Processos de Software

Apostila: Processos de Software

Processos de Software

Introdução

Não existe engenharia de software sem as **pessoas**, e se temos pessoas, os processos são fundamentais para o desenvolvimento de softwares profissionais no **prazo** e com **qualidade**.

Veremos nesta apostila:

- Os modelos/tipos de processos existentes, assim como sua aplicabilidade;
- Entender o que são ferramentas CASE;
- Os modelos de referência para evolução dos processos de software (CMM).

Revisão: Processos

Um processo é um conjunto de passos, normalmente ordenados, que devem ser executados para atingir uma meta, um resultado. (a meta deve ser quantificável).

Insumo: O que é consumido no processo. (Pode ser a informação)

Papel: Proficiência + competências + responsabilidades. (Ex: Papel de PO).

Etapa: Divisão temporal do processo. (Ex: Atividade no BPM).

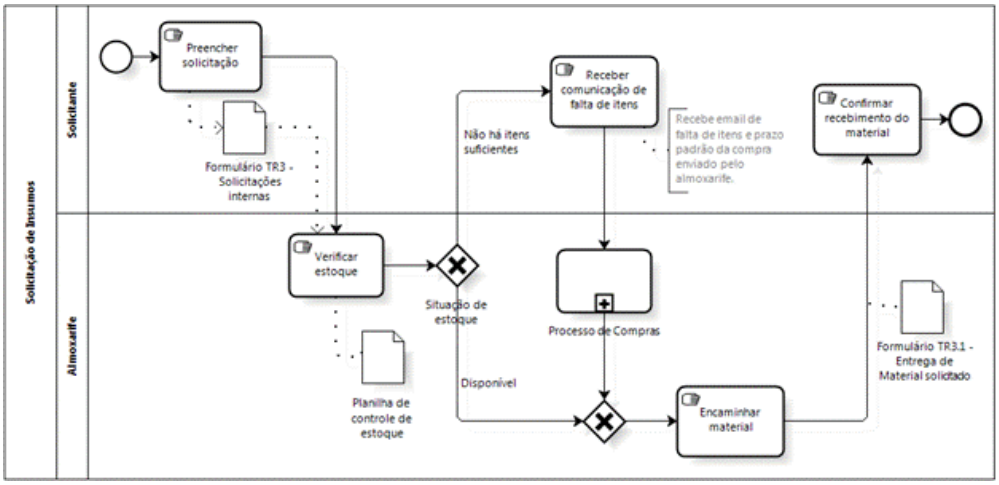
Produto: O que é produzido no final do processo. (quantificável)

Resultado: O que é produzido ao final de todos os processos. (Um processo pode fazer parte de outros processos ou atividades).

Estágio do processo	Descrição
Incompleto	Não realizado ou efetuado apenas parcialmente.
Efetuado	Realizou o trabalho necessário
Planejado	Está documentado por uma descrição/plano.
Gerido	Planejado, efetuado e executado conforme uma política.
Definido	Gerido e personalizado a partir dos processos da organização.

Adaptado de Wilson Pádua. 4ª edição.

Dentro do ecossistema de tecnologia, a notação BPMN é habitualmente utilizada para representar processos definidos.

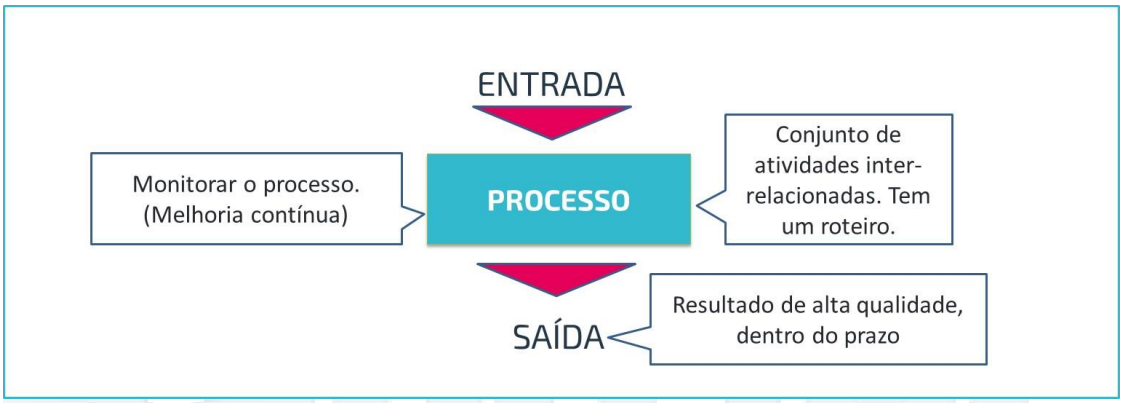


https://www.devmedia.com.br/introducao-ao-business-process-modeling-notation-bpmn/29892

O que é Processo de Software?

É um conjunto de atividades que leva à produção de um produto de software (Sommerville)

...um roteiro que ajude a criar um resultado de alta qualidade e dentro do prazo estabelecido. (Pressman)



A qualidade é o objetivo principal do processo de software. As empresas irão implementar e ou melhorar os seus processos buscando qualidade.

As operações de produção de software vêm evoluindo ao longo do tempo e no quadro abaixo podemos visualizar a evolução dos modelos de processos. Apesar de novos processos terem surgido, isso não significa que os processos "antigos" foram deixados de lado. Modelos tradicionais, também chamados de prescritivos continuam sendo utilizados por diversos segmentos de mercado. Podemos citar como exemplo, empresas de ERP (Enterprise Resource Planning) que utilizam o método cascata (waterfall) para a gestão de projetos de implantação.

Operações				
Artesanal	Artesanal	Fábrica	Fábrica + Outsourcing Integrado	Linha de produção de Software
Processos				
	Processos Proprietários	CMM	PMI, RUP, ISOs	XP, ASD (Adaptative Software Developmen)t
Plataformas				
Fortran, Assembler	Cobol, PL1 (IBM)	Natural, C, C++, Clipper	VB, Delphi	Java, .NET
Modelos de Processos				
Waterfall		Evolucionários	Especializados: Componentes, OO, UML	?Agile
1960-1970	**1970-1980**	**1980-1990**	**1990-2000**	**Século XXI**

Fernandes, Aguinaldo Aragon. Fábrica de Software. 1.ed. São Paulo: Atlas; 2007. (Cap. 1)

Modelos de Processos

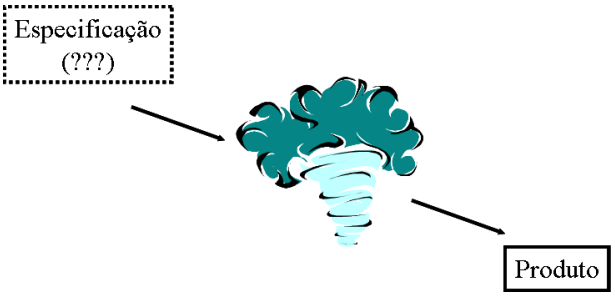
Vamos fazer um tour pelos principais modelos de processos, é sempre importante reforçar que na prática, as empresas não adotam um único modelo, também vale mencionar que muitos dos modelos são adaptados para as realidades das empresas.

Métodos tradicionais ou Modelos prescritivos são os que adotam a estratégia da previsibilidade. No caso do modelo cascata os requisitos são levantados o máximo possível antes de iniciar a fase seguinte. Mudar os requisitos quando estes métodos são utilizados, também dependem de um forte controle. Nestes métodos/modelos, o planejamento é concentrado na fase inicial. Não podemos afirmar que não há mudanças, mas sim que deve existir um forte controle e análise de impacto para cada mudança que ocorrer.

Os métodos ágeis, ao contrário, adotam a adaptabilidade, ou seja, os requisitos são levantados o suficiente para iniciar o trabalho, o planejamento é contínuo, as mudanças são recebidas e adaptadas a todo o tempo. Enquanto nos métodos tradicionais o foco está na documentação, artefatos e processos, nos métodos ágeis o foco está nas pessoas.

Modelo Codifica-remenda (Sem modelo)

O modelo dito por alguns autores como o mais utilizado (infelizmente), tem alto risco, é impossível de fazer gestão e não permite que compromissos sejam assumidos. A qualidade é baseada no acaso e apesar do redemoinho, não é um modelo espiral.

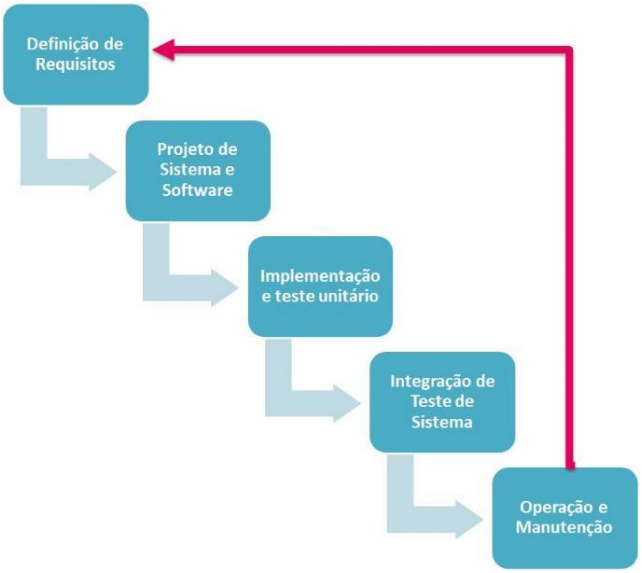


(Wilson de Pádua. 4ª edição)

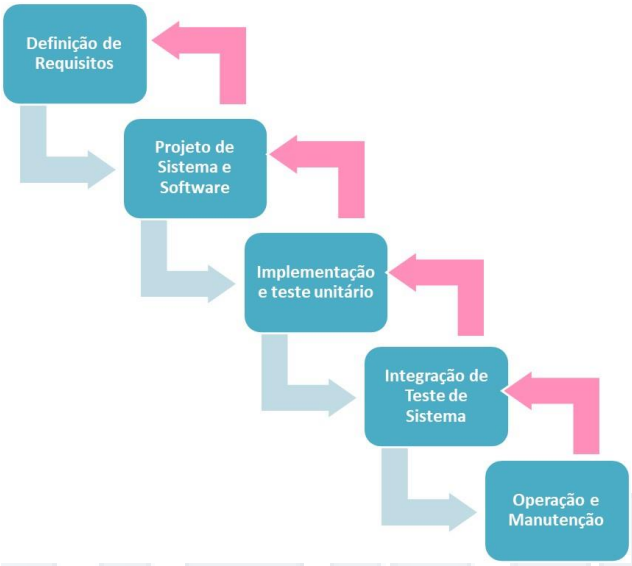
O "sem modelo" mais conhecido e difundido é o XGo-Horse.

Modelo Cascata (waterfall)

Modelo clássico, cada etapa é realizada é só ao final do processo os requisitos são alterados.



Modelo com realiamentação, cada etapa é realizada e uma etapa pode revisar etapas ou fases anteriores.



As empresas e equipes fazem adaptações no modelo, exemplos: sobreposição de fases, mudanças de requisitos, redução de burocracia, maior flexibilidade, contudo, vamos analisar o método tradicional.

Características do processo:

- Segue os processos gerais de Engenharia; é o mais antigo.
- É sequencial; costuma levar mais tempo.
- O resultado de cada fase consiste em uma validação formal de qualidade;
- Correções de Falhas e melhorias são realizadas posteriormente após todo o processo terminar;
- Gera muita documentação;
- É previsível; normalmente os requisitos estão bem definidos e mensuráveis;
- Utilizado quando há muito risco envolvido;

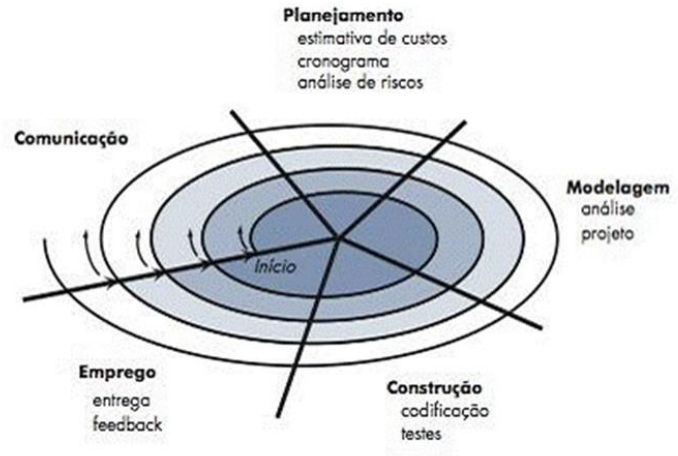
Desvantagens:

- Você precisa terminar um estágio para começar o próximo. Projetos reais raramente respeitam essa regra;
- Dificuldade de conseguir todos os requisitos de uma única vez;
- Falta de paciência do cliente em ver o resultado só no final de todo o ciclo;
- As entregas são normalmente enormes;
- Problemas de desenho, requisitos são detectados muito tarde;
- Os requisitos, após o término da fase de levantamento, são congelados.

Alguns exemplos em que o método/abordagem é utilizado(a):

- Desenvolvimento de projetos para governos, que vem de licitações, ou seja, os requisitos precisam ser muito bem definidos e controlados para a empresa de desenvolvimento não perder dinheiro, considerando que o preço está fixo.
- Projetos que podem gerar impacto alto em caso de falha, como software para indústria de saúde onde os requisitos são estáveis e muitos testes devem ser realizados para garantir que a entrega atenda os resultados determinados.
- Softwares muito complexos como ERP, onde sua implantação afetará toda a empresa e mudanças nos requisitos de um setor podem impactar outro setor. Uma alteração no requisito de um tributo pode impactar o software de gestão financeira, de suprimentos (compras) e até mesmo de RH.

Modelo de Processo Evolucionário – Espiral

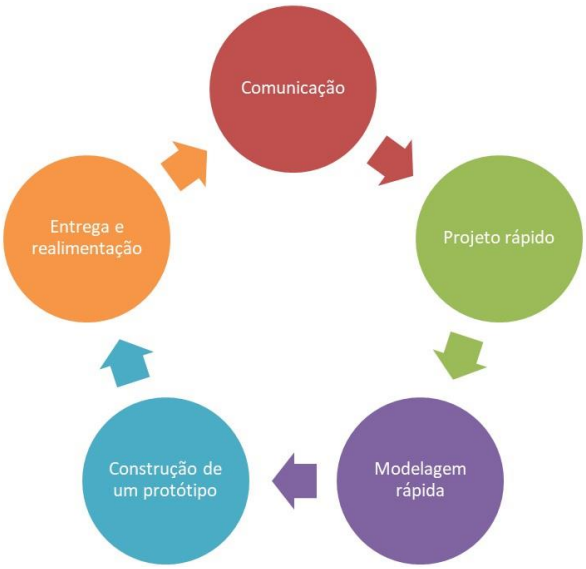


sommerville

O modelo espiral puro é pouco utilizado, mas o conceito é base para outras metodologias. Algumas características do modelo:

- O software deve ser desenvolvido em uma série de versões evolucionárias;
- Começa como um protótipo, mas à medida que o tempo passa são produzidas versões cada vez mais completas;
- Os custos e cronograma são ajustados de acordo com a evolução e feedback;
- O modelo pode continuar mesmo depois do software entregue;
- Os riscos são considerados a cada fase de evolução do projeto.
- Custo é alto;
- Difícil convencer o cliente que está tudo sob controle;

Prototipação e o modelo MVP são baseados no processo evolucionário.



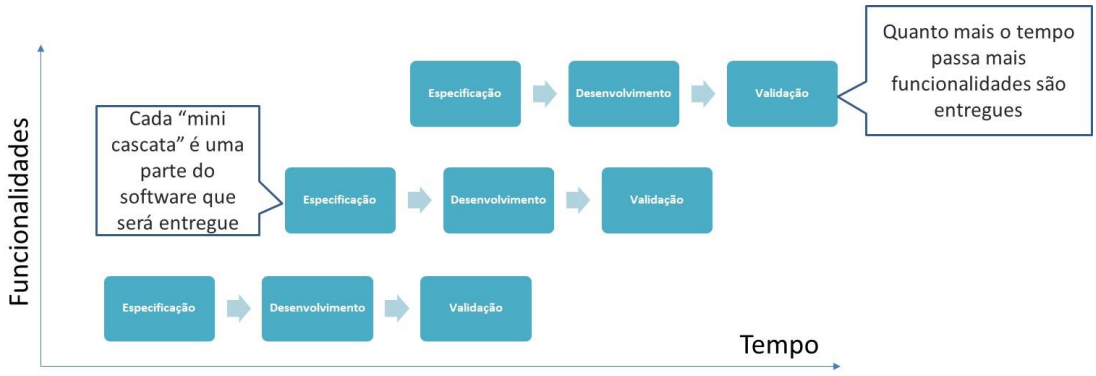
Os modelos de prototipação são aplicados quando:

- A empresa ou o desenvolvedor estão inseguros;
- Uma tecnologia muito recente (insegurança pela falta de conhecimento);
- Não há requisitos suficientes ou há muitas dúvidas sobre os requisitos iniciais.

Modelo de Processo Incremental

Na medida da evolução dos modelos as ideias começam a se relacionar. O modelo incremental é base para métodos ágeis, como o XP.

São fases de cada etapa do modelo incremental. **ESPECIFICAÇÃO, DESENVOLVIMENTO e VALIDAÇÃO.**



Características do processo incremental.

- Combina o uso do modelo cascata aplicado a forma interativa;
- Tem fluxos paralelos, o desenvolvimento ocorre com entregas sucessivas; podem existir equipes diferentes em cada incremento;
- Cada sequência gera uma entrega; O cliente devolve feedback a cada entrega;
- Não existe uma fase de manutenção, cada incremento tem novas solicitações, correções e mudanças;
- Os requisitos mais importantes são priorizados nas primeiras entregas; os incrementos iniciais podem ser usados como protótipos;
- Métodos ágeis como o XP são baseados neste processo;

Desvantagens:

- Precisa de um bom planejamento e desenho para que os incrementos possam ser "combinados"; você precisa desenhar o sistema todo antes de "quebrar em partes";
- O custo é mais alto que o processo cascata;
- Os requisitos comuns são mais difíceis de serem identificados;

Conclusão

Vimos os processos de software, sua importância, os modelos/tipos existentes e as aplicações destes modelos nos diversos ambientes corporativos.

Discutiremos em aula sobre os modelos tradicionais (prescritivos) atualmente "discriminados" nas atuais formas de gestão de processos/projetos de software, estes modelos ainda são relevantes e fundamentais para diversos segmentos de negócios.

Também complementar a esta apostila, deve ser analisado o conteúdo relacionado as abordagens ágeis de desenvolvimento de software.

Hands On

Realizar o QUIZ no Moodle. Processos de Software.

Material Complementar

Agile vs tradicional.

https://www.agile-minds.com/when-to-use-waterfall-when-agile/

Bibliografia

PRESSMAN, R. S. Engenharia de software. 7.ed. São Paulo: McGraw-Hill Brasil, 2011.

PFLEEGER, Shari Lawrence. Engenharia de Software: teoria e prática. Tradução de Dino Franklin. Revisão técnica Ana Regina Cavalcanti da Rocha. 2.ed. São

Paulo: Prentice Hall, 2004.

SOMMERVILLE, Ian. Engenharia de Software. Tradução de Selma Shin Shimizu Melnikoff; Reginaldo Arakaki; Edilson de Andrade Barbosa. Revisão técnica Kechi Hirama. 8. ed. São Paulo: Pearson Prentice Hall, 2007. 552 p.

Bibliografia Complementar

MAGELA, R. Engenharia de software aplicada – fundamentos. Alta Books

GONÇALVES, Daniel; FONSECA, Manuel J.; CAMPOS, Pedro. Introdução ao Design de Interfaces. 3. ed. Lisboa: FCA, 2017. 377 p.

PAULA FILHO, Wilson de Pádua, Engenharia de Software. 4.ed. Rio de Janeiro:LTC, 2019.