TRANS_
FORMAR
O PAÍS PELA
EDUCAÇÃO
É O QUE
NOS MOVE

ecossistema
ănima

Modelos, Métodos e Técnicas da Engenharia de Software

Aula 04

EXERCÍCIOS

E

Modelo Unificado de Processos (PU)

Prof. Luis Ybarra

Identificando Conjuntos de tarefas

Conjuntos de tarefas

Um conjunto de tarefas define o trabalho a ser feito para atingir os objetivos de uma ação de engenharia de software.

Por exemplo, levantamento (mais comumente denominado “levantamento de requisitos”) é uma importante ação de engenharia de software que ocorre durante a atividade de comunicação.

A meta do levantamento de requisitos é compreender o que os vários envolvidos esperam do software a ser desenvolvido. (Pressman, 2016).

Para um projeto pequeno e relativamente simples, o conjunto de tarefas para levantamento dos requisitos seria semelhante a este:

1. Fazer uma lista dos envolvidos no projeto.
2. Fazer uma reunião informal com todos os envolvidos.
3. Solicitar a cada envolvido uma lista com as características e funções necessárias.
4. Discutir sobre os requisitos e elaborar uma lista final.
5. Organizar os requisitos por grau de prioridade.
6. Destacar pontos de incertezas.

Para um projeto de software maior e mais complexo, é preciso usar um conjunto diferente de tarefas.
Esse conjunto pode incluir as seguintes tarefas de trabalho:

1. Fazer uma lista dos envolvidos no projeto.
2. Entrevistar separadamente cada um dos envolvidos para levantamento geral de suas expectativas e necessidades.
3. Fazer uma lista preliminar das funções e características, com base nas informações fornecidas pelos envolvidos.
4. Agendar uma série de reuniões facilitadoras para especificação de aplicações.
5. Realizar reuniões.
6. Incluir cenários informais de usuários como parte de cada reunião.
7. Refinar os cenários de usuários, com base no feedback dos envolvidos.

8. Fazer uma lista revisada dos requisitos dos envolvidos.
9. Empregar técnicas de implantação de funções de qualidade para estabelecer graus de prioridade dos requisitos.
10. Agrupar os requisitos de modo que possam ser entregues em incrementos.
11. Fazer um levantamento das limitações e restrições que serão aplicadas ao sistema.
12. Discutir sobre os métodos para validação do sistema.

ă

Esses dois conjuntos de tarefas atingem o objetivo do "levantamento de requisitos"; porém, são bem diferentes quanto ao seu grau de profundidade e formalidade.

A equipe de software deve escolher o conjunto de tarefas que possibilite atingir o objetivo de cada ação, mantendo, inclusive, a qualidade e a agilidade.

Casa segura – parte 1

Seleção de um modelo de processo, Parte 1

Cena: Sala de reuniões da equipe de engenharia de software da CPI Corporation, empresa (fictícia) que fabrica produtos de consumo para uso doméstico e comercial.

Atores: Lee Warren, gerente de engenharia; Doug Miller, gerente de engenharia de software; Jamie Lazar, membro da equipe de software; Vinod Raman, membro da equipe de software; e Ed Robbins, membro da equipe de software.

Conversa:

Lee: Recapitulando. Discuti bastante sobre a linha de produtos CasaSegura, da forma como a visualizamos no momento. Sem dúvida, temos muito trabalho a fazer para definir as coisas, mas eu gostaria que vocês começassem a pensar em como vão abordar a parte do software desse projeto.

Doug: Acho que fomos bastante desorganizados em nossa abordagem de software no passado.

Ed: Eu não sei, Doug, nós sempre conseguimos entregar o produto.

ă

Doug: É, mas não sem grande sofrimento, e esse projeto parece ser maior e mais complexo do que qualquer outro que já fizemos.

Jamie: Não parece assim tão difícil, mas eu concordo... a abordagem improvisada que adotamos em projetos anteriores não dará certo neste caso, principalmente se tivermos um cronograma muito apertado.

Doug (sorrindo): Quero ser um pouco mais profissional em nossa abordagem. Participei de um curso rápido na semana passada e aprendi bastante sobre engenharia de software... bom conteúdo. Precisamos de um processo aqui.

Jamie (franzindo a testa): Minha função é desenvolver programas, não ficar mexendo em papéis.

Doug: Dê uma chance antes de dizer não. Eis o que quero dizer. (Doug prossegue descrevendo a metodologia de processo descrita e os modelos de processo prescritivo apresentados até agora.)

Doug: De qualquer forma, parece-me que um modelo linear não é adequado para nós... ele presume que temos todos os requisitos antecipadamente e, conhecendo este lugar, isso é pouco provável.

Vinod: Isso mesmo, e parece orientado demais à tecnologia da informação… provavelmente bom para construir um sistema de controle de estoque ou algo parecido, mas certamente não é adequado para o CasaSegura.

Doug: Concordo.

Ed: Essa abordagem de prototipação me parece boa. Bastante parecida com o que fazemos aqui.

Vinod: Isso é um problema. Estou preocupado que ela não nos dê estrutura suficiente.

Doug: Não se preocupe. Temos várias opções e quero que vocês escolham o que for melhor para a equipe e para o projeto.

(Pressman, 2016).

Casa segura – parte 2

Seleção de um modelo de processo, Parte 2

Cena: Sala de reuniões do grupo de engenharia de software da CPI Corporation, empresa (fictícia) que fabrica produtos de consumo de uso doméstico e comercial.

Atores: Lee Warren, gerente de engenharia; Doug Miller, gerente de engenharia de software; Vinod e Jamie, membros da equipe de engenharia de software.

Conversa: (Doug descreve as opções do processo evolucionário.)

Jamie: Agora estou vendo algo de que gosto. Faz sentido uma abordagem incremental, e eu realmente gosto do fluxo dessa coisa de modelo espiral. Isso tem a ver com a realidade.

Vinod: Concordo. Entregamos um incremento, aprendemos com o feedback do cliente, reformulamos e, então, entregamos outro incremento. Também se encaixa na natureza do produto. Podemos colocar alguma coisa no mercado rapidamente e, depois, acrescentar funcionalidade a cada versão, digo, incremento.

Lee: Espere um pouco. Você disse que reformulamos o plano a cada volta na espiral, Doug? Isso não é tão legal; precisamos de um plano, um cronograma, e temos de nos ater a ele.

Doug: Essa linha de pensamento é antiga, Lee. Como o pessoal disse, temos de manter os pés no chão. Acho que é melhor ir ajustando o planejamento à medida que formos aprendendo mais e as mudanças forem sendo solicitadas. É muito mais realista. Para que serve um plano se não para refletir a realidade?

Lee (franzindo a testa): Suponho que esteja certo, porém... a alta direção não vai gostar disso... eles querem um plano fixo.

Doug (sorrindo): Então, você terá que reeducá-los, meu amigo.

(Pressman, 2016).

EXEMPLO – ESCOLHA DO MODELO

A empresa Xpto tem trabalhado em um projeto com o objetivo desenvolver um sistema de um *e-commerce*.

1. Caso opte por utilizar o modelo cascata, reunir-se-á com o cliente uma única vez, documentará os requisitos que o cliente solicitou e os apresentará para validação do cliente.

Uma vez aceitos os requisitos, o *e-commerce* é criado e o cliente apenas o visualiza após todos os testes terem sido executados e o projeto encerrado.

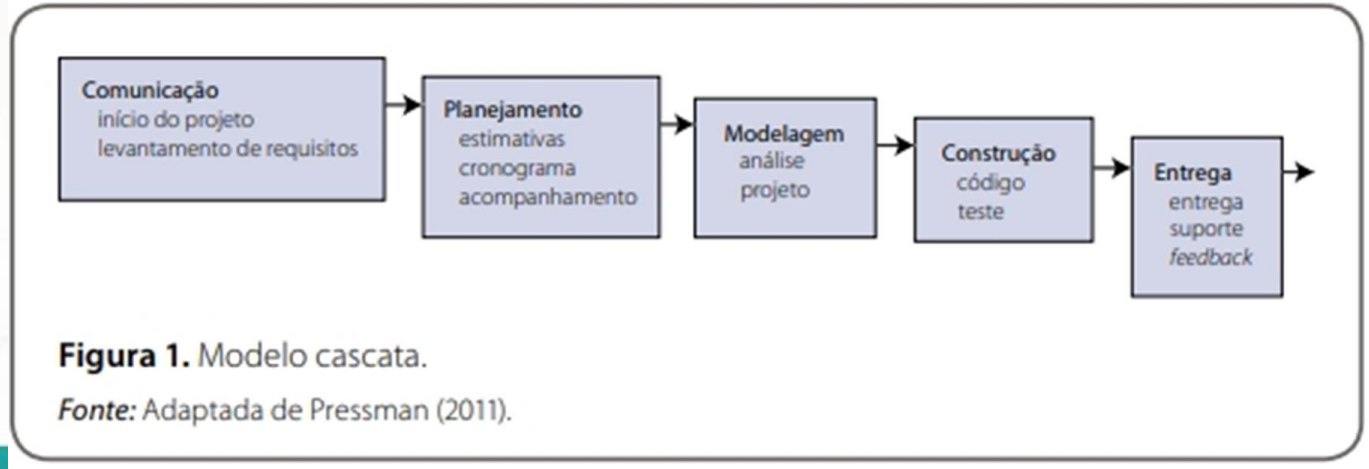
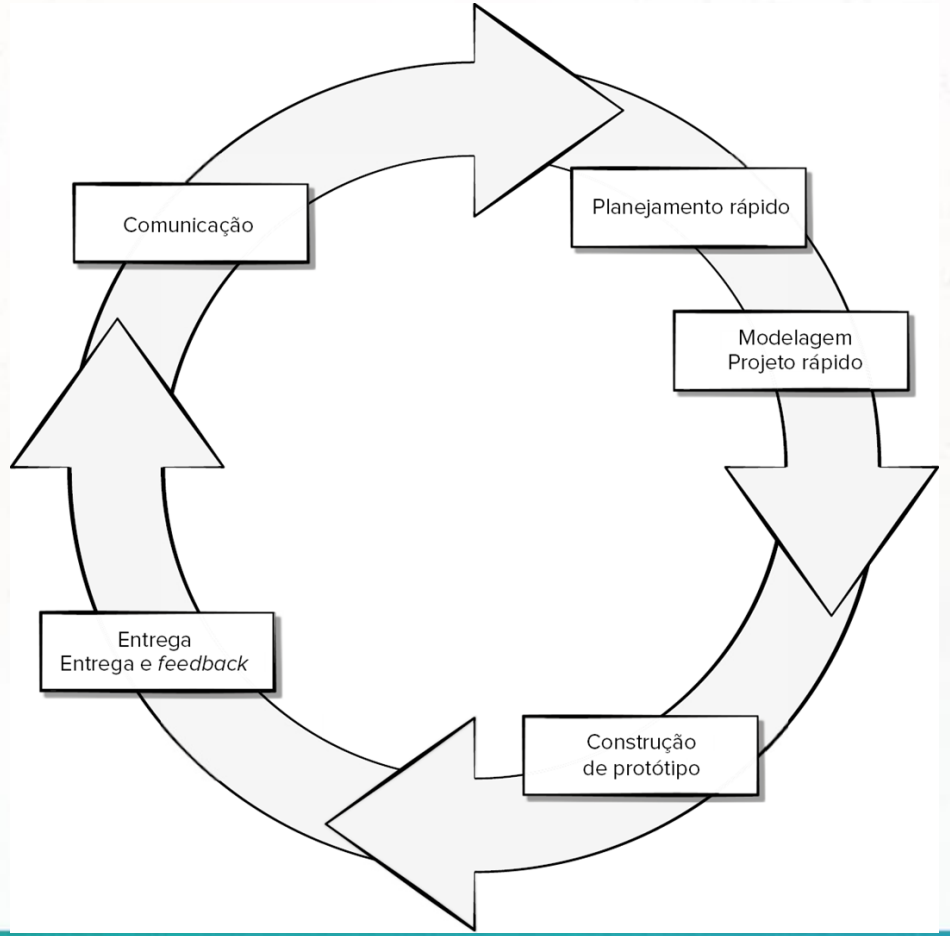


Figura 1. Modelo cascata.

Fonte: Adaptada de Pressman (2011).

(Morais, 2017)

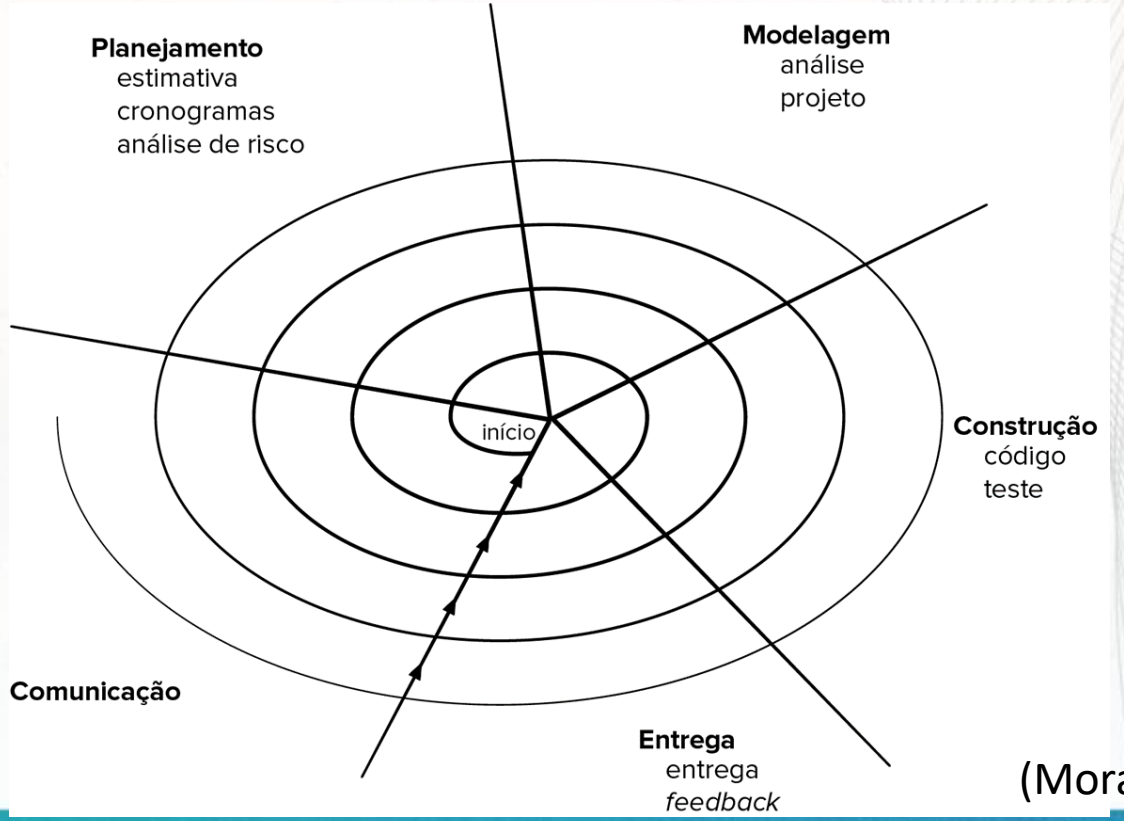
2. <u>Caso se opte pelo modelo prototipação</u>, antes de iniciar o desenvolvimento, os profissionais criarão as telas do e-commerce e as mostrarão para o cliente. O cliente deverá interagir como se estivesse usando o software real, para, então, aprovar o desenvolvimento ou sugerir alterações.



(Morais, 2017)

3. <u>No caso de utilizar o modelo espiral</u>, será feito inicialmente o protótipo de uma tela de cadastro dos produtos, que será avaliada pelo cliente e, depois, desenvolvida. Em seguida, será feita uma tela para compras, também avaliada pelo cliente e, então, desenvolvida, e assim sucessivamente, prototipando cada parte do programa antes de ele ser desenvolvido. Além da prototipação, a cada etapa serão analisados os riscos possíveis em cada etapa,

por exemplo, ter um estagiário que está se formando e que precisará deixar a equipe, uma mudança de versão de alguma tecnologia durante o processo, entre outros riscos previsíveis que, nesse modelo, são analisados previamente a cada ciclo.



(Morais, 2017)

Modelo de Processo Unificado (PU)

Sob certos aspectos, o Processo Unificado (PU) [Jac99] é uma tentativa de 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.

O Processo Unificado reconhece a importância da comunicação com o cliente e de métodos racionalizados para descrever a visão do cliente sobre um sistema (os casos de uso).

Ele enfatiza o importante papel da arquitetura de software e “ajuda o arquiteto a manter o foco nas metas corretas, como compreensibilidade, confiança em mudanças futuras e reutilização” [Jac99].

Ele sugere um fluxo de processo iterativo e incremental, proporcionando a sensação evolucionária que é essencial no desenvolvimento de software moderno.

(Pressman, 2016).

BIBLIOGRAFIA

Pressman, Roger S. Engenharia de software : uma abordagem profissional [recurso eletrônico] / Roger S. Pressman, Bruce R. Maxim ; [tradução: João Eduardo Nóbrega Tortello ; revisão técnica: Reginaldo Arakaki, Julio Arakaki, Renato Manzan de Andrade]. – 8. ed. – Porto Alegre : AMGH, 2016.

Morais, Izabelly Soares de. Engenharia de software [recurso eletrônico] / Izabelly Soares de Morais, Aline Zanin ; revisão técnica : Jeferson Faleiro Leon. – Porto Alegre : SAGAH, 2017.

PRESSMAN, Roger; MAXIM, Bruce. Engenharia de Software. Uma abordagem profissional. 8a. Ed. Bookman, 2016. https://integrada.minhabiblioteca.com.br/#/books/978580555349/cfi/3!/4/2@100:0.00

SOMMERVILLE, Ian. Engenharia de Software. 9. ed. São Paulo: Pearson Prentice Hall, 2011. https://bv4.digitalpages.com.br/?term=engenharia%2520de%2520software&searchpage=1&filtro=todos&from=busca&page=_14&section=0#/legacy/276

LARMAN, Craig. Utilizando UML e padrões: uma introdução à análise e ao projeto orientados a objetos e desenvolvimento iterativo. 3. ed Porto Alegre: Bookman, 2007. https://integrada.minhabiblioteca.com.br/#/books/978857800476/cfi/0!/4/2@100:0.00

Prikladnicki Rafael, Willi Renato, Milani Fabiano. Métodos ágeis para desenvolvimento de software / Organizadores, Rafael Prikladnicki, Renato Willi, Fabiano Milani. – Porto Alegre : Bookman, 2014.

IFSC. Ciclo de Vida Iterativo e Incremental. Wiki Instituto Federal de Santa Catarina, São José, out. 2006. Disponível em: <https://wiki.sj.ifsc.edu.br/wiki/index.php/Ciclo_de_Vida_Iterativo_e_Incremental>. Acesso em: 31 ago. 2017.