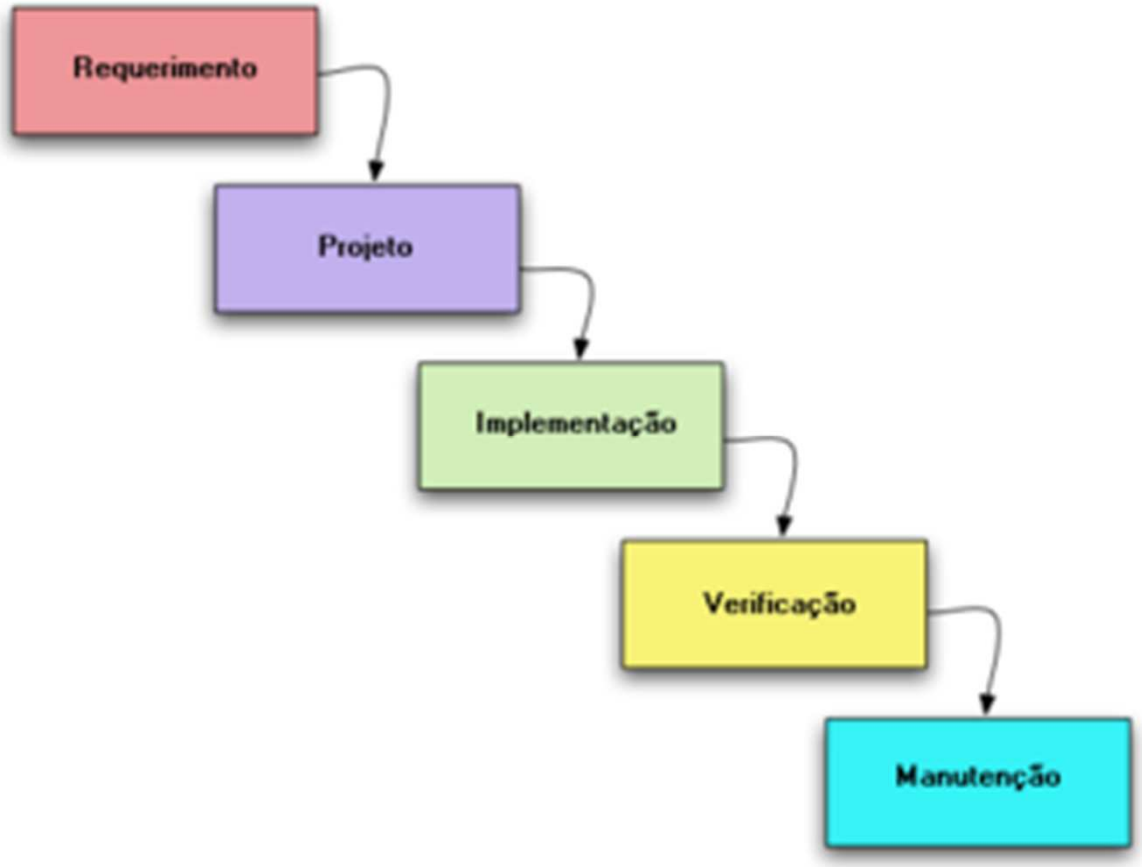


Figura 1 – Representação do processo em Cascata

O modelo em Cascata - Desvantagens

- Não está em conformidade com a realidade, devido algumas fases poderem ser realizadas em paralelo;
- Detecção de erros somente quando o usuário inicia a utilização do sistema;
- Demora na entrega do sistema;
- Sistema é entregue ao final de um ciclo de desenvolvimento;

O modelo Iterativo e Incremental

- Solução aos problemas identificados no modelo em cascata;
- Divide o desenvolvimento de software em ciclos;
- Cada ciclo é composto das fases de:
  - Análise
  - Projeto
  - Implementação e
  - Testes

O modelo Iterativo e Incremental

- A cada ciclo de desenvolvimento, um subconjunto de requisitos são trabalhados;
- Os requisitos são particionados conforme seu grau de prioridade e risco;
- Ao final de cada ciclo uma versão do produto é entregue ao usuário(extensão);
- Isso produz versões evolutivas do produto;
- Ocorre até a entrega final do produto.

O modelo Iterativo e Incremental

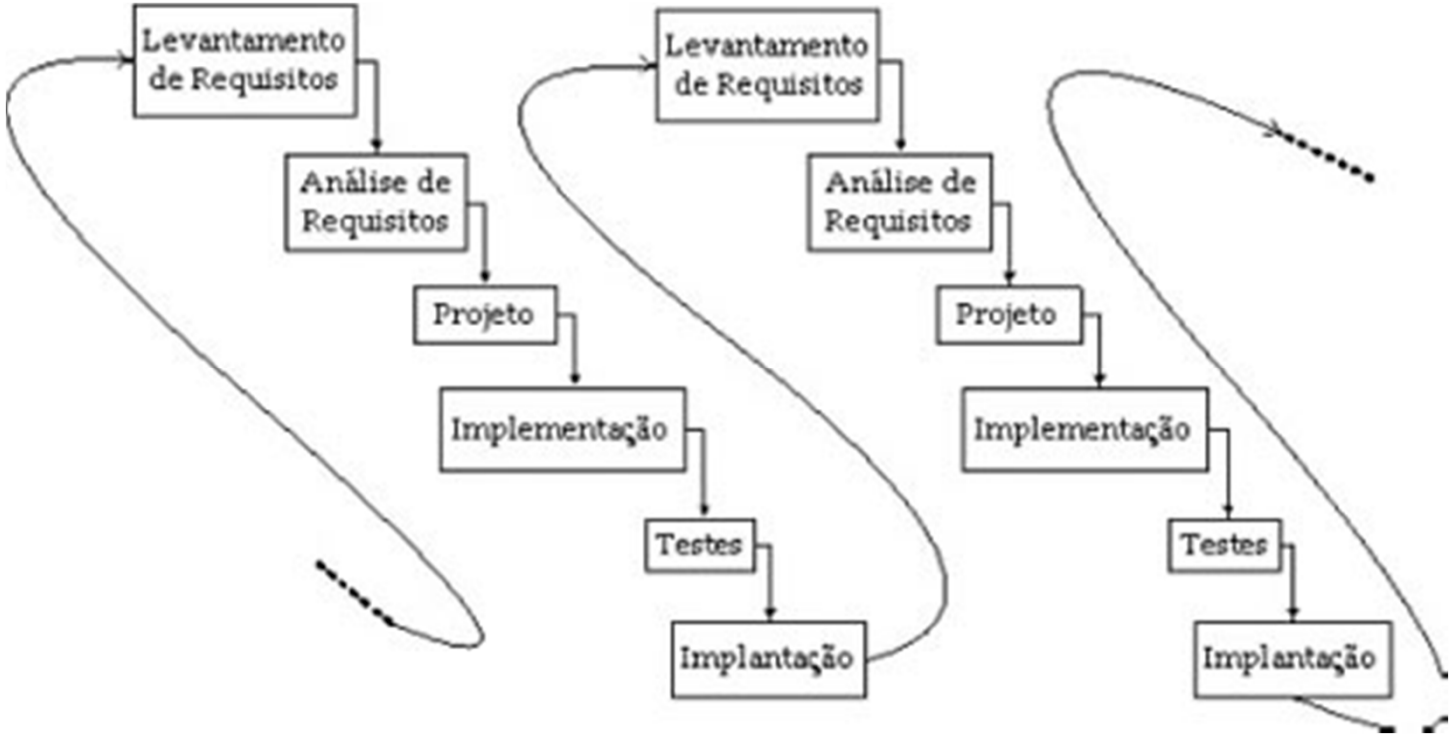


Figura 2 – Representação do processo iterativo e incremental

O modelo Iterativo e Incremental

- Pode ser visto como uma generalização do modelo em cascata;
- Pois a cada iteração, o produto é incrementado com uma nova versão;
- Cada incremento é realizado de forma sequencial.

O modelo Iterativo e Incremental

- Incentivo a participação do usuário nas atividades do desenvolvimento;
- Permite reduzir erros de interpretação;
- Permite gerenciar os riscos;

* Riscos – Possibilidade da ocorrência de eventos que acarretem em prejuízo ao projeto (custos, prazos, satisfação do cliente, etc).

O modelo Iterativo e Incremental

- Riscos inerentes ao desenvolvimento de software:
  - Não satisfação dos requisitos;
  - Verba acaba antes do projeto terminar;
  - O software pode não ser:
    - Adaptável, manutenível ou extensível
  - O software pode ser entregue tarde de mais ao usuário.

O modelo Iterativo e Incremental

- Os requisitos mais arriscados são considerados primeiro;
- Permite identificar mais cedo problemas como:
  - Inconsistências entre requisitos;

"Se você não atacar os riscos ativamente, então estes irão ativamente atacar você"

Tom Gilb, 1988.

/FabioJose

O modelo Iterativo e Incremental - Desvantagens

- Gerenciamento do projeto é mais difícil;

O modelo Iterativo e Incremental

- Dimensões do processo iterativo incremental:
  - Temporal
  - De atividades

O modelo Iterativo e Incremental

- Dimensão Temporal:
  - Processo estruturado em fases;
  - Em cada fase pode existir uma ou mais iterações;
  - Cada iteração tem um período pré-estabelecido: 2 a 6 semanas;
  - Ao final de cada iteração é entregue uma versão do produto;
  - O produto pode ser entregue ao usuário ou ser um incremento interno.

O modelo Iterativo e Incremental

- Dimensão de Atividades:
  - Conjunto de atividades realizadas durante uma iteração;

O modelo Iterativo e Incremental

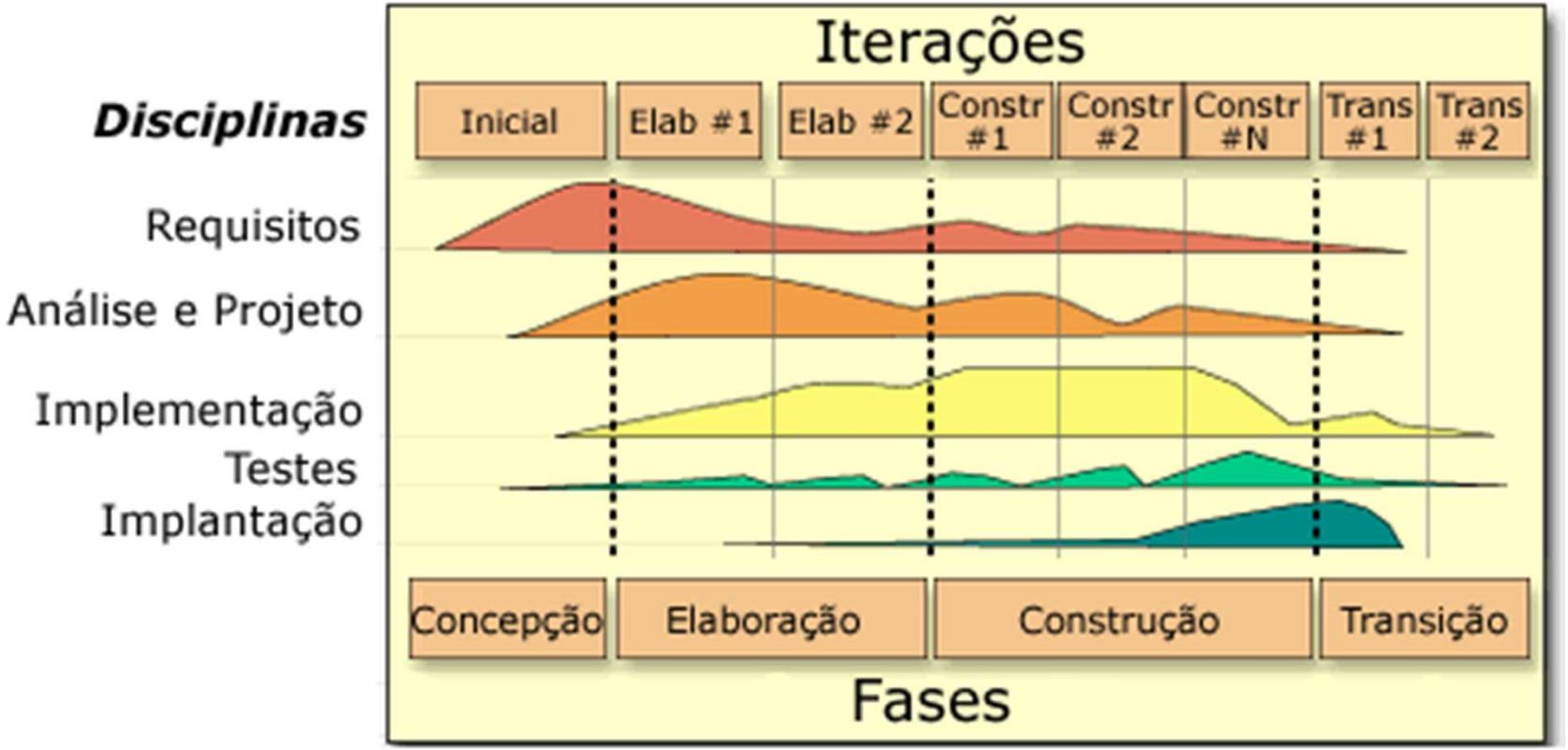


Figura 3 – Representação temporal do processo iterativo e incremental

Ferramentas

- UML – Unified Modeling Language;
- Prototipagem;
- Ferramentas CASE;

UML

- Linguagem de Modelagem de Software;
- Independente de Processo de Desenvolvimento;
- Ferramenta para construção de modelos do sistema de software orientados a objetos.

Prototipagem

- Técnica de complemento à análise de requisitos;
- É um esboço de parte do sistema;
- Exemplos de Protótipos:
  - Telas de entrada de dados;
  - Telas de saída;
  - Subsistemas;
  - etc.

Prototipagem

- Faz uso de IDEs para construção de GUIs;
- Essas ferramentas em geral são gráficas e utilizam a RAD – Construção Rápida de Aplicações;
- Qualquer aspecto que precise ser bem entendido é alvo potencial da prototipagem.

Prototipagem

- Após o levantamento de requisitos um protótipo do sistema é construído;
- Esse protótipo é então validado pelo usuário;
- Permite assegurar que os requisitos foram bem entendidos;
- O protótipo, após validado e refinado, pode ser descartado ou utilizado como versão inicial do sistema.

Ferramentas CASE

- CASE – *Computer Aided Software Engineering*
- Ferramentas que auxiliam na construção de modelos do sistema;
- Permitem:
  - a integração do trabalho de cada membro da equipe;
  - Gerenciamento do andamento do desenvolvimento;
  - etc.

Ferramentas CASE

- Tipos de Ferramentas:
  - Ferramentas CASE;
  - IDEs
  - Suporte ao Ciclo de Desenvolvimento do software

Ferramentas CASE

- Características:
  - Criação e manutenção de diagramas;
  - Manutenção da consistência entre os modelos gerados;
  - Engenharia *Round-Trip* – Capacidade de interagir com código-fonte;
    - Engenharia Direta – Gera o código;
    - Engenharia Reversa – Gera o modelo a partir do código;
  - Rastreamento de requisitos – Localizar artefatos para atualização de um requisito;

IDEs

- *Integrated Development Environment*
- Permitem a codificação do sistema;
- Facilidades que fornece:
  - Edição do código-fonte;
  - Compilação;
  - Execução;
  - Depuração;
  - Refatoração.

Suporte ao Ciclo de Desenvolvimento do software

- Facilidades que fornece:
  - Geração de relatórios de testes;
  - Gerenciamento de versões;
  - Suporte a definição de testes automatizados;
  - Monitoração e averiguação do desempenho;
  - Tarefas Gerenciamento.

Bibliografia Básica

- BEZERRA, Eduardo. Princípios de análise e projeto de sistemas com UML. Ed.Campus, 2002 – Capítulo 2 (p.21 a 45)