Técnico em Desenvolvimento de Sistemas Online

ANÁLISE DE SISTEMAS E PROJETOS

GEEaD - Grupo de Estudo de Educação a Distância

Centro de Educação Tecnológica Paula Souza

Expediente

GEEaD – CETEC
GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
ANÁLISE E PROJETO DE SISTEMAS I

Autores:
Eliana Cristina Nogueira Barion

Revisão Técnica:
Lilian Aparecida Bertini

Revisão Gramatical:
Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:
Flávio Biazim

APRESENTAÇÃO

Este material didático do Curso Técnico em Desenvolvimento de Sistemas modalidade EaD foi elaborado especialmente por professores do Centro Paula Souza para as Escolas Técnicas Estaduais – ETECs.

O material foi elaborado para servir de apoio aos estudos dos discentes para que estes atinjam as competências e as habilidades profissionais necessárias para a sua plena formação como Técnicos em Desenvolvimento de Sistemas.

Esperamos que este livro possa contribuir para uma melhor formação e aperfeiçoamento dos futuros Técnicos.

AGENDA 08

INTRODUÇÃO À ANÁLISE E PROJETOS DE SISTEMAS

MERGULHANDO NO TEMA...

As empresas de desenvolvimento de software têm enfrentado problemas com produtos de baixa qualidade, atrasos nas entregas dos projetos, inconformidades com as solicitações dos clientes não atendendo as suas necessidades, aumento significativo dos custos previstos etc. Esses e outros problemas podem ser causados pela falta de gerenciamento nos processos de desenvolvimento do software decorrentes da coleta de dados, na etapa de levantamento dos requisitos.

Esses problemas podem trazer impactos econômicos e humanos enormes. Por essa e outras razões, é recomendado o uso de metodologias de desenvolvimento capazes de melhorar a produtividade e a qualidade do software.

Metodologia de desenvolvimento é um conjunto de práticas recomendadas para o desenvolvimento de softwares, sendo que essas práticas, geralmente, passam por fases ou passos, que são subdivisões do processo para ordená-lo e melhor gerenciá-lo (Sommerville, 2000).

Modelos e Metodologias de Desenvolvimento Tradicionais

Modelo em Cascata ou Clássico

O Modelo Cascata é o primeiro processo publicado (ROYCE, 1970), que também é conhecido por abordagem top down, foi o mais utilizado nos anos 80, onde exigia uma complementação a cada fase. O termo “cascata” vem da proposta de desenvolvimento de projeto encadeada, sequencial e sistemática, que pressupõe terminar uma etapa para iniciar outra, conforme pode ser observado na Imagem 5.

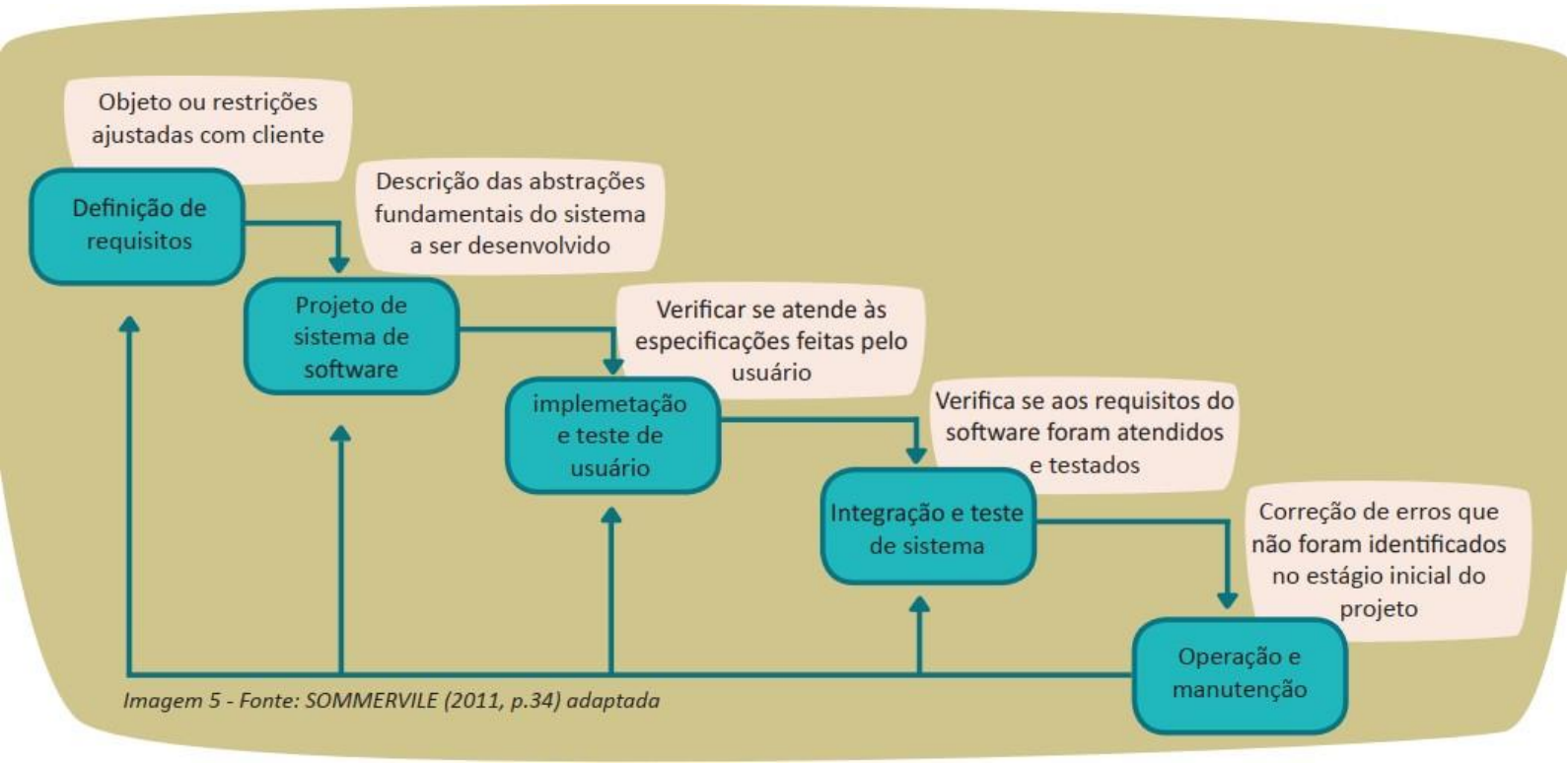


Imagem 5 - Fonte: SOMMERVILE (2011, p.34) adaptada

Esse modelo considera cinco etapas mais comuns:

Etapa 1 - Análise e definição de requisitos: devemos entender quais as funcionalidades que o sistema irá executar. Neste processo, é necessário definir os requisitos do sistema e validar os serviços e restrições.

Etapa 2 - Projeto de sistema e software: compõe atividades para assegurar que o sistema ou produto de software seja desenvolvido e entregue ao cliente no prazo estipulado.

Etapa 3 - Implementação e teste de unidade: a implementação ocorre após a homologação do software, o objetivo do teste de unidade é testar todas as unidades do sistema.

Etapa 4 - Integração e teste de sistema: verifica se foram solucionados erros de "comportamento" do software e assegura que as entradas definidas produzam resultados reais que coincidam com os requisitos especificados pelo cliente.

Etapa 5 - Operação e manutenção: garante o bom funcionamento dos serviços direcionando-o para um bom desempenho.

Apesar do resultado de cada fase ser uma aprovação para o início da fase seguinte, como mostra a imagem 05, na prática das empresas, esse modelo não funcionava de forma tão linear como se esperava pois, por vezes, acontecia encontrar problemas relacionados à fase anterior, o que causava retrabalho e atraso na fase atual (PUC **https://www.maxwell.vrac.puc-rio.br/21839/21839_3.PDF**).

"Devido aos custos de produção e aprovação de documentos, as iterações são onerosas e envolvem um 'retrabalho' significativo. Portanto, após um pequeno número de iterações, é normal suspender partes do desenvolvimento, como a especificação e prosseguir com os estágios posteriores do desenvolvimento. Os problemas são resolvidos em um outro momento, ignorados ou reprogramados. O congelamento prematuro de requisitos pode significar que o sistema não fará o que o usuário deseja. Isso pode também levar a sistemas mal estruturados, pois os problemas de projeto foram contornados por meio de artifícios de implementação."

(SOMMERVILLE, 2007).

Assim, a grande desvantagem do Modelo Cascata é com relação à mudança necessária para aprimoramento do projeto tornando-o engessado, ou seja, impossibilita atender novas mudanças.

Por outro lado, uma das maiores vantagens desse modelo é a documentação abrangente e detalhada, realizada em cada fase do projeto, o que possibilita maior monitoramento do processo, de acordo com as especificações do cliente.

Modelo Iterativo e Incremental

O modelo iterativo e incremental foi criado para solucionar os problemas identificados no Modelo Cascata aquelas atividades sequenciais que necessitavam a finalização de uma tarefa para iniciar a outra.

Os dois padrões mais conhecidos de sistemas iterativos de desenvolvimento são o RUP (Processo Unificado da Rational) e o Desenvolvimento ágil de software.

Esse modelo foi desenvolvido por meio de divisões em etapas, denominadas “incrementos”, que produzirão incrementalmente o sistema, até a sua versão final, conforme mostra a Imagem 06:

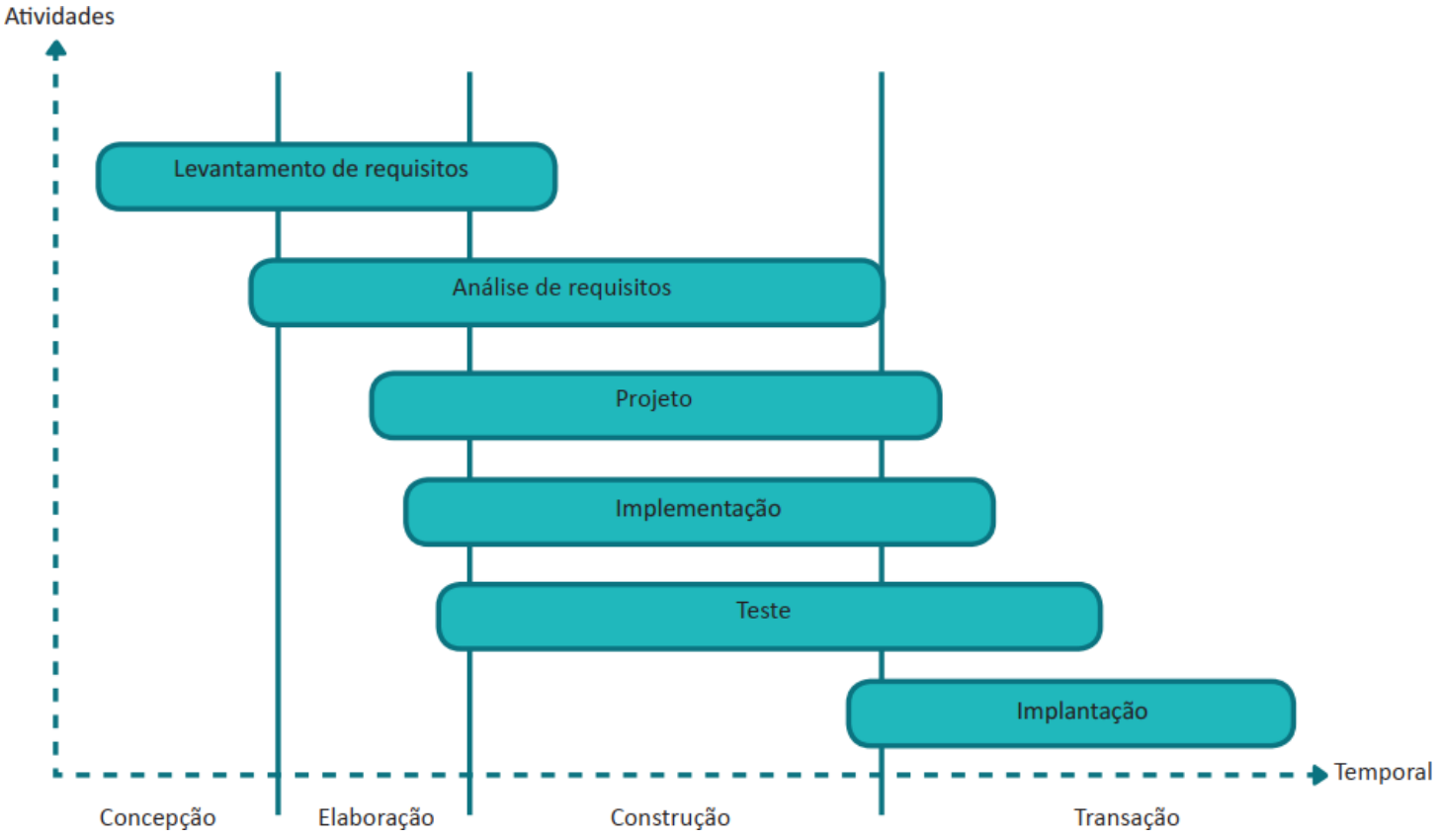


Imagem 06 - Adaptada RUP - Rational Unified Process (Processo Unificado Racional)

Em cada incremento, é realizado todo o ciclo do desenvolvimento de software, do planejamento aos testes do sistema já em funcionamento. Cada etapa produz um sistema totalmente funcional, apesar de ainda não cobrir todos os requisitos.

Um exemplo de um processo incremental seria de uma tela de cadastro de cliente que inicialmente contém funções de exibição de pessoa Física e Jurídica. Em um segundo incremento, poderia acrescentar algumas informações como dependentes, situação cadastral no CPF, no terceiro incremento, poderia adicionar relatórios com a relação de documentos pendentes.

No modelo Incremental, o produto é desenvolvido por partes, cada fase representa um subconjunto de requisitos, ou seja, um produto sem erros e pronto para ser utilizado pelo usuário final.

Observe na Imagem 06 que as atividades podem ocorrer simultaneamente e, a cada passo, o sistema é concluído com mais funcionalidades.

Analise, a seguir, as vantagens e desvantagens do Modelo Incremental:

Vantagens

- Redução de Custos: caso o desenvolvedor tenha problema no decorrer do projeto ele irá perder uma parte do valor e não o custo do projeto todo em função de permitir o feedback entre as fases;
- Redução de Riscos: com a divisão do projeto em pequenas etapas, as atividades são mensuradas de forma eficiente;
- Tempo: como os desenvolvedores estão dedicados em tempo integral e com escopo pequeno, o resultado é satisfatório;
- Mudança de Requisitos: são pontuais, serão efetivas, após o acordo das partes envolvidas;
- Forte participação e comunicação entre os desenvolvedores e usuários.

Desvantagens

- Atividades simultâneas podem dificultar o gerenciamento do projeto.
- O usuário pode se deslumbrar com a primeira entrega e solicitar mais funcionalidades ou confundir com um sistema final já pronto e operando.
- A primeira entrega pode não satisfazer o cliente.
- Falta de gerenciamento de custo.
- Pode haver atraso na entrega final do projeto, em função da possibilidade do escopo aumentar com a aparição de novos requisitos.

Modelo Espiral

Diferentemente do modelo Iterativo e Incremental, o modelo espiral utiliza a metodologia de desenvolvimento em sequência de interações e a cada volta representa uma fase (atividade), conforme pode observar na imagem 07:

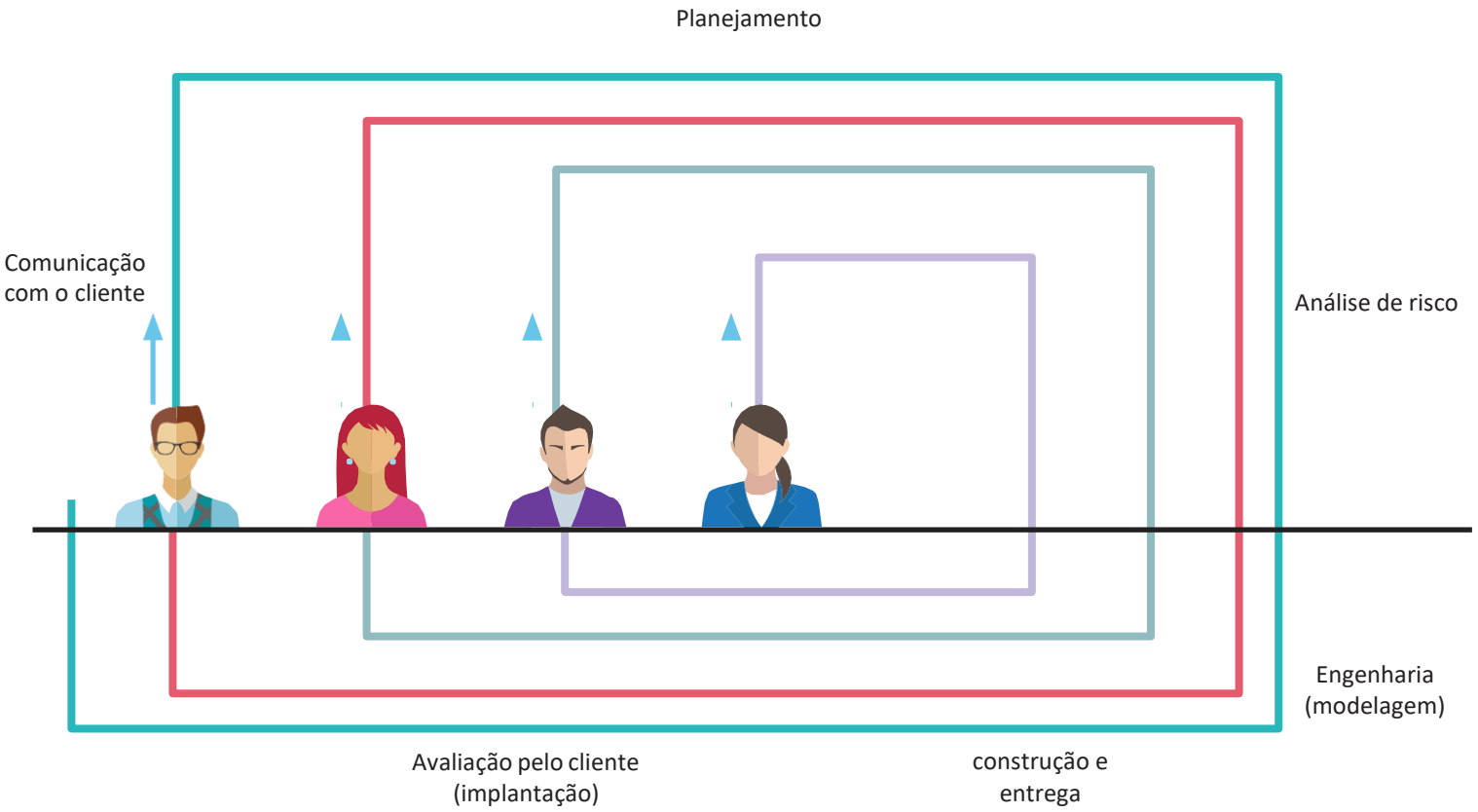


Imagem 07 - Arquivo GEEaD

O modelo espiral, criado por Barry Boehm em 1988, permite que as ideias e a inovação sejam verificadas e avaliadas constantemente. Isso porque, a cada interação, existe a volta da espiral que pode ser baseada em um modelo diferente e ter diferentes atividades. Esse padrão caracteriza-se, portanto, pelo desenvolvimento em sequência, aumentando a complexidade do processo conforme se chega mais próximo do produto final (PIVA; OLIVEIRA, 2010).

A cada ciclo da espiral, uma série de atividades é realizada em uma determinada ordem, como comunicação com o cliente, planejamento, análise de riscos, Modelagem, Construção e Entrega e Avaliação pelo Cliente.

Cabe ao analista projetar o que realmente o cliente deseja. Para isso, deve conversar com o cliente, seja em uma entrevista ou por meio de um questionário, estudar o processo atual da empresa, envolvendo todas as ferramentas importantes para o desenvolvimento do projeto.

Vamos imaginar o seguinte cenário: Há um ano, o Senhor Martins montou uma empresa de comércio eletrônico, porém o controle de seu estoque é realizado manualmente em planilhas de Excel e livro caixa. Com este processo, o Sr. Martins vem perdendo diariamente vendas importantes devido à falta de controle de seus produtos.

Com base nesse cenário vamos criar um grupo de desenvolvedores?

As três figuras - stakeholder, usuários e desenvolvedor são extremamente importantes para entendimento e levantamento de requisitos do software em questão

Imagem 08 - Arquivo GEEaD

Após a identificação das pessoas envolvidas, será preciso aplicar algumas técnicas de levantamento de requisitos, vamos a elas:

- **Estudo de viabilidade (Relatório de Viabilidade)** – verifica se as informações coletadas possibilitam implementar o desenvolvimento do software. É importante deixar claro para o cliente o que pode ou não ser feito no software para não criar uma expectativa falsa.
- **Elicitação e análise de requisitos (Modelo de sistema)** - neste momento, conseguimos identificar os detalhes para o desenvolvimento software. A entrevista é uma das ferramentas que se pode utilizar para o cliente e/ou usuário relatar o fluxo de sua rotina de trabalho e as necessidades do processo como um todo.
- **Especificação de requisitos (Requisitos de usuário e de sistema)** – descrições necessárias para as funções essenciais no sistema.
- **Validação de requisitos (Documentação de requisitos)** – todos os processos de levantamento de requisitos devem ser documentados e qualquer mudança sugerida ou realizada no projeto também devem ser citadas, desta forma o cliente consegue validar todos os processos com clareza.

Os requisitos da engenharia de processo:

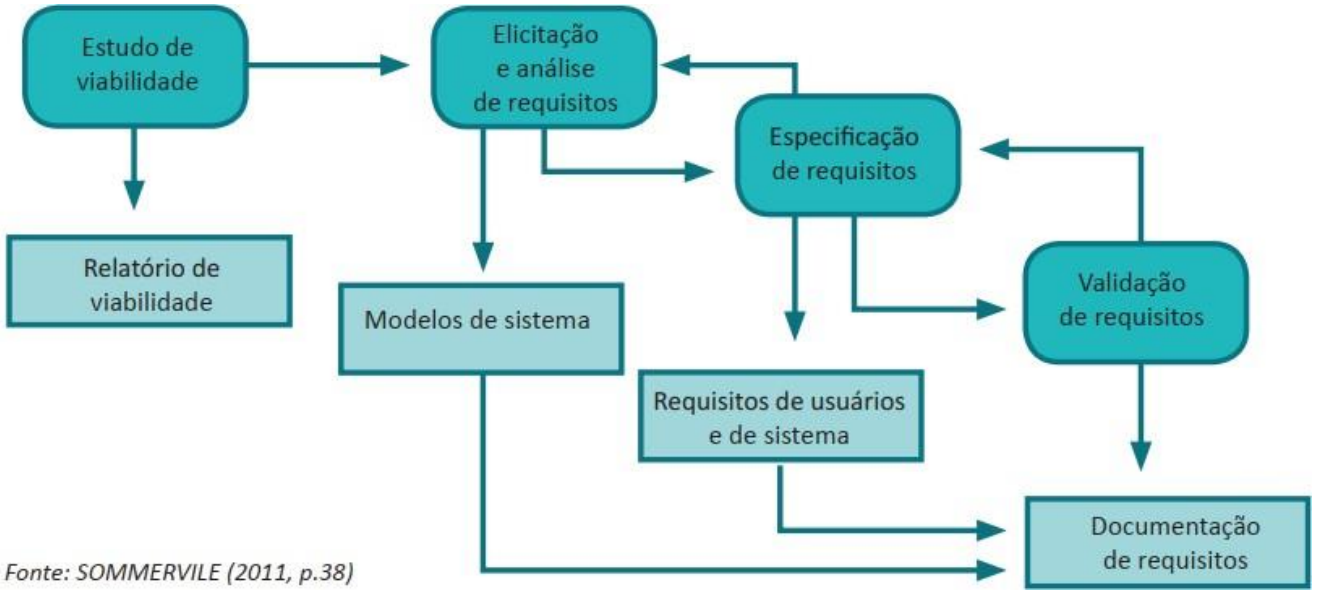


Imagem 09 - Fonte: SOMMERVILE (2011, p.38)

Veja que as setas apontadas para cada processo indicam que a comunicação deve ser constante e que a qualquer momento pode-se realizar uma nova análise, sempre atento ao tempo de entrega do Software.

Entretanto, antecedendo a etapa de projeto da arquitetura de software, há a necessidade de fazer o levantamento dos requisitos do sistema.

De um modo geral, o conjunto de requisitos de um sistema é definido durante as fases iniciais do processo de desenvolvimento. Os requisitos são descrições de como o sistema deveria se comportar, e contêm informações do domínio da aplicação e restrições sobre a operação do sistema.

Vamos conhecer os tipos de requisitos?

Requisitos Funcionais e Requisitos Não Funcionais

Os Requisitos Funcionais (RF) descrevem explicitamente as funcionalidades e os serviços do sistema, elicitados sob o ponto de vista do usuário. Por exemplo: Imagine um sistema de controle de presença do aluno em determinado curso:

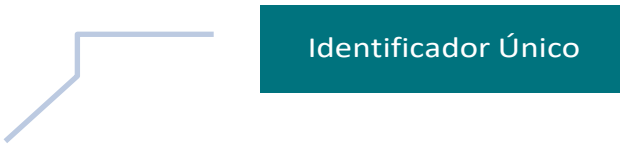


Tabela 01

ID	Nome
RF001	Consulta se aluno está presente
RF002	Inclui presença para aluno
RF003	Alterar presença em caso de falta
RF004	Consulta a turma que aluno está matriculado
RF005	Alterar para aluno desistente
RF006	Exclui aluno com falta consecutiva

Arquivo GEEaD

Sempre que for especificar os Requisitos Funcionais, procure colocar no nome da ação o que o sistema "FAZ". Para a declaração do RF não existe um padrão, porém é adotada esta estrutura na maioria das empresas. Contudo, Pressman (1995), apud PUC 2016, destaca que existe um mito a respeito dos requisitos, afirmando que muitos clientes imaginam que os requisitos podem ser alterados com facilidade, por acharem que o desenvolvimento de software é flexível. A realidade é que qualquer mudança nos requisitos em uma fase mais avançada pode causar atrasos e problemas no desenvolvimento do software, principalmente se forem fases encadeadas, como no modelo em cascata.

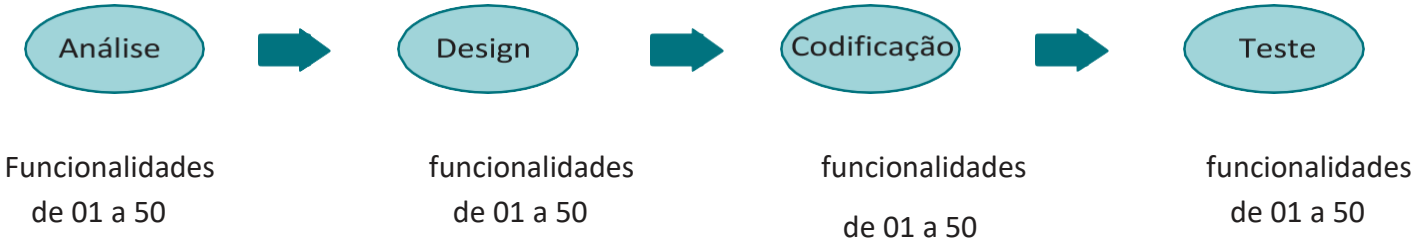
Diante dos problemas encontrados nos modelos tradicionais, como os que estudamos aqui nessa agenda, na década de 1990 começaram a aparecer alguns métodos que sugeriam uma abordagem de desenvolvimento ágil, capaz de fornecer uma maior autonomia e suporte às equipes de desenvolvimento.

Curioso para saber como essas informações irão quebrar o paradigma e melhorar os processos no desenvolvimento software?

Então, vamos lá!

Na imagem 09, temos uma prévia de como podemos trabalhar com a **metodologia ágil** em comparação ao **método Cascata**:

Cascata



Ágil

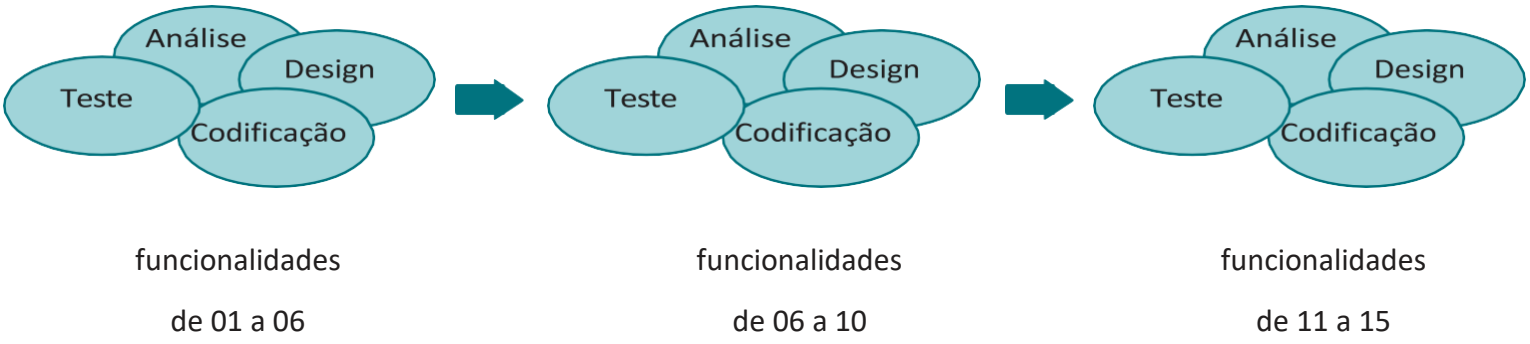


Imagem - 09 - Comparação entre Cascata e Ágil por SY, (2007). Tradução por Christiane Melcher. (PUC)

As metodologias ágeis foram inseridas no mercado na década de 1990, mas somente em 2001 se tornaram mais populares, quando um grupo de 17 especialistas em processos de desenvolvimento de software, liderados por Kent Beck assinaram o "Manifesto para o Desenvolvimento Ágil de Software" (https://blog.geekhunter.com.br/manifesto-agil/) com conceitos que serviriam como base para qualquer método ágil (PUC 2014):

- Indivíduos e interações ao invés de processos e ferramentas;
- Software executável ao invés de documentação;
- Colaboração do cliente ao invés de negociação de contratos;
- Respostas rápidas a mudanças ao invés de seguir planos. Para Pressman, 2011:

> *"A engenharia de software ágil combina filosofia com um conjunto de princípios de desenvolvimento. A filosofia defende a satisfação do cliente e a entrega de incremental prévio; equipes de projetos pequenas e altamente motivadas; métodos informais; artefatos de engenharia de software mínimos e, acima de tudo, simplicidade no desenvolvimento geral. Os princípios de desenvolvimento priorizam a entrega mais que a análise e projeto (embora essas atividades não sejam desencorajadas); também priorizam a comunicação ativa e contínua entre desenvolvedores e clientes". (Pressman, 2011, pag. 81)*

EXERCITANDO E APRIMORANDO

Jonas, técnico em Desenvolvimento de Sistemas, decidiu criar um controle das vendas para seu pai que estava tendo problemas para calcular com precisão o seu lucro e o total das despesas diárias. Vendo a necessidade do seu pai, Jonas decidiu adiantar o seu projeto, eliminando a etapa de validação do software.

Quais são as possíveis consequências que ele poderá enfrentar por ter excluído essa etapa?

Resposta: Ao deixar uma das etapas da validação de software, Jonas não consegue assegurar se o desenvolvimento do software estará correto ou não e se todos os requisitos foram atendidos. A validação é a certificação de que o sistema está de acordo com as expectativas do cliente.