Engenharia de Requisitos de Software

3º período
Professora: Michelle Hanne Soares de Andrade
mhsandrade@sga.pucminas.br
michellehanne.andrade@gmail.com

Sumário

- Tipos de Processos de Software
- Software como Produto e Serviço
  - Linhas de produto de software

Etapas de Processos de Software

PUC Minas

Atividades Genéricas do Processo de Software:

- **Comunicação:** é a atividade na qual são levantados os requisitos junto ao cliente.
- **Planejamento:** são elaborados o cronograma, as análises de riscos e os recursos necessários.
- **Modelagem:** traduz a especificação em formatos simbólicos, tanto em nível de requisito quanto em nível
- de projeto.
- **Construção:** são gerados os códigos e testes do *software*.
- **Implantação:** é quando se coloca o produto em produção, ou seja, é nesta atividade que o cliente recebe o *software*.

Modelo do Ciclo de Vida

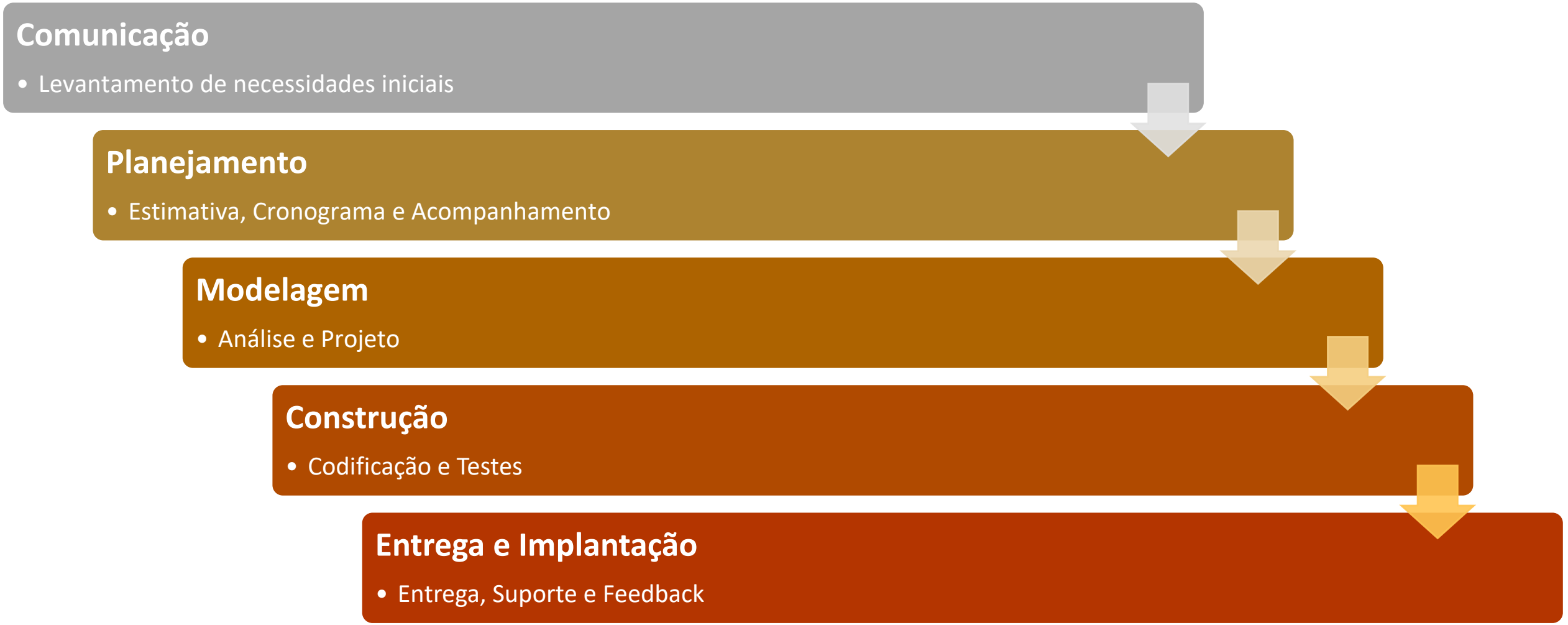
A escolha de um modelo de processo é fortemente dependente das características do projeto. Os principais modelos de ciclo de vida podem ser agrupados em três categorias principais:

- **modelos sequenciais**
- **modelos incrementais**
- **modelos evolutivos**

Modelo em Cascata

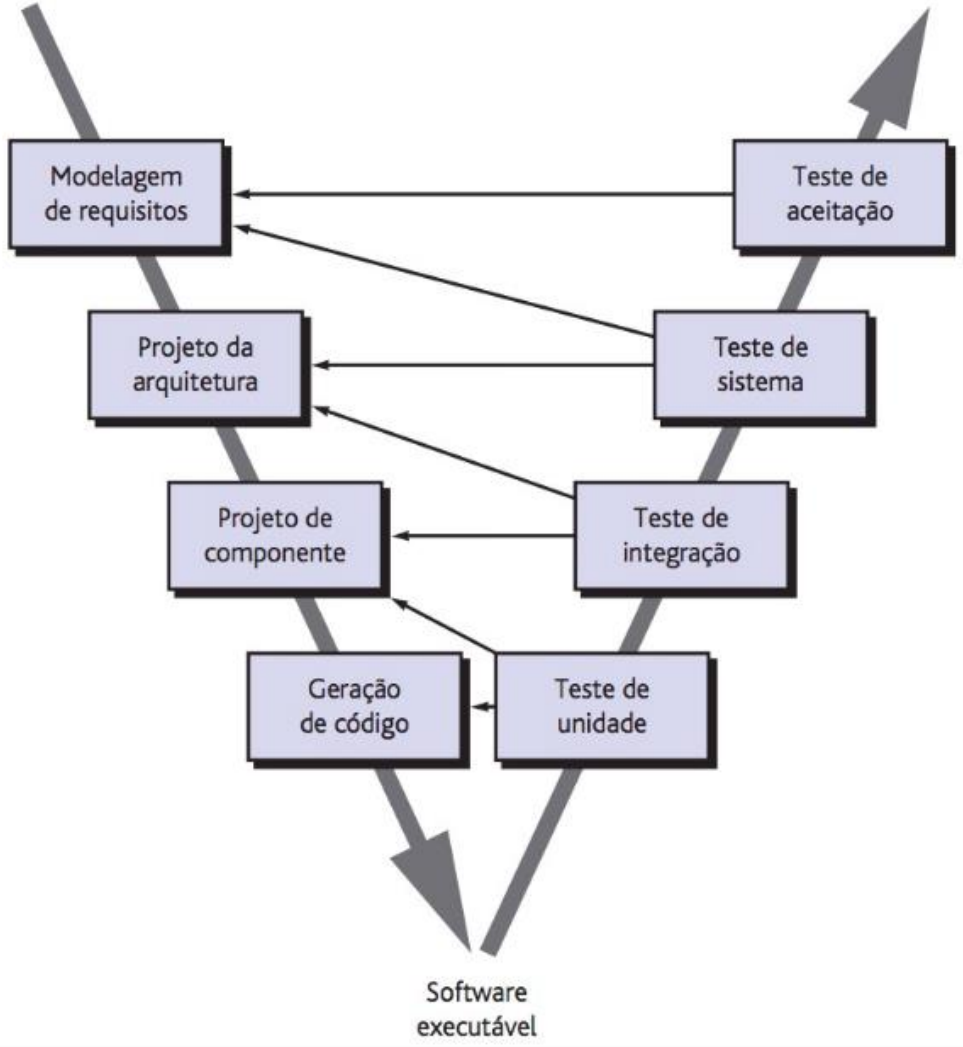
- O principal modelo da categoria sequencial é o modelo em cascata, a partir do qual outros modelos foram propostos.
- **O Modelo em Cascata (waterfall) também chamado de "modelo de ciclo de vida clássico", organiza as atividades do processo de desenvolvimento de forma sequencial.**

Modelo em Cascata



Modelo em Cascata. Baseado em Pressman, 2016.

Modelo em Cascata



O modelo V descreve a relação entre ações de garantia da qualidade e ações associadas a comunicação, modelagem e atividades de construção iniciais.

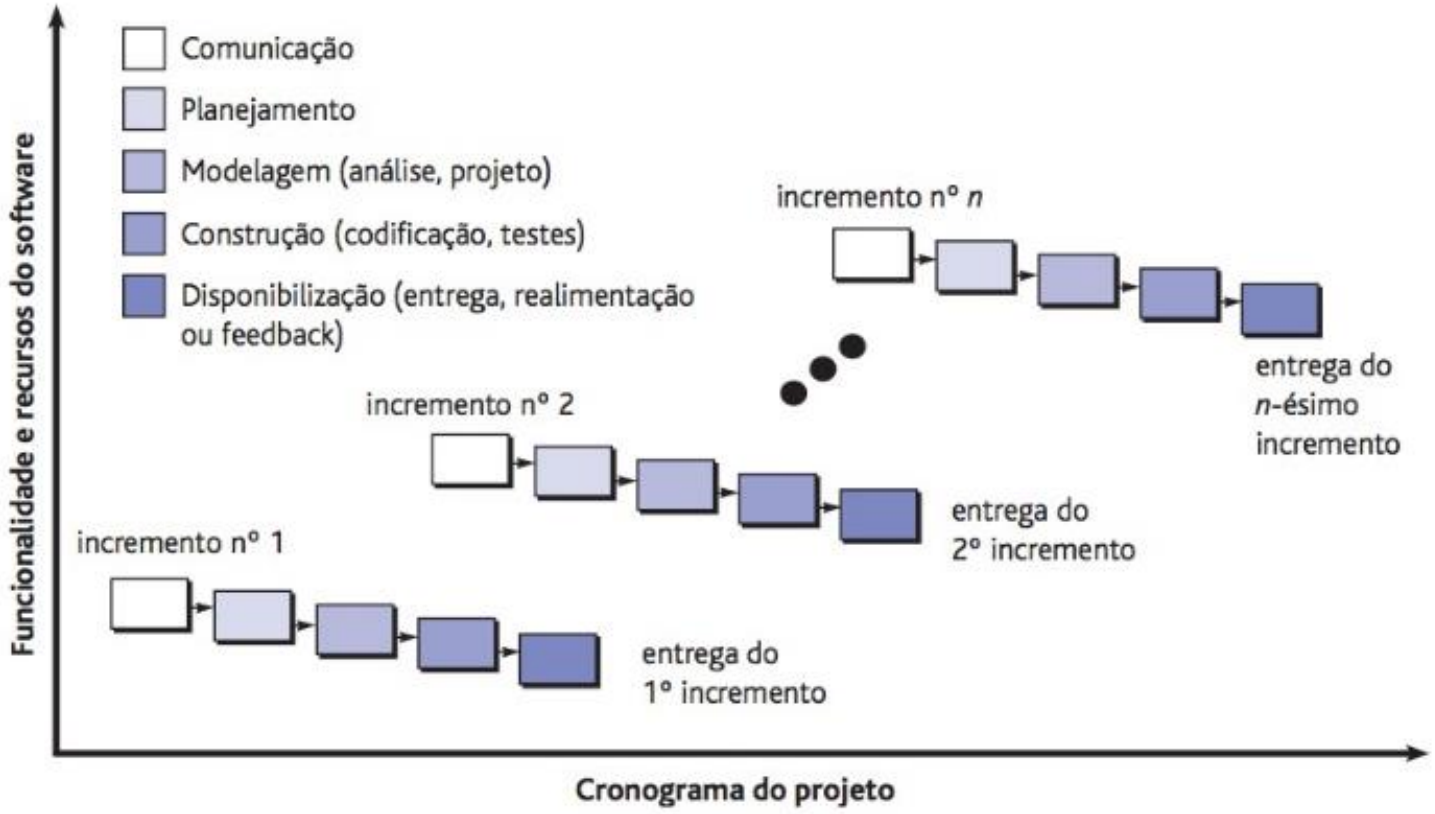
Modelo em V. Fonte: Pressman, 2016.

Modelo Incremental

- O modelo incremental combina os fluxos de processo linear e paralelo. O modelo incremental aplica sequencias lineares de forma escalonada, à medida que o tempo vai avançando. **Produzindo “incrementos” entregáveis do software.**

Fonte: Pressman, 2016

Modelo Incremental



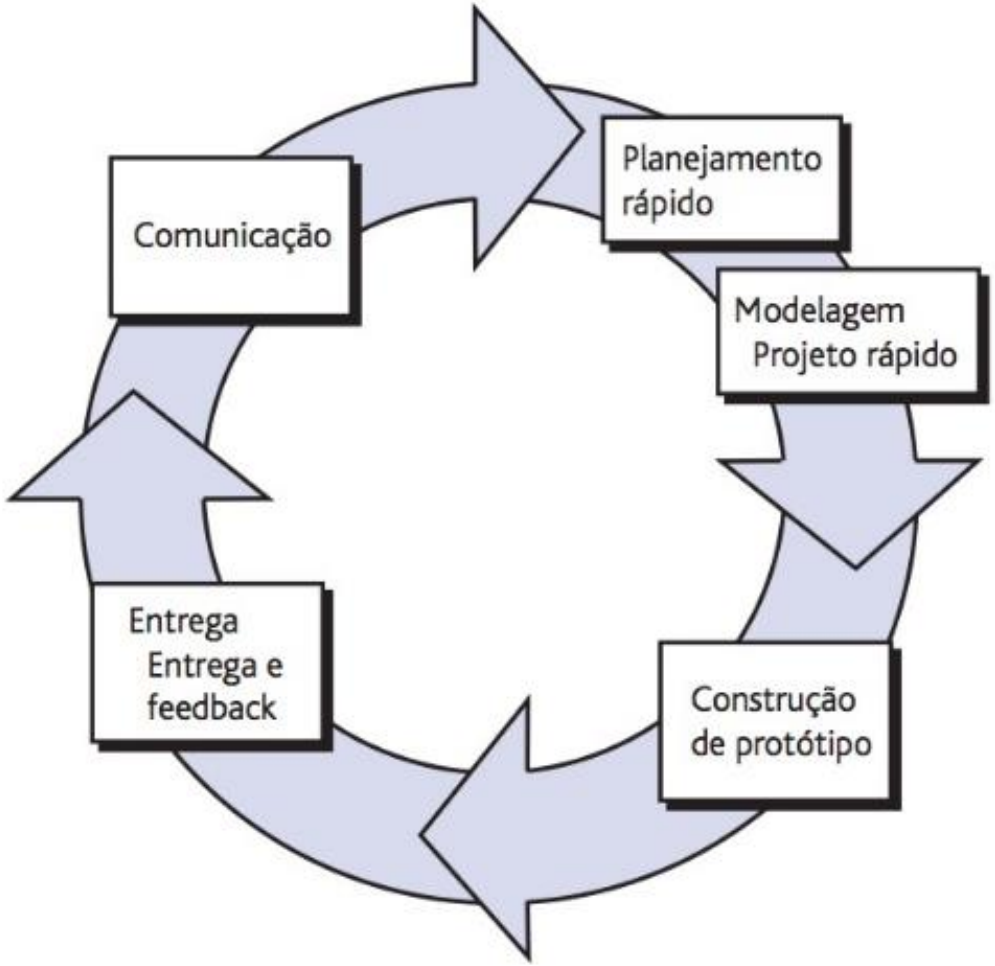
Modelo Incremental. Fonte: Pressman, 2016.

Prototipação

A prototipação pode ser utilizada como um modelo de processo isolado, comumente utilizada como uma técnica a ser implementada no contexto de qualquer um dos modelos de ciclo de vida.

Fonte: Pressman, 2016.

Prototipação



A Prototipação é usada no MVP (Minimum Viable Product)

Paradigma da Prototipação. Fonte: Pressman, 2016.

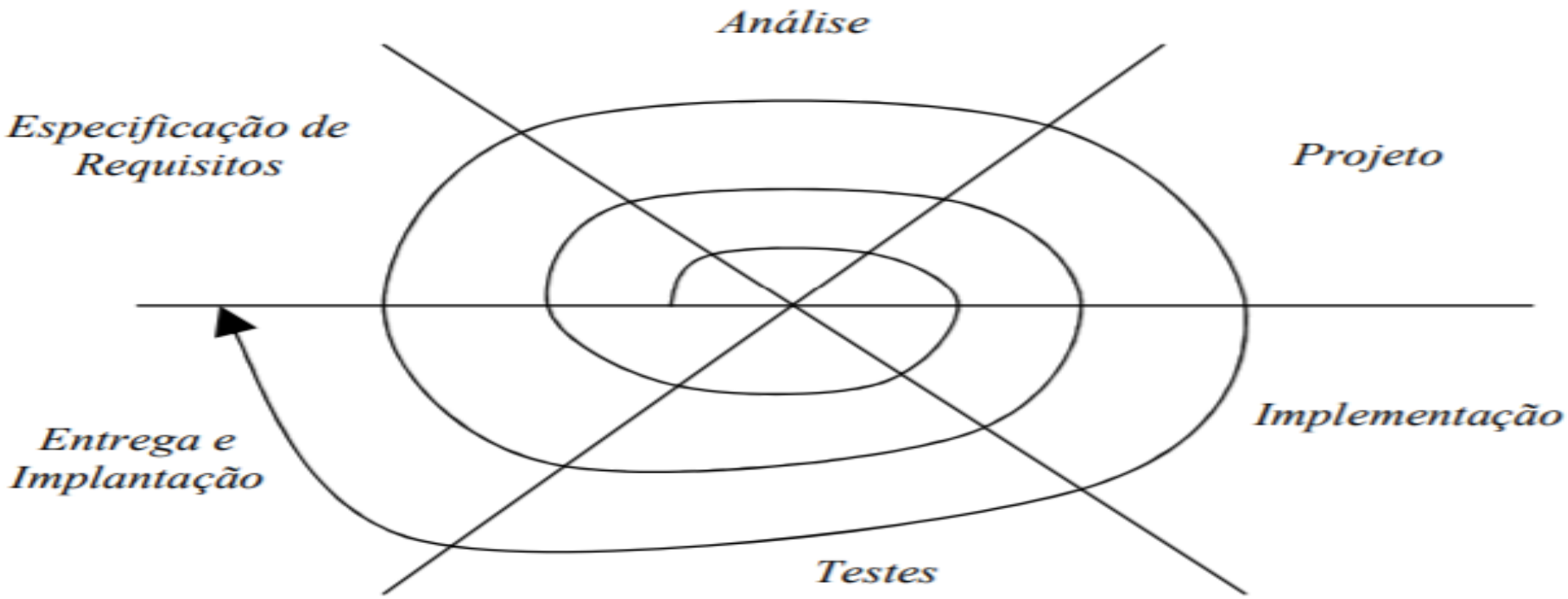
Modelos Evolucionários ou Evolutivos

Sistemas de software, como quaisquer sistemas complexos, evoluem ao longo do tempo.

Os requisitos, muitas vezes, são difíceis de serem estabelecidos ou mudam com frequência ao longo do desenvolvimento.

Os modelos evolucionários ou evolutivos lidam com incertezas e absorvem melhor as contínuas mudanças dos sistemas.

Modelo Espiral



Modelo em Cascata. Fonte: Falbo (2005).

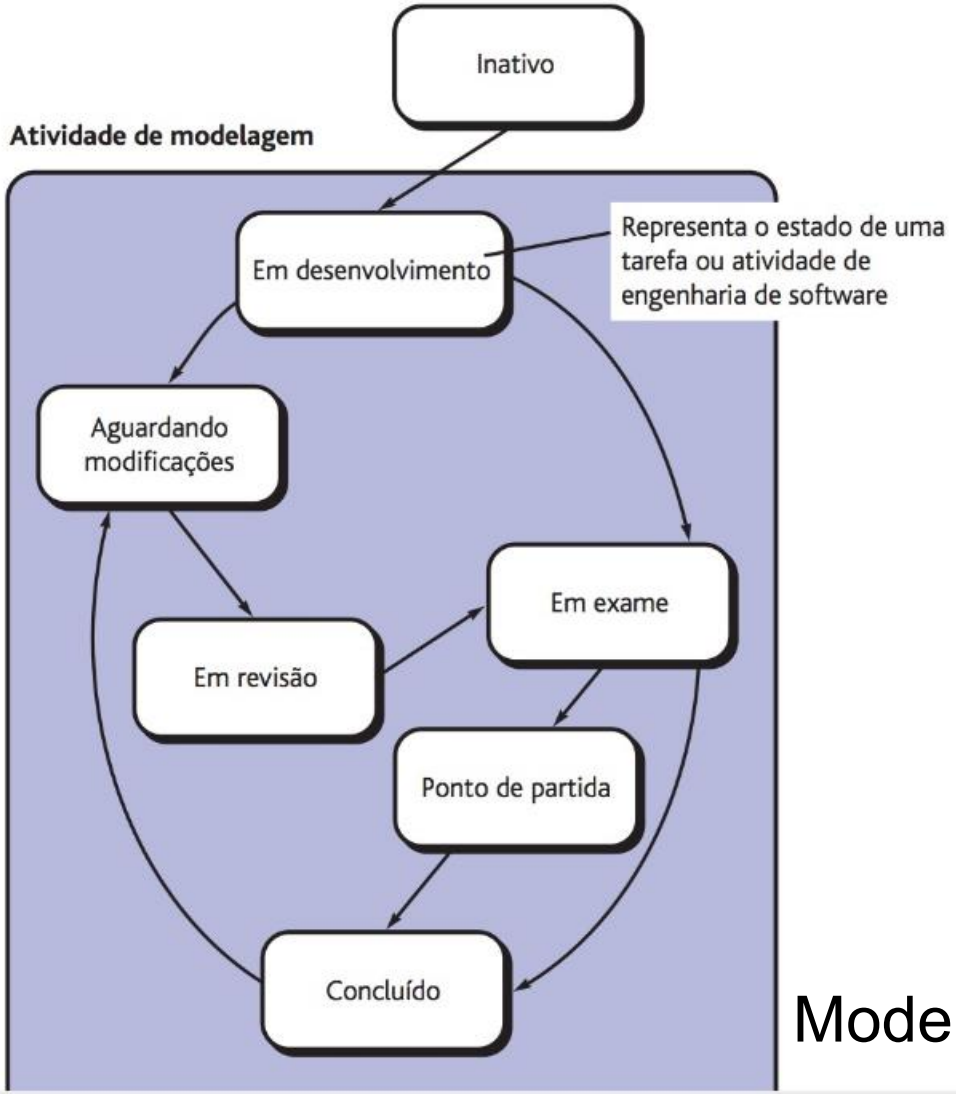
Modelo Espiral

- **O sistema é desenvolvido em ciclos, sendo que nos primeiros ciclos nem sempre todas as atividades são realizadas.**
- Por exemplo, o produto resultante do primeiro ciclo pode ser uma especificação do produto ou um estudo de viabilidade.
- As etapas subsequentes ao longo da espiral podem ser usadas para desenvolver protótipos, chegando progressivamente a versões operacionais do software, até se obter o produto completo

Modelo de Processo Especializado

- Incluem características de um ou mais dos modelos tradicionais, usado quando se opta por uma abordagem de engenharia de software especializada ou definida de forma restrita:
  - Desenvolvimento baseado em Componentes
  - O modelo de métodos formais
  - Desenvolvimento de software orientado a aspectos

Modelo de Processo Concorrente



Define uma rede de processos. Cada atividade, ação ou tarefa na rede existe simultaneamente. Eventos gerados disparam transições entre os estados associados a cada atividade.

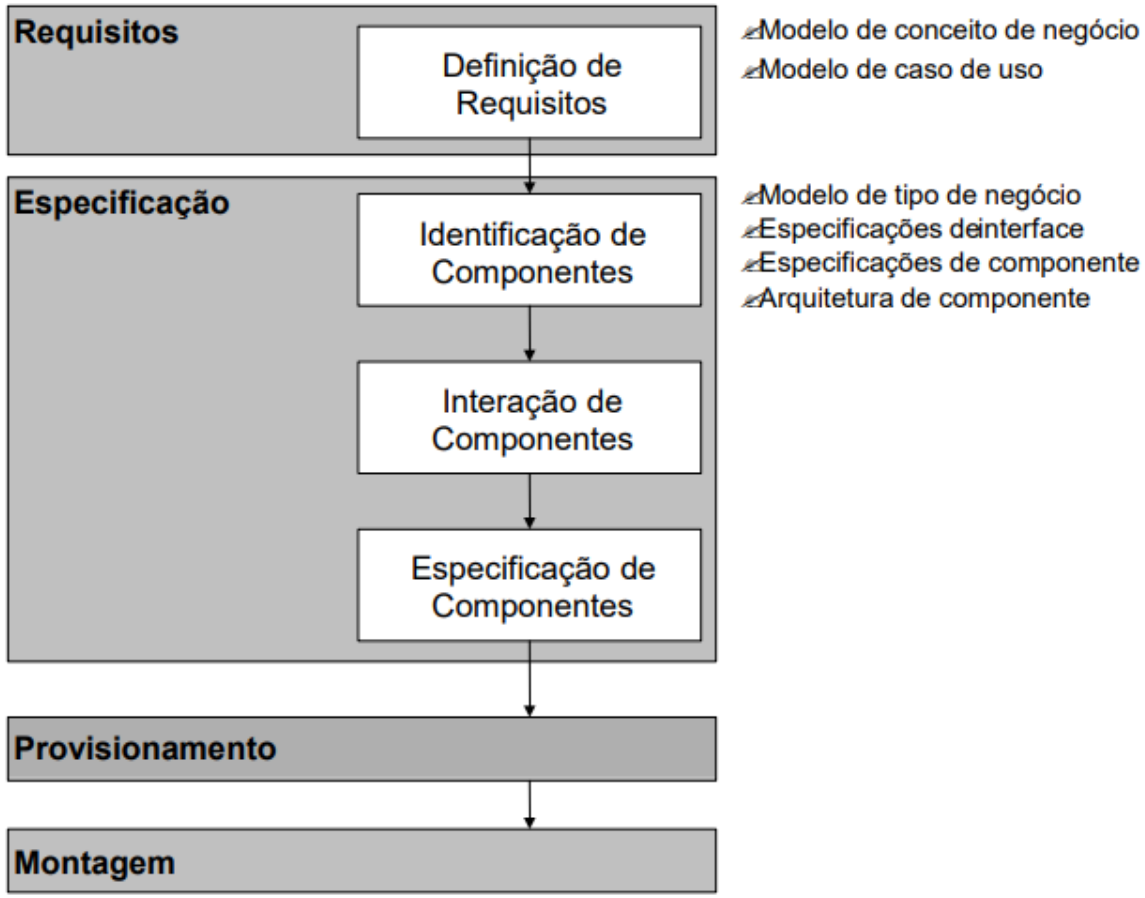
Modelo de Processo Concorrente. Fonte: Pressman, 2016.

Modelo Baseado em Componentes

PUC Minas

- **O modelo de desenvolvimento baseado em componentes incorpora muitas das características do modelo espiral.**
- Permitem que o componente seja integrado ao software a ser desenvolvido.
- Componentes de software comercial de prateleira

Modelo Baseado em Componentes



Modelo Baseado em Componentes. Fonte: https://edisciplinas.usp.br/pluginfile.php/130351/mod_resource/content/1/DSBC_UML_Components_1-2-3.pdf

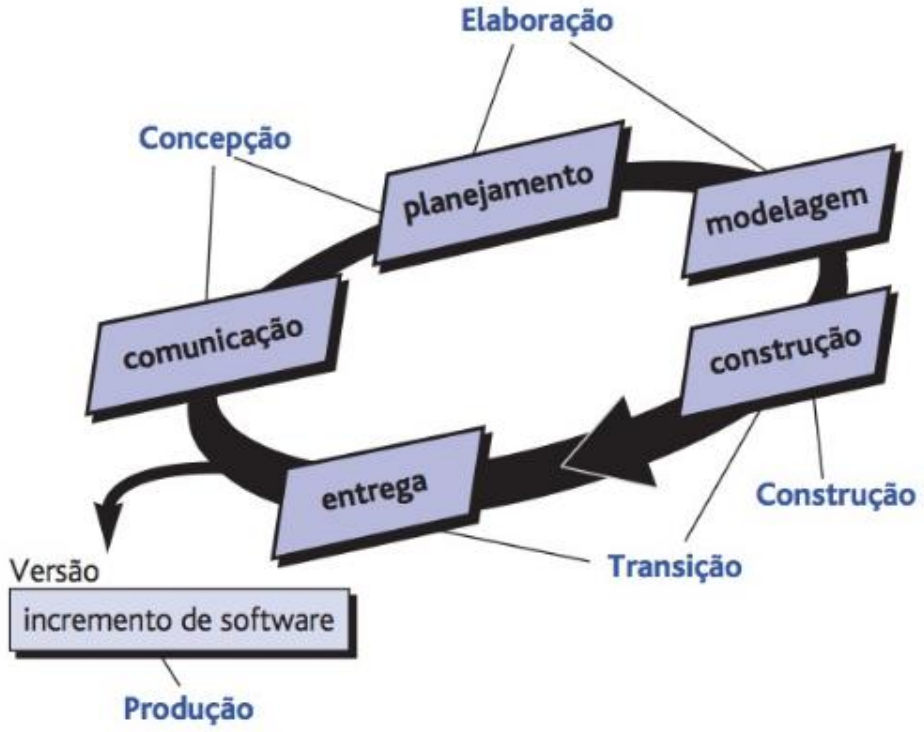
Desenvolvimento de Software Orientado a Aspecto

- **AOSD** (*aspect-oriented software development*), Desenvolvimento de Software Orientado a Aspecto que representam preocupações do cliente que transcendem várias funções, recursos e informações do sistema.
- Usam características tanto dos modelos de processo evolucionário quanto de processo concorrente.

Processo Unificado (PU)

- Aproveitar os melhores recursos e características dos modelos tradicionais de processo de software, mas caracterizando-os de modo a implementar muitos dos melhores princípios do desenvolvimento ágil de software.
- **Sugere um fluxo de processo iterativo e incremental, proporcionando a sensação evolucionária que é essencial no desenvolvimento de software moderno.**

Processo Unificado Racional (*RUP, Rational Unified Process*)



Modelo Processo Unificado. Fonte: Pressman, 2016.

Processo Unificado Racional (*RUP, Rational Unified Process*)

- A fase de concepção do PU inclui a atividade de comunicação com o cliente e a de planejamento.
- Identificar as necessidades de negócio para o software, propõe-se uma arquitetura rudimentar para o sistema e desenvolve-se um **planejamento para a natureza iterativa e incremental do projeto decorrente.**

Processo de Software

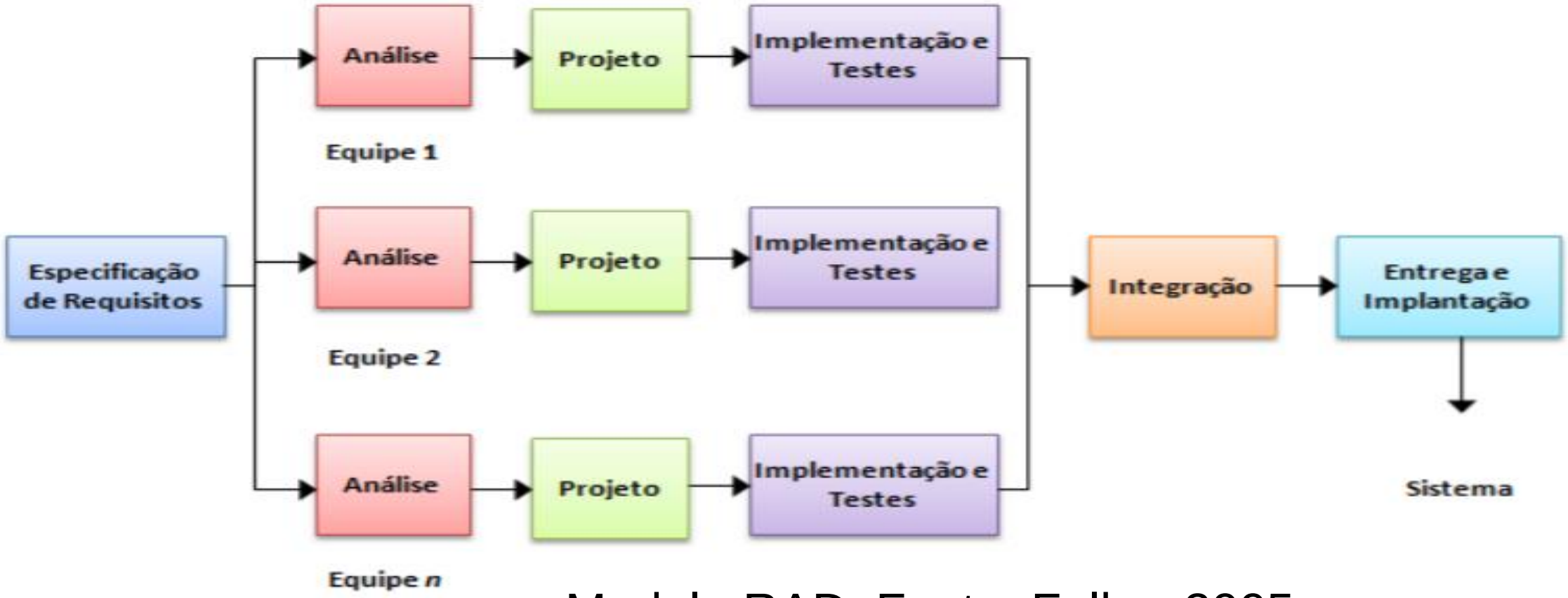
- **O Processo de Software Pessoal (*PSP, Personal Software Process*)** enfatiza a medição pessoal, tanto do artefato de software gerado quanto da qualidade resultante dele. Além disso, responsabiliza o profissional pelo planejamento de projetos e lhe permite controlar a qualidade de todos os artefatos de software desenvolvidos.
- **Processo de Software de Equipe (TSP, Team Software Process).** O objetivo do TSP é criar uma equipe de projetos "autodirigida".

Modelo RAD (*Rapid Application Development*)

PUC Minas

- O **Rapid Application Development (RAD)** é um modelo de processo de software incremental que enfatiza um ciclo de desenvolvimento curto.
- O Modelo RAD é uma adaptação, de alta velocidade, do modelo em cascata, no qual a agilidade é conseguida com o uso de uma abordagem de construção baseada em componentes.

Modelo RAD (*Rapid Application Development*)



Modelo RAD. Fonte: Falbo, 2005.

Modelos de Processo de Software - Resumo

PUC Minas

Cascata: organiza as atividades do processo de desenvolvimento de forma sequencial

Incremental: aplica sequencias lineares de forma escalonada, à medida de avanço do tempo

Evolucionário: modelo que absorvem as incertezas, pois trabalham com ciclos de desenvolvimento.

Espiral: O sistema é desenvolvido em ciclos, porém, nem todas as atividades de um ciclo necessitam ser realizadas

Prototipado: A prototipação pode ser utilizada como um modelo de processo isolado, pois é uma técnica a ser implementada no contexto de qualquer um dos modelos de ciclo de vida.

RAD (**Rapid Application Development)** - é um modelo de processo de software incremental que enfatiza um ciclo de desenvolvimento curto.

RUP (***Rational Unified Process)*** - Sugere um fluxo de processo iterativo e incremental, proporcionando a sensação evolucionária que é essencial no desenvolvimento de software moderno.

Manifesto Ágil

- Criado em fevereiro de 2001, por 17 profissionais, que já praticavam métodos ágeis como XP, DSDM, Scrum, FDD.
- Os valores Manifesto Ágil são:

Indivíduos e interação entre eles mais que processos e ferramentas;	**Software em funcionamento mais que documentação abrangente;**	**Colaboração do cliente mais que negociação de contratos;**	**Responder a mudanças mais que seguir um plano.**

Os doze princípios do Manifesto Ágil

PUC Minas

1 - Nossa maior prioridade é **satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado.**

2 - **Aceitar mudanças de requisitos**, mesmo no fim do desenvolvimento. Processos ágeis se adequam a mudanças, para que o cliente possa tirar vantagens competitivas.

3 - **Entregar frequentemente software funcionando**, de poucas semanas a poucos meses, com preferência à menor escala de tempo.

4 - Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.

5 - Construir projetos em torno de indivíduos motivados, dando a eles o ambiente e o suporte necessário e confiando neles para fazer o trabalho.

Os doze princípios do Manifesto Ágil

PUC Minas

6 - O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento **é por meio de conversa face a face**.

7 - **Software funcionando é a medida primária de progresso**.

8 **- Os processos ágeis promovem desenvolvimento sustentável**. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.

9 - **Contínua atenção a excelência técnica e bom design aumenta a agilidade**.

10 - **Simplicidade**: a arte de maximizar a quantidade de trabalho não realizado é essencial.

11 - **As melhores arquiteturas, requisitos e designs emergem de times auto-organizáveis**.

12 - Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

SCRUM

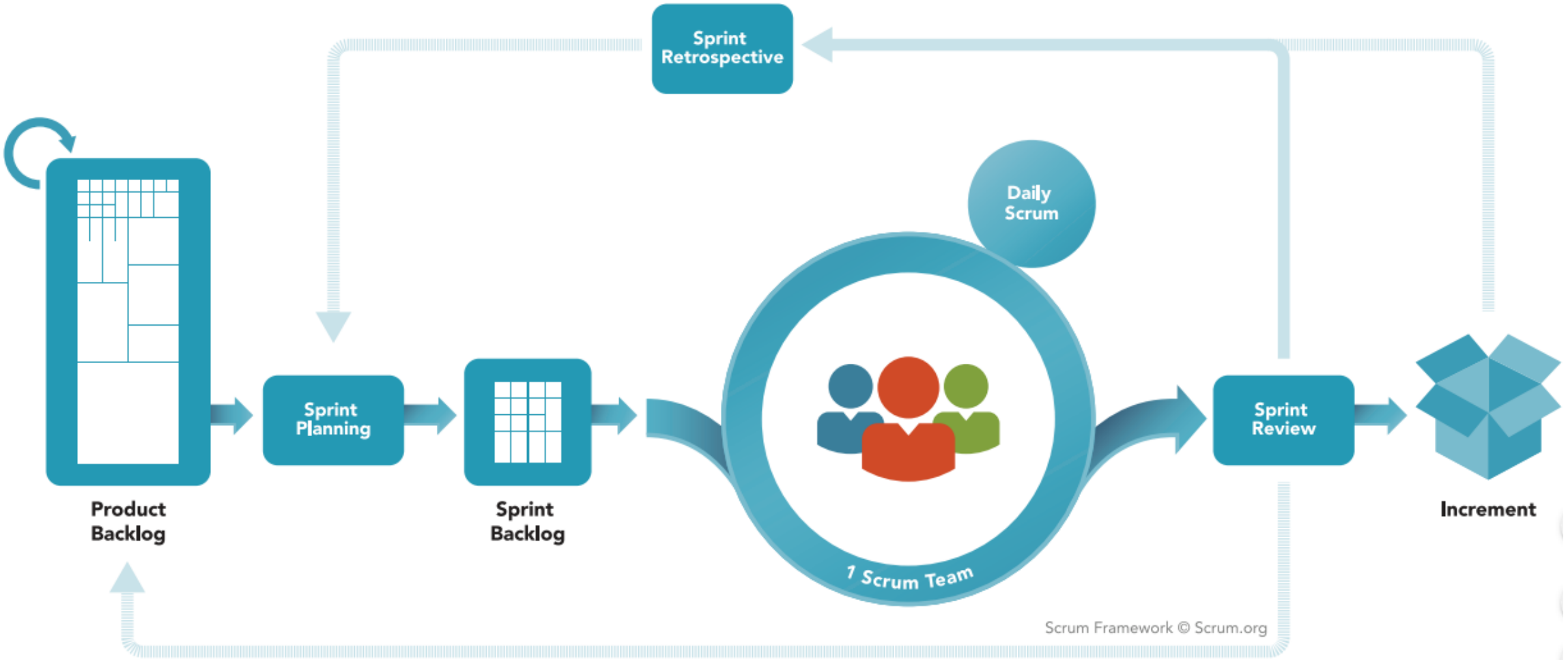
PUC Minas

"Um framework dentro do qual pessoas podem tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregam produtos com o mais alto valor possível."

(Schwaber & Sutherland, 2017)

Scrum Framework

PUC Minas



https://s3.amazonaws.com/scrumorg-website-prod/drupal/2016-06/ScrumFramework_17x11.pdf

Software como Produto ou Serviço

PUC Minas

Software como produto (*Software as a Product - SaaP*) modelo *on-premises*

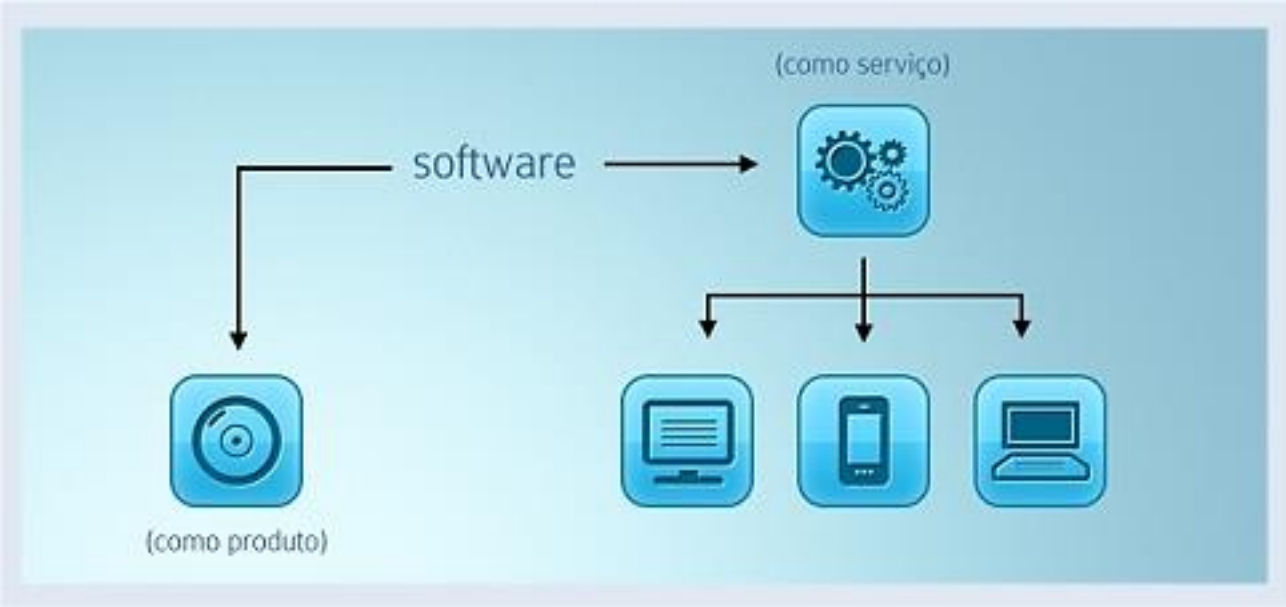
- Alto grau de customização para uma dada aplicação de software;
- Incerteza de demanda para uma funcionalidade específica;
- grande número de usuários para uma dada aplicação.

Software como serviço também é conhecido pelo acrônimo **SaaS** (*Software as a Service*).

- Clientes com alta incerteza de volume de demanda para uma dada aplicação de software;
- Clientes que são mais receptíveis à influência de organizações de pares;
- Clientes com alto custo de capital.

Fonte: http://www.creativante.com/new/index.php/2013-02-03-19-36-05/2013-02-04-18-19-01/8-software-como-produto-ou-como-servico

Software como Produto ou Serviço



Windows 10

Minitab

Microsoft Azure

Google Apps

salesforce

Fonte: http://www.creativante.com/new/index.php/2013-02-03-19-36-05/2013-02-04-18-19-01/8-software-como-produto-ou-como-servico

Aplicações mobile e Computação na Nuvem

- **IaaS:** É construída sobre a camada do data center e permite o fornecimento de componentes de armazenamento, hardware, servidores e rede. O cliente paga apenas pelo que utilizar, sua infraestrutura pode ser expandida ou reduzida dinamicamente conforme necessário, privilegiando a escalabilidade. Alguns exemplos são: **Amazon Elastic Cloud Computing e Simple Storage Service.**
- **PaaS:** A Plataforma como Serviço oferece um ambiente integrado e avançado para criação, teste e implantação de aplicativos personalizados. Entre os mais conhecidos, estão o ***Google App Engine, o Microsoft Azure e o Amazon Map Reduce/Simple Storage Service***.
- **SaaS:** O Software as a Service suporta uma distribuição de sistema com requisitos específicos. Nessa camada, os usuários podem acessar um aplicativo e informações remotamente pela internet e pagar apenas pelo que utilizam.

O **mobile cloud computing (MCC)** traz um novo paradigma aos aplicativos móveis, em que o **processamento** e o **armazenamento de dados** são movidos para **plataformas de computação poderosas e centralizadas.** Elas, por sua vez, são acessadas pela conexão sem fio, com base em um app nativo ou navegador da Web.

Fonte: https://www.ipsense.com.br/blog/entenda-como-o-mobile-cloud-pode-impactar-a-sua-empresa/#:~:text=O%20que%20%C3%A9%20mobile%20cloud,dos%20aparelhos%20para%20a%20nuvem.

Linhas de produtos de software

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.

“A reutilização de software se baseia no uso de conceitos, produtos ou soluções previamente elaboradas ou adquiridas para criação de um novo software, visando melhorar significativamente a qualidade e a produtividade”.

Reuso Horizontal: O reuso horizontal tem como meta a utilização de partes dentro de diferentes domínios de aplicação. Como exemplo, temos bibliotecas de funções matemáticas e manipulação de string, bibliotecas para construção de interfaces gráficas, entre outras.

Reuso Vertical: Reuso vertical é o que ocorre dentro de um mesmo domínio de aplicação. O objetivo é derivar um modelo genérico para ser usados dentro de um único domínio de aplicação na criação de novos sistemas.

Fonte: https://www.devmedia.com.br/reutilizacao-de-software-revista-engenharia-de-software-magazine-39/21956

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.
RUBIN, Kenneth**, Essential Scrum**: A Practical Guide to the Most Popular Agile Process, Pearson Education, 2013.
SCHWABER, Ken, SUTHERLAND, Jeff, **The Scrum Guide™** The Definitive Guide to Scrum: The Rules of the Game, November/2017, Disponível em <https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100> Acesso em 04 de fevereiro de 2020.