
Newton
Quem se prepara, não para.

Engenharia de Software

2º período

Professora: Michelle Hanne

Engenharia de Software

Newton
Quem se prepara, não para.

De acordo com Pressman (2011, p.11), “*software* consiste em:

- instruções (programas de computador) que, quando executadas, fornecem características, funções e desempenho desejados;
- estruturas de dados que possibilitam aos programas manipular informações adequadamente;
- e informação descritiva, tanto na forma impressa como na virtual, descrevendo a operação e o uso dos programas”.

Conjunto de Atributos de Software

Newton
Quem se prepara, não para.

Esses atributos são sumarizados a seguir (Sommerville 2011):

- **Manutenibilidade**: o *software* deve ser escrito de forma que permita a evolução para atender às necessidades dos clientes.
- **Confiança e proteção**: um *software* deve ser confiável, ou seja, não causar prejuízos físicos ou econômicos no caso de falha do sistema. Em termos de proteção, o *software* deve garantir que usuários maliciosos não sejam capazes de acessar ou prejudicar o sistema.
- **Eficiência**: um *software* é eficiente quando ele responde rapidamente às ações dos usuários e não desperdiça recursos do sistema, como memória e tempo de processamento.
- **Aceitabilidade**: este atributo refere-se a aceitação por parte do usuário, ou seja, o *software* deve ser desenvolvido de acordo com o tipo de usuário (crianças, adultos). Por exemplo, um *software* para crianças requer mais cores, mais imagens e é mais fácil de interagir.

Aplicações Mobile e Computação na Nuvem

Newton
Quem se prepara, não para.

- Novo Mercado
- Grande quantidade de dispositivos móveis

Software como serviço: a tecnologia é ponte para a real necessidade do cliente.
Serviço de armazenamento na nuvem (Cloud Computing), em que o *software* é um mero meio para o compartilhamento e armazenamento de dados do usuário.

Linha de Produto de Software

Newton
Quem se prepara, não para.

A abordagem Linhas de Produtos de Software (LPS) ou Software Product Lines (SPL) é definida como o uso de **técnicas de engenharia** que permitem criar um **grupo de softwares similares** a partir de um conjunto de **características comuns** a todos esses sistemas (Reuso de Software).

Linha de Produto de Software

Newton
Quem se prepara, não para.

- Bastante utilizado principalmente para as abordagens e Metodologias Ágeis.
- Produz Softwares complexos e maiores

Engenharia de Software

Newton
Quem se prepara, não para.

A base para engenharia de *software* é definida por Pressman (2016) como uma tecnologia em camadas, são elas:

(i) Foco na qualidade: o que permeia o desenvolvimento do software

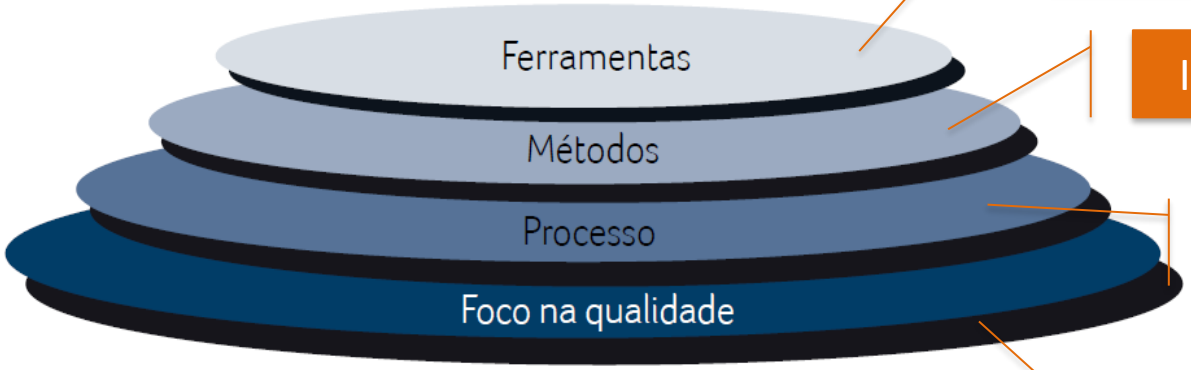
(ii) Processo: metodologia de desenvolvimento de software

(iii) Métodos: Técnicas/Etapas como comunicação, análise de requisitos, modelagem de projeto, construção de programa, testes e suporte.

(iv) Ferramentas: fornecem suporte para os Processos e Métodos.

Engenharia de Software

Figura 2 – Conceito de ética na engenharia de *software*



Fonte: PRESSMAN; MAXIM, 2011, p. 39.

Suporte automatizado ou semiautomatizado

Informações Técnicas

Metodologia

A base de sustentação da Engenharia de Software é a Qualidade

Ética na Engenharia de Software

Newton
Quem se prepara, não para.

- **Confidencialidade**: deve-se respeitar a confidencialidade das informações de seus empregados ou clientes, independentemente de ter sido assinado um acordo formal de confidencialidade.
- **Competência**: não aceitar um trabalho acima do seu nível de competência.
- **Direitos de propriedade intelectual**: deve-se ter conhecimento das leis do país a respeito da propriedade intelectual, como patentes e *copyright*.
- **Mau uso do computador**: não fazer mau uso de seus conhecimentos técnicos a outras pessoas, como disseminar vírus ou outros malwares pela rede.

Apresenta-se no contexto da moralidade

Processo de Software

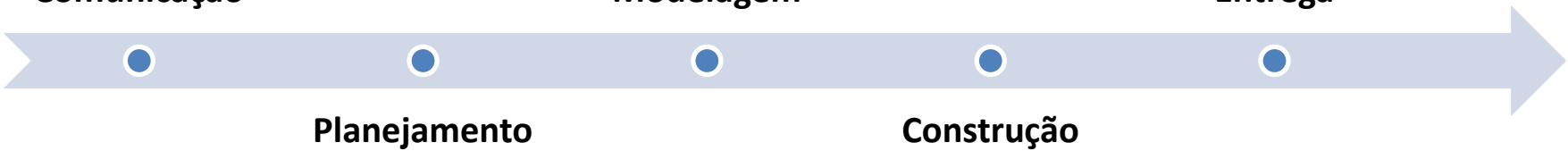
Newton
Quem se prepara, não para.

Processo de *software* é também conhecido como modelo do ciclo de vida de um *software*.
Cada atividade é composta por um conjunto de ações; cada ação é definida por um conjunto de tarefas que identifica:

- *as tarefas de trabalho a serem completadas.*
- *os artefatos de software que serão produzidos.*
- *os fatores de garantia da qualidade que serão exigidos.*
- *os marcos utilizados para indicar progresso.*

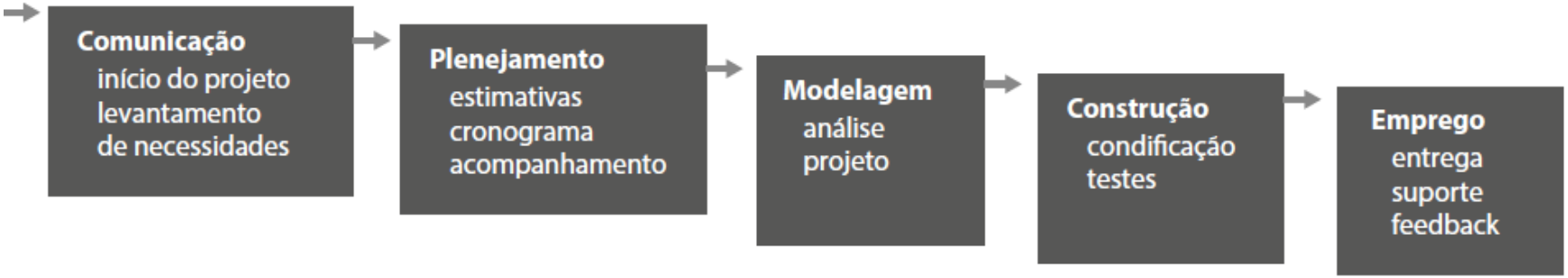
Atividades Fundamentais no Processo de Software

Newton
Quem se prepara, não para.



Modelo em Cascata

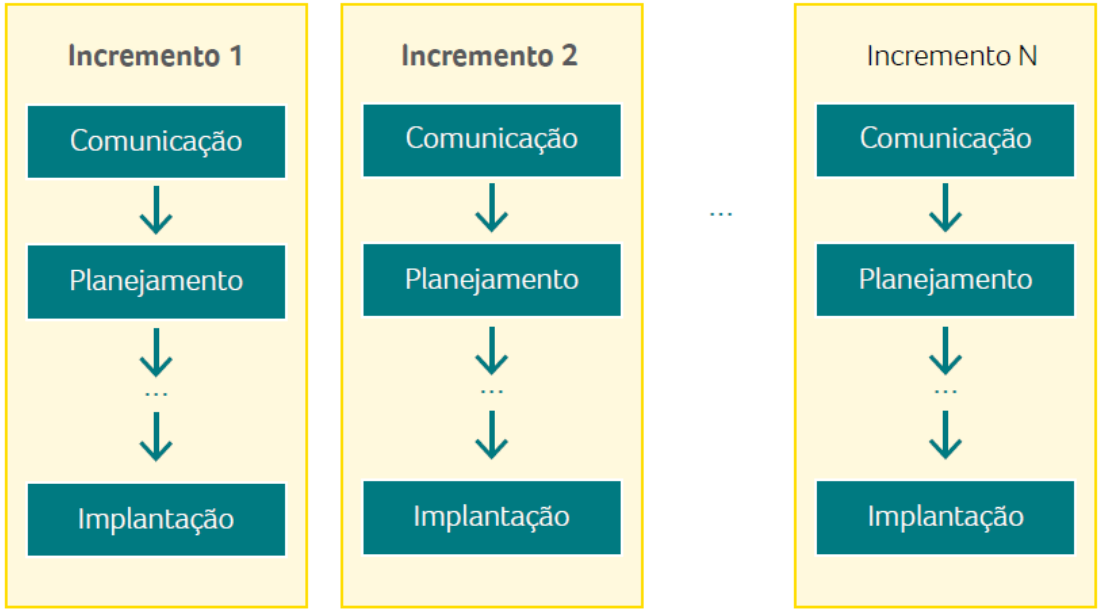
Figura 5 – Modelo cascata



Fonte: PRESSMAN; MAXIM, 2016.

Modelo Incremental

Figura 6 – Modelo incremental



Fonte: SOUZA, [200-?].

Modelo Evolucionário

O modelo Evolucionário é iterativo e permite a evolução ao longo do tempo, de versões cada vez mais completas do software.

No modelo espiral cada iteração na espiral representa uma fase do processo na qual são reavaliados os riscos. As fases são:

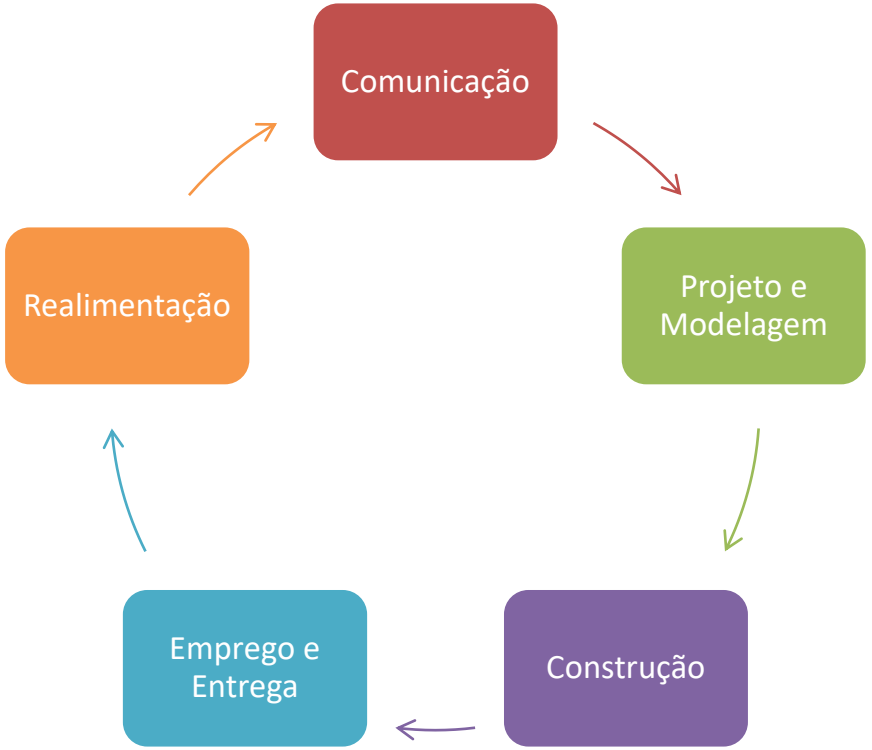
- Comunicação
- Planejamento
- Modelagem
- Construção
- Emprego

Figura 8 – Modelo espiral típico

Fonte: PRESSMAN; MAXIM, 2016.

Modelo de Processo Prototipação

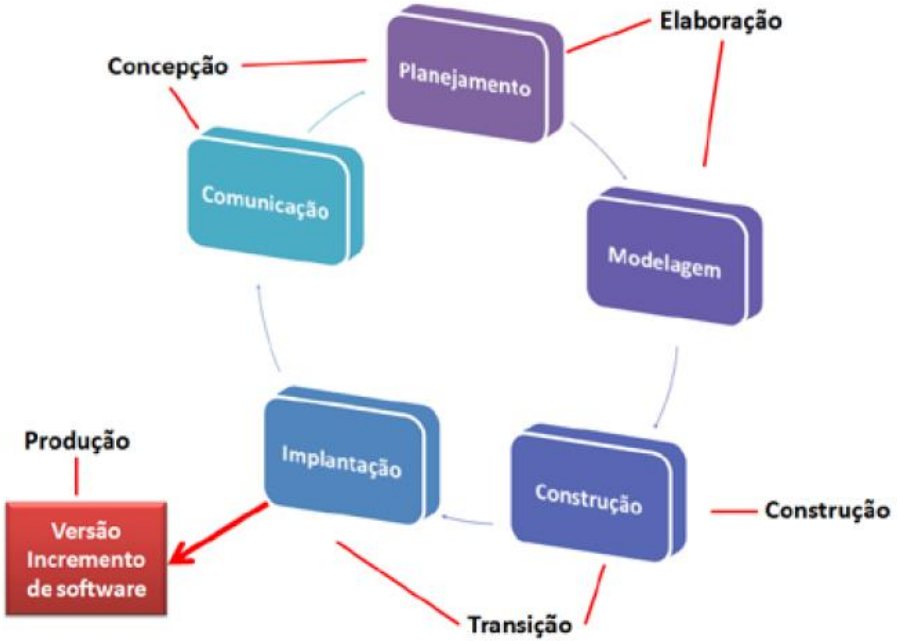
A prototipação é uma versão do sistema, ou parte, desenvolvida rapidamente para verificar e validar as necessidades do cliente.



Modelo Processo Unificado

O processo unificado **é um modelo iterativo** constituído de fases. Pressman e Maxim (2016) identificam cinco fases distintas no modelo do processo de *software*.

Figura 9 – O processo unificado



Fonte: PRESSMAN; MAXIM, 2016.

Modelo de Processo Pessoal

Newton
Quem se prepara, não para.

O modelo de processo pessoal (PSP) foi criado em 1997, um processo de software projetado para a **medição pessoal do desenvolvedor:**

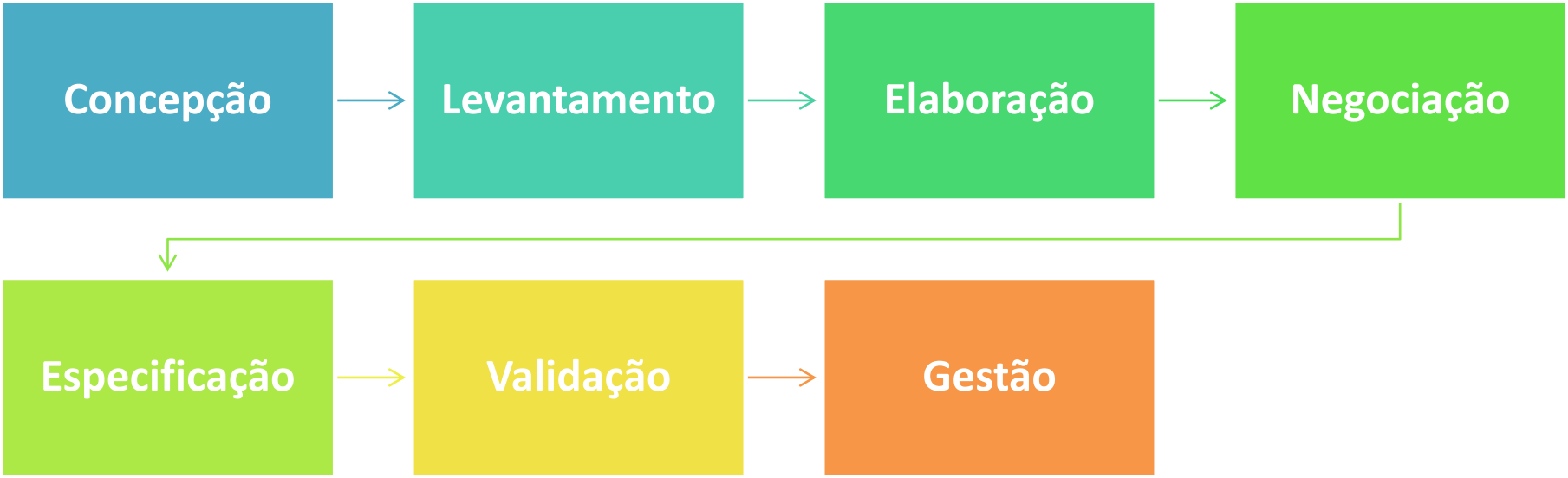
- **Planejamento**
- **Projeto de alto nível**
- **Revisão de projeto de alto nível**
- **Desenvolvimento**
- **Autópsia**

Autópsia: por meio de medidas e métricas coletadas, é determinada a eficácia do processo.

A responsabilidade é do programador.

Atividades da Engenharia de Requisitos

Newton
Quem se prepara, não para.



As Atividades não precisam ser realizadas exatamente de forma sequencial.

Fonte: Pressman, 2016

Etapas de Levantamento de Requisitos

Newton
Quem se prepara, não para.

Obtenção de requisitos: ocorre a interação com as partes interessadas (*stakeholders*) para coletar informações sobre os requisitos.

Classificação e organização de requisitos: se agrupam os requisitos relacionados, e esses são organizados de forma coerente.

Priorização e negociação de requisitos: é usada no momento em que se têm requisitos no sistema que começam a ser conflitantes entre si. Nesse momento, o analista deve procurar negociar os requisitos para evitar conflitos e também para priorizá-los.

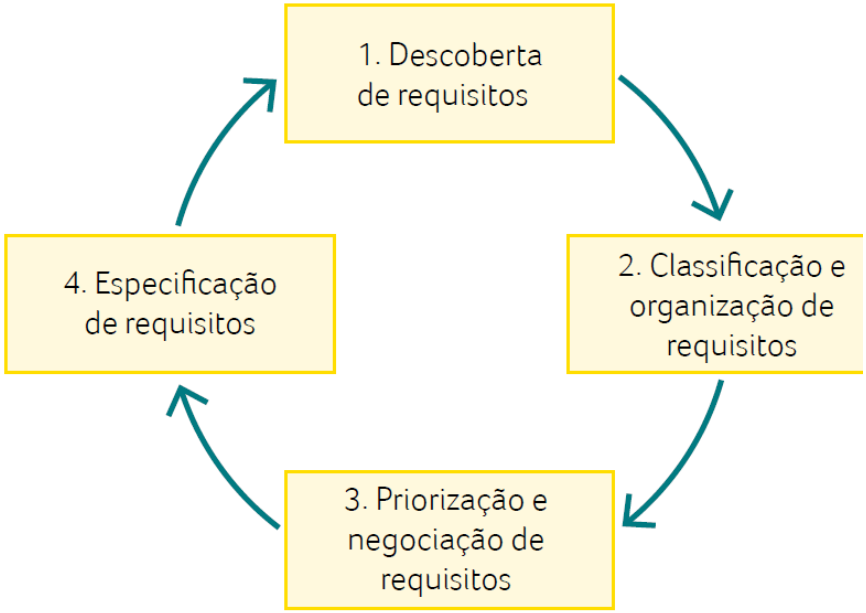
Documentação de requisitos: os requisitos levantados são documentados, podendo ser documentos formais ou informais, passando por um ciclo iterativo.

Fonte: Pressman, 2016

Etapas de Levantamento de Requisitos

Newton
Quem se prepara, não para.

Processo de levantamento e análise de requisitos



Fonte: SOMMERVILLE, 2011.

Fonte: Somerville, 2011

Especificação de Requisitos

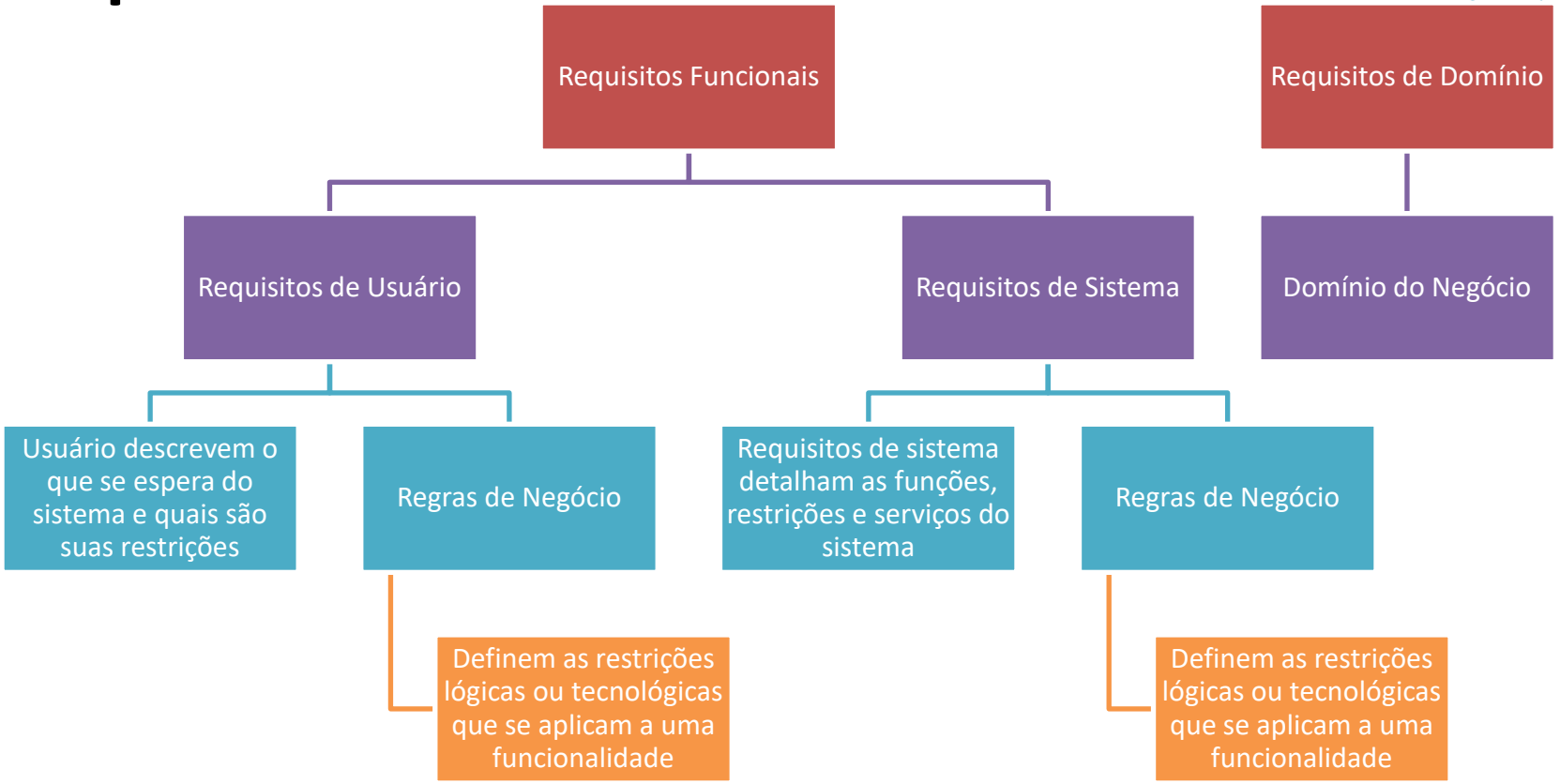
Newton
Quem se prepara, não para.

Especificação é escrito em nível altamente detalhado das especificações de requisitos de software podendo ser feitas através de modelos gráficos, bem como diagramas, documento por escrito, modelo matemático etc

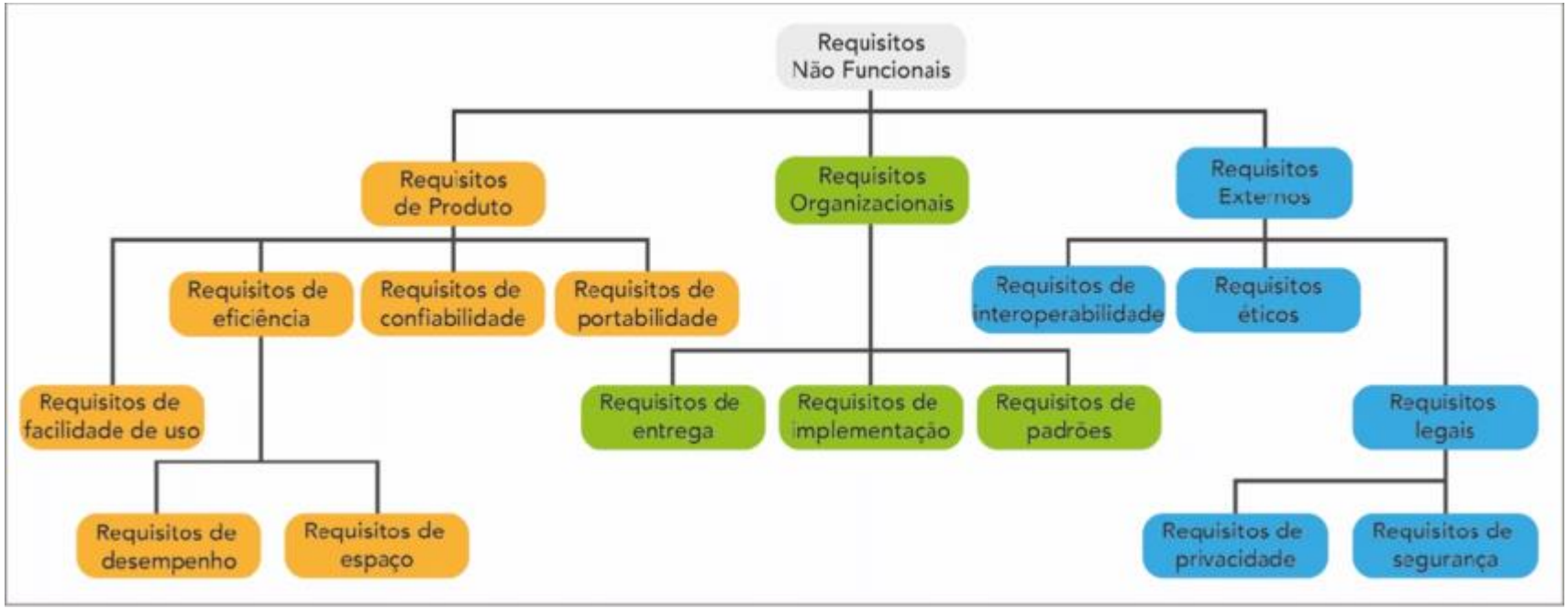
Requisitos

- **Requisito Funcional** descreve a funcionalidade do software. Exemplo: O sistema deverá realizar o cadastro do usuário; O sistema permitirá que o usuário altere os seus dados; O sistema permite que o usuário consulte o seu histórico.
- As **regras de negócio** são restrições escritas para os requisitos. Exemplo:
  - Requisito Funcional: O sistema deverá realizar o cadastro do usuário;
  - Regras de negócio: Os usuários deverão preencher obrigatoriamente os campos nome, e-mail e senha; A senha deverá possuir 8 caracteres no mínimo.
- **Requisitos Não Funcional:** são requisitos que definem o comportamento do produto de software; estão relacionados a qualidade, usabilidade, velocidade, segurança, etc. Também podem ser externos, como adequação a leis ou internos da organização, como utilizar uma linguagem de programação específica. Exemplo: Todas as senhas deverão ser criptografas.; Todos os relatórios deverão ser gerados em PDF.

Requisitos Funcionais



Requisitos Não Funcionais



Tipos de Requisitos Não Funcionais. Fonte: Sommerville (2011) - https://www.researchgate.net/figure/Figura-1-Tipos-de-Requisitos-Nao-Funcionais-Fonte-Sommerville-2011-Os-RNFs-que-vem_fig1_311459854

Requisitos Não Funcionais

Newton
Quem se prepara, não para.

Requisitos de produto: definem o comportamento do produto, como, por exemplo, requisitos de velocidade, quanto de memória ele requer, quantidade de falhas aceitáveis, portabilidade e usabilidade.

Requisitos organizacionais: são definições das políticas do cliente ou do desenvolvedor, como padrões de processos e linguagem de programação.

Requisitos externos: como o próprio nome sinaliza, esses requisitos são definidos por outros, como sistemas no qual o produto irá interagir (interoperabilidade), conformidade com a lei e questões éticas.

- Desempenho
- Usabilidade
- Confiabilidade
- Segurança
- Disponibilidade
- Manutenção
- Tecnologias

Técnicas utilizadas para Levantar os Requisitos

Newton
Quem se prepara, não para.

Pontos de vista: a captura de requisitos por diversos pontos de vista, através de workshops por exemplo.

Entrevistas: podem ser feitas com um roteiro predefinido ou de uma forma exploratória. Podem ocorrer em grupo ou individuais.

Cenário: exemplos reais que representam a interação com o sistema. Desenho de Fluxo ou Processo.

Caso de uso: identifica ações e atores que irão interagir com o sistema.

Etnografia: baseada na observação com o usuário, "o sombra".

IHM

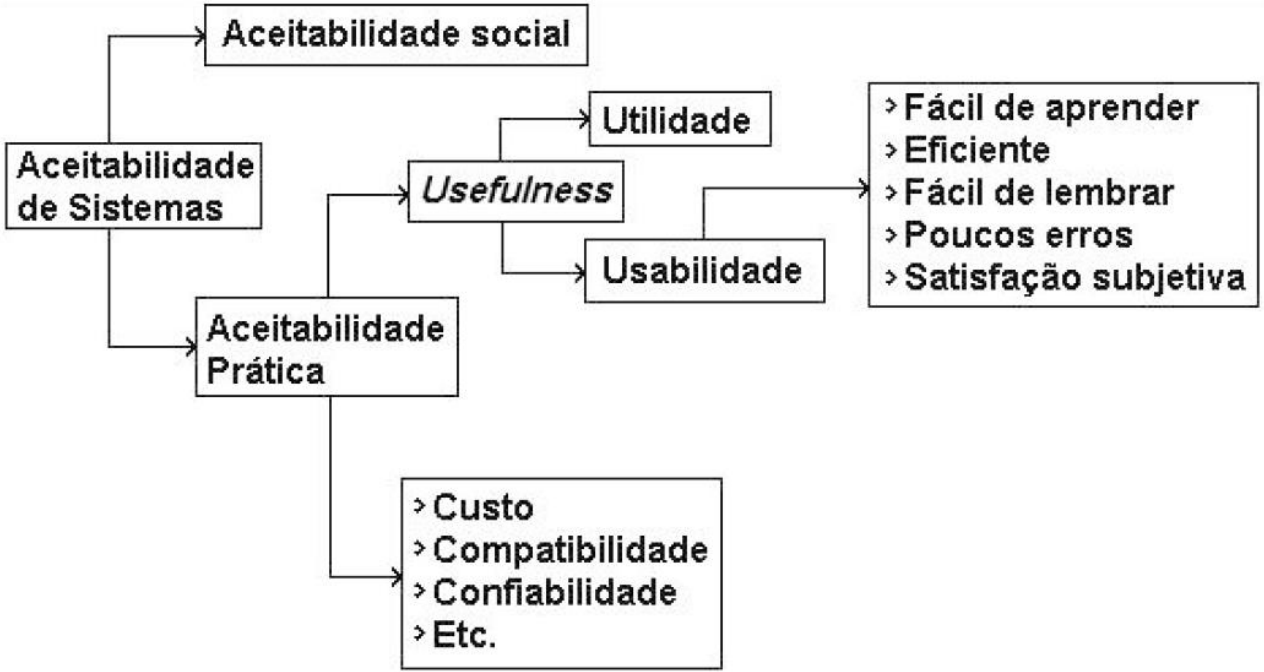
Newton
Quem se prepara, não para.

"A interface é definida como o local onde o contato entre duas entidades ocorre. Por exemplo, a interface entre usuário e computador é o monitor (ROCHA; BARANAUSKAS, 2000)."

IHM – Atributos da Aceitabilidade

Newton
Quem se prepara, não para.

Figura 14 – Atributos da aceitabilidade de um sistema



Fonte: ROCHA; BARANAUSKAS, 2000.

Análise e projeto de interfaces

Newton
Quem se prepara, não para.

Modelo de usuário visa estabelecer o perfil dos usuários finais do sistema.

- **Novatos**: trata-se de usuários que não possuem conhecimento pregresso do estilo e significado da interface.
- **Usuários intermitentes e com conhecimento**: trata-se de usuários que conhecem os termos da interface, mas ainda não têm incorporado o conhecimento para a execução das tarefas na aplicação.
- **Usuários frequentes e com conhecimento**: referem-se aos usuários que possuem o domínio completo da aplicação e que procuram atalhos e modos de interação abreviados.

Análise e projeto de interfaces

Newton
Quem se prepara, não para.

Modelo de projeto ou modelo do *designer*: refere-se aos conceitos que o *designer* (projetista) ou engenheiro de *software* tem em sua mente sobre o sistema. É criado a partir das informações capturadas dos usuários.

Modelo mental do usuário: também é conhecido como percepção do sistema. Refere-se à imagem do sistema que os usuários finais possuem em suas mentes, ou seja, a imagem da estrutura de como a aplicação funciona de acordo com que o usuário final internalizou na sua memória. Já o **modelo de usuário** refere-se ao que o usuário desenvolve para entender e explicar a operação do sistema.

Modelo de implementação ou imagem do sistema: refere-se à imagem construída do sistema computacional em conjunto com livros, manuais, arquivos de ajuda, entre outras informações de apoio. A **imagem do sistema** é formada pelo *layout* (aparência física) e a forma como o sistema responde.

Projeto de Interface

Nielsen e Wagner (1996) *apud* Pressman (2016, p. 339) propuseram uma lista do que não se deve fazer em projetos de interfaces, sendo um ótimo complemento aos princípios elencados anteriormente:

- Não force o usuário a ler muitos textos, principalmente quando for para explicar uma operação ou auxiliar na navegação.
- Não obrigue o usuário a descer até o fim da página.
- Não permita que a estética exceda a funcionalidade da aplicação.
- Não obrigue o usuário a procurar na tela como acessar *links* que o direcionem a outros conteúdos ou serviços.

Avaliação do projeto de interface

Newton
Quem se prepara, não para.

A avaliação de um projeto de interface se dá após a criação de um protótipo da(s) tela(s)

- Na primeira forma, será solicitado ao usuário que expresse seus pensamentos à medida que está navegando, o que permite ao projetista coletar o *feedback* imediato do usuário em relação à eficácia da interface.
- A segunda forma consiste em aplicar métodos estatísticos (por exemplo, questionários, formulários de avaliação) para avaliar a interface.

Referências

PFLEEGER, S. L. **Engenharia de software**: teoria e prática. Belo Horizonte: Prentice Hall, 2004.

PRESSMAN, Roger S. MAXIM, Bruce R. **Engenharia de Software - Uma Abordagem Profissional**. 8.ed. Porto Alegre: Amgh Editora, 2016. 968p. ISBN 9788580555332.

SOMMERVILLE, Ian. **Engenharia de software**. 8.ed. São Paulo: A. Wesley publishing company, 2010.