Pós-Graduação em Engenharia de Software

Disciplina
Processo de Software
24h
04.11 a 09.12

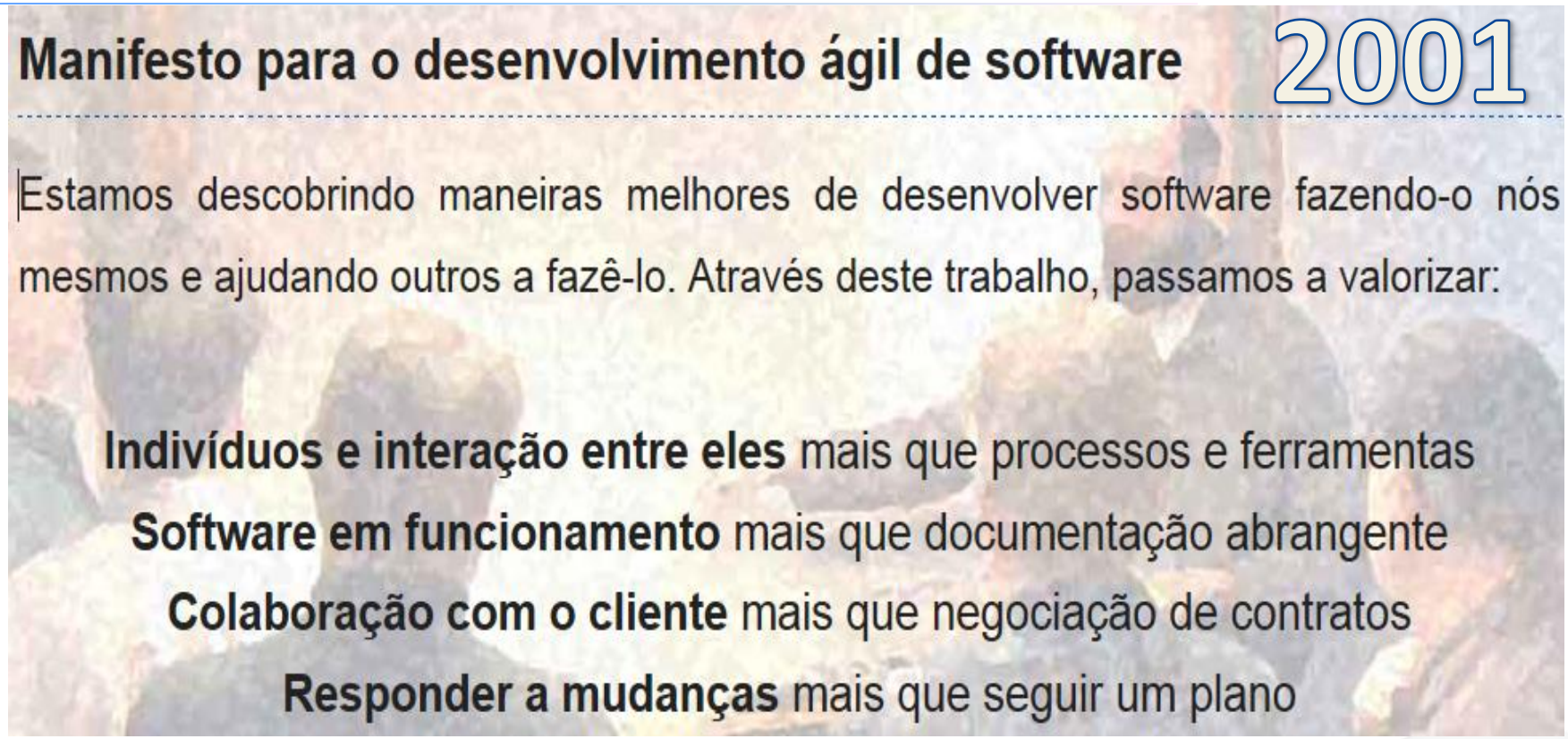
Segundas-feiras

Revisão

Metodos Ágeis

*A coisa mais importante sobre os **Métodos Ageis** é que **não existe processo, existem times ageis.***
Os processos descritos tem a ver com ambientes onde equipes podem ser ageis.

Manifesto Ágil



http://manifestoagil.com.br/

Extreme Programming

Extreme Programming

Práticas XP

- ✓ Time Coeso (Whole Team)
- ✓ Testes de Aceitação (Customer Tests)
- ✓ Ritmo Sustentável (Sustainable Tests)
- ✓ Reuniões em pé (Stand-up Meeting)
- ✓ Posse Coletiva (Collective Ownership)
- ✓ –O código fonte não tem dono
- ✓ Programação em Pares (Pair Programming)

Questions

Faculdade IMPACTA TECNOLOGIA

Agenda Aula-5

Processo de Software

- ✓ História
- ✓ Conceitos da Engenharia de Software
- ✓ Processo de Software
- ✓ Modelos de Processo
- ✓ Introdução Métodos Ageis
- ➢ Introdução Scrum
- ➢ Introdução UP
- ✓ Introdução RUP

Faculdade IMPACTA TECNOLOGIA

SCRUM

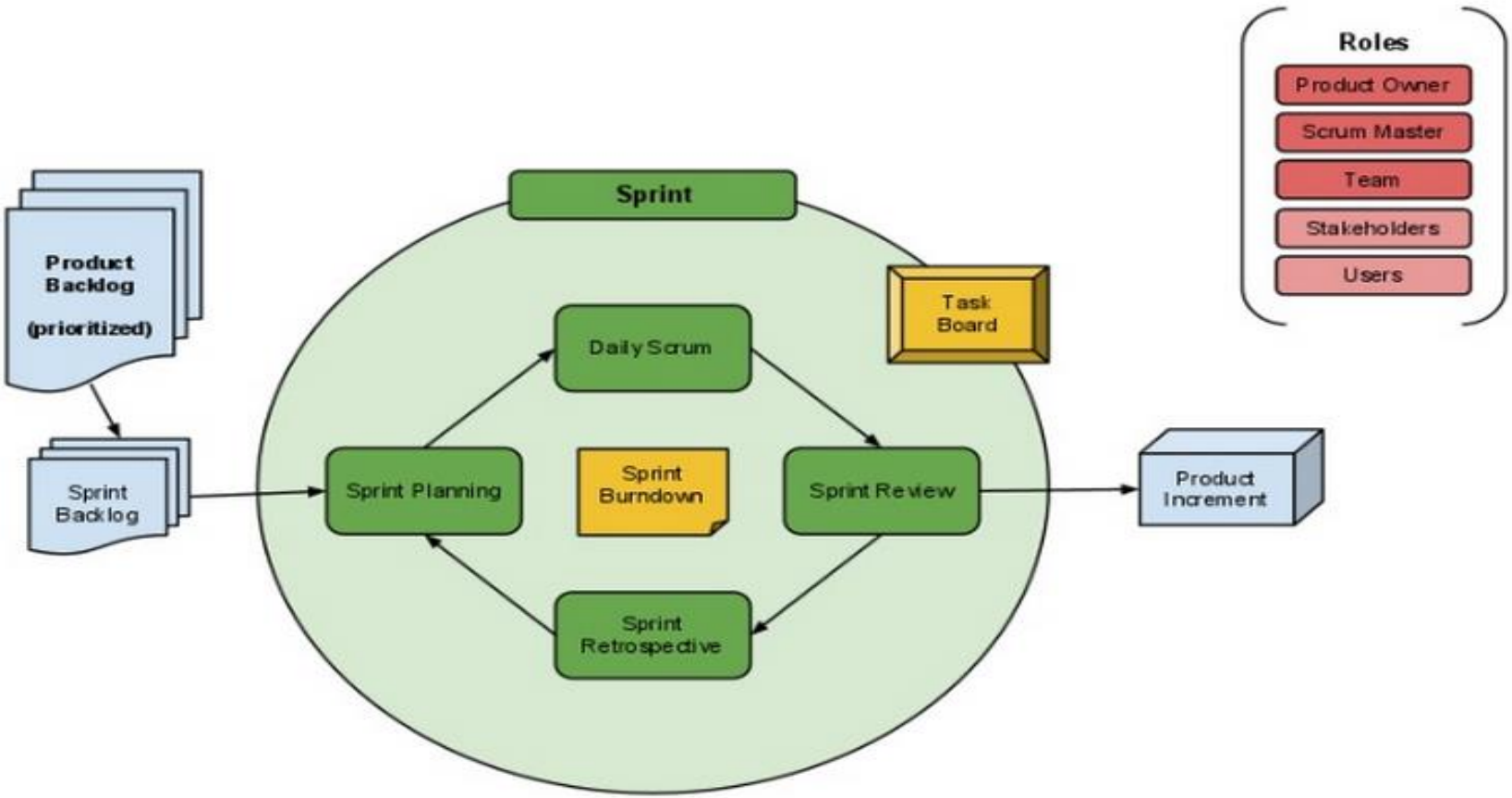
https://www.scrum.org/

Definição do Scrum

Scrum(subs): 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.

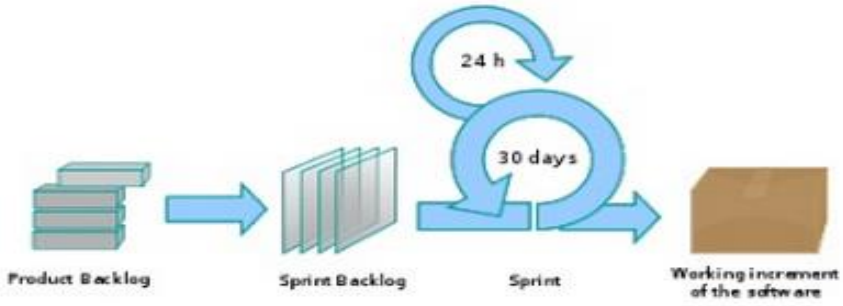
SCRUM

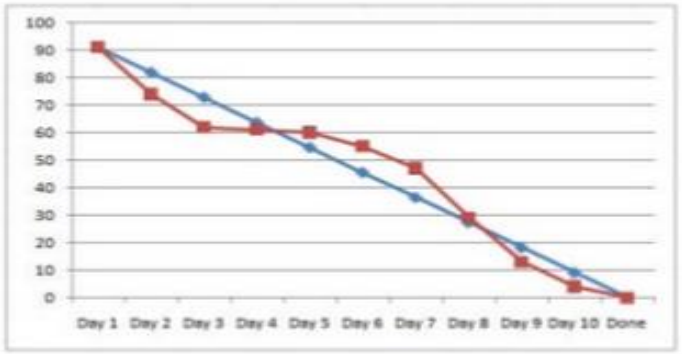
Scrum


Roles
Product Owner
Scrum Master
Team
Stakeholders
Users
Sprint
Product Backlog
(prioritized)
Sprint Backlog
Daily Scrum
Task Board
Sprint Planning
Sprint Burndown
Sprint Review
Product Increment
Sprint Retrospective

SCRUM

Scrum





1. Roles

- Product Owner
- ScrumMaster
- Team

2. Ceremonies

- Daily scrum
- Sprint planning
- Sprint review
- Sprint retrospective

3. Artifacts

- Product backlog, sprint backlog, user stories
- Burn-down chart

Scrum é:

- ✓ Leve
- ✓ Simples de entender
- ✓ Extremamente difícil de dominar

Scrum não é um processo em si, é um Framework

O *framework* Scrum consiste nos times do Scrum associadas a papéis, eventos, artefatos e regras.

Cada componente dentro do *framework* serve a um propósito específico e é essencial para o uso e sucesso do Scrum.

Scrum suporta e incentiva algumas necessidades puramente humanas:

- ✓ Pertencimento
- ✓ Aprendizado
- ✓ Fazer e produzir
- ✓ Criatividade
- ✓ Desenvolver-se
- ✓ Evoluir
- ✓ Interagir com outras pessoas

Utiliza os artefatos, ferramentas conforme a CULTURA organizacional, ou seja, não impoem ferramentas, tecnologias, documentos e artefatos.

Teoria do Scrum

Scrum é fundamentado nas teorias empíricas de controle de processo, ou empirismo.

O empirismo afirma que o conhecimento vem da experiência e de tomada de decisões baseadas no que é conhecido.

O Scrum emprega uma abordagem iterativa e incremental para aperfeiçoar a previsibilidade e o controle de riscos.

SCRUM

- ✓ Scrum é a forma como as equipes trabalham juntas para desenvolver um produto.
- ✓ O desenvolvimento de um produto ocorre em pequenas partes, onde cada parte é feita com base na parte anterior já pronta.
- ✓ Scrum fornece um conjunto de regras apenas com o objetivo de estruturar e focar as equipes na solução de problemas

Teoria do Scrum

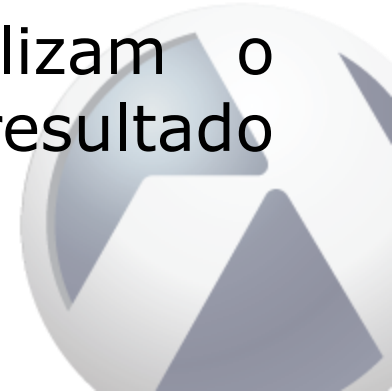
Três pilares apoiam a implementação de controle de processo:

- ✓ Transparência
- ✓ Inspeção
- ✓ Adaptação

Transparência

Aspectos do processo visíveis para compartilharem o mesmo entendimento
Por exemplo:

- ✓ Uma linguagem comum referindo-se ao processo deve ser compartilhada por todos os participantes; e,
- ✓ Uma definição comum de "Pronto" deve ser compartilhada por aqueles que realizam o trabalho e por aqueles que aceitam o resultado do trabalho.



Inspeção

- ✓ Os usuários Scrum devem, frequentemente, inspecionar os artefatos Scrum e o progresso em direção a detectar variações.
- ✓ As inspeções são mais benéficas quando realizadas por inspetores especializados no trabalho a se verificar.

Adaptação

Desvios nos aspectos de um processo ou que o produto resultante será inaceitável, processo e produto devem ser ajustados.

SCRUM

O Scrum prescreve quatro Eventos formais, contidos dentro da Sprint, para inspeção e adaptação:

- ✓ Reunião de planejamento da Sprint
- ✓ Reunião diária
- ✓ Reunião de revisão da Sprint
- ✓ Retrospectiva da Sprint

SPRINT (corrida, largada) é a unidade básica de desenvolvimento em Scrum.

SPRINTS duram entre uma semana e um mês, é um esforço dentro de uma caixa de tempo (ou seja, restrito a uma duração específica)

Cada sprint é precedido por uma reunião de planejamento (**Sprint Planning**), onde as tarefas para o sprint são identificadas e um compromisso estimado para o objetivo do sprint é definido

O progresso é revisto e lições para os próximos sprints são identificadas na **reunião de revisão ou de retrospectiva**,

SCRUM

Scrum define uma estrutura com 3 papéis

- ✓ ***Product Owners*** - determina o que precisa ser construido/desenvolvido (30 dias-caixa de tempo)
- ✓ ***Scrum Masters*** – garante que o processo seja executado
- ✓ ***Development Teams*** – constroe / desenvolve e demonstra o que foi feito.

O Product Owner

- ✓ Dono do produto, é o responsável por maximizar o valor do produto e do trabalho do Time de Desenvolvimento.
- ✓ Define as necessidades do produto
- ✓ Dá o direcionamento, de negócio, de requisitos
- ✓ Prioriza funcionalidades
- ✓ Aceita / rejeita o “entregável”
- ✓ É uma pessoa e não um comitê.
- ✓ É a único responsável por gerenciar o Backlog do Produto.

Scrum Master

- Foco no PROCESSO
  - Aplicação dos valores e práticas
- Remove Obstáculos
- Escudo para interferências externas

O Scrum Master

- ✓ É responsável por garantir que o Scrum seja entendido e aplicado.
- ✓ Ajuda aqueles que estão fora do Time, a entender quais as suas interações com o Time e se são úteis e quais não são.

O Scrum Master

Apoiar o Product Owner

- ✓ No gerenciamento do Backlog do Produto;
- ✓ Na comunicação da visão, objetivo e itens do Backlog do Produto para o Time de Desenvolvimento;
- ✓ Na prática da agilidade;
- ✓ Na gestão dos eventos

O Scrum Master

Liderar o Time de Desenvolvimento

- ✓ Ensinar e liderar o Time na criação de produtos de alto valor;
- ✓ Remover impedimentos para o progresso do Time de Desenvolvimento;
- ✓ Facilitar os eventos Scrum conforme exigidos ou necessários; e,
- ✓ Treinar o Time de Desenvolvimento em ambientes organizacionais nos quais o Scrum não é totalmente adotado e compreendido.

O Scrum Master

Na Organização

- ✓ Planejando implementações Scrum dentro da organização;
- ✓ Ajudando funcionários e partes interessadas a compreender e tornar aplicável o Scrum e o desenvolvimento de produto empírico;
- ✓ Trabalhando com outros Scrum Masters para aumentar a eficácia da aplicação do Scrum nas organizações.

O Time de Desenvolvimento

- ✓ Formado de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto "Pronto" ao final de cada Sprint.

- ✓ São estruturados e autorizados pela organização para organizar e gerenciar seu próprio trabalho.

- ✓ São auto-organizados. Ninguém (nem mesmo o Scrum Master) diz ao Time como transformar o Backlog do Produto em incrementos de funcionalidades potencialmente utilizáveis;

O Time de Desenvolvimento

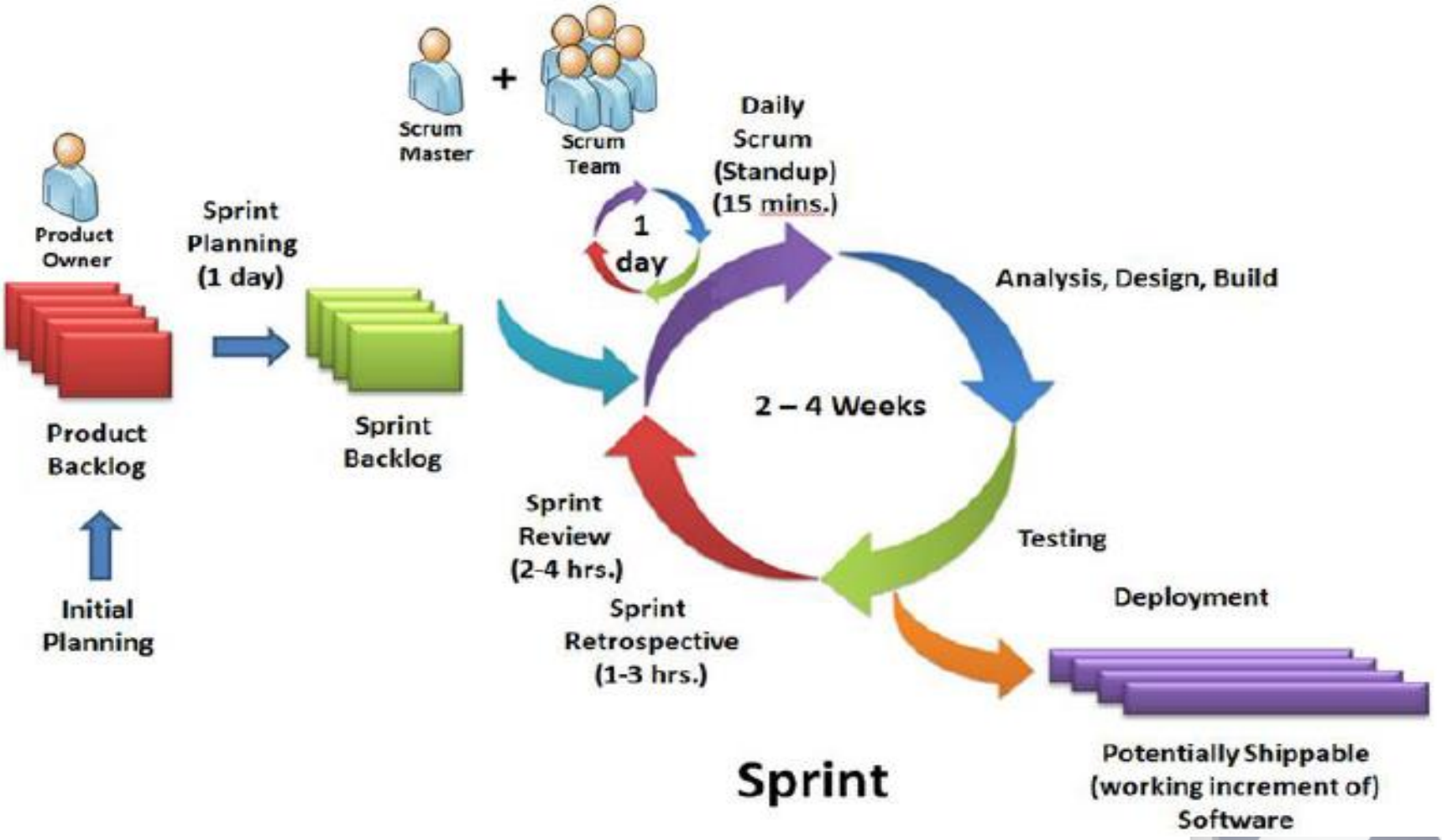
- ✓ São multifuncionais, possuindo todas as habilidades necessárias, enquanto equipe, para criar o incremento do Produto.
- ✓ O Scrum considera todos como Desenvolvedores, independentemente do trabalho que está sendo realizado pela pessoa;
- ✓ Não contém sub-times dedicados a domínios específicos de conhecimento, tais como teste ou análise de negócios.

O Time de Desenvolvimento

- ✓ Tamanho ideal - pequeno o suficiente para se manter ágil e grande o suficiente para completar uma parcela significativa do trabalho dentro dos limites da Sprint.
  Ou seja, entre 3 e 6 desenvolvedores.

- ✓ Os papéis de Product Owner e de Scrum Master não são incluídos nesta contagem, a menos que eles também executem o trabalho do Backlog da Sprint.

SCRUM


Scrum Master
+
Scrum Team
Daily Scrum (Standup) (15 mins.)
1 day
Product Owner
Sprint Planning (1 day)
Analysis, Design, Build
Product Backlog
Sprint Backlog
2 – 4 Weeks
Sprint Review (2-4 hrs.)
Testing
Initial Planning
Sprint Retrospective (1-3 hrs.)
Deployment
Potentially Shippable (working increment of) Software
Sprint

Questions

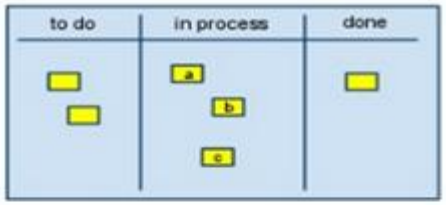
KANBAN – Quadro de Sinais
3 Princípios Básicos

Inicia-se com "O que fazer agora"
Não define papéis nem processos

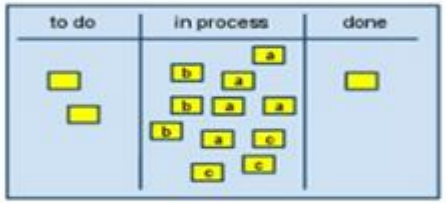
Adota mudanças incrementais e evolucionárias.

Respeita o processo corrente, papéis e responsabilidades.

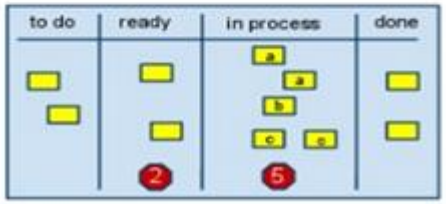
KANBAN – Quadro de Sinais



Começa com um quadro de tarefas, três colunas:
A fazer, em processo e terminado



Acumular cartões em andamento
Sem tempo definido



Limitar a qtde de cartões no quadro, para garantir o fluxo;
Limitar 2 cartões por pessoa;

KANBAN – 5 Propriedades Principais

Visualização do Fluxo

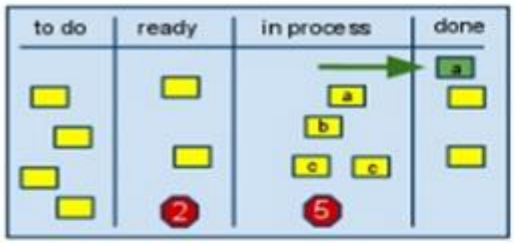
Limitar o trabalho “em andamento”

Gerenciar o fluxo, medir e reportar a cada estado

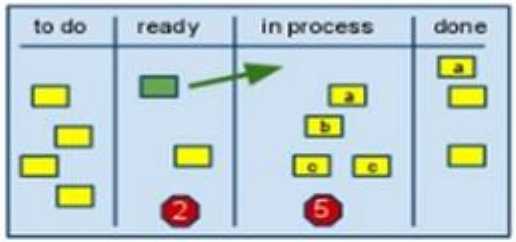
Tornar os processos e políticas explícitas

Aumentar e melhorar a colaboração

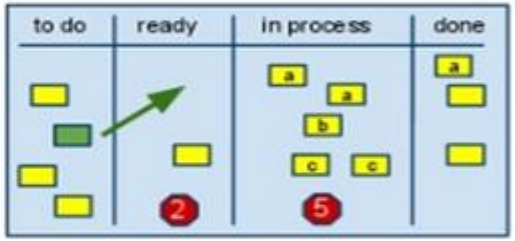
KANBAN – Mecanismo



1. Team member A completes a card and moves it to the "done" column.



2. Team member A pulls a new card from the "ready" column and starts working on it by placing it in the "in process" column.



3. The team responds to the pull event and selects the next priority card by moving it to the "ready" column.

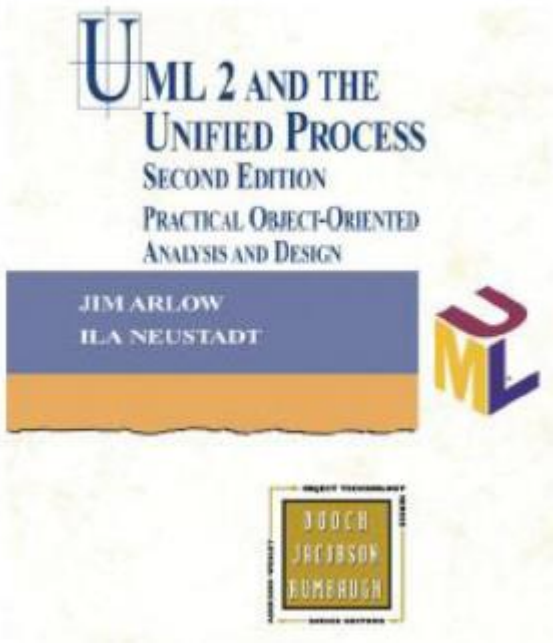
Learn more at Torak.com


break

Unified Process (UP)

Processo Unificado (PU)

Bibliografia



- http://www-01.ibm.com/software/rational/rup/
- http://epf.eclipse.org/wikis/openup

Processo Unificado (PU)

É usado para a construção de sistemas orientados a objetos, sendo um modelo de desenvolvimento de software iterativo e adaptativo

É o conjunto de atividades necessárias para transformar requisitos do usuário em um sistema de software.

Processo Unificado (PU)

Quando Surgiu?

O Processo Unificado é um Framework de Processo documentado no livro “The Unified Software Development Process“ de três amigos Grady Booch, James Rumbaugh e Ivar Jacobson - (Addison-Wesley, 1999), sendo baseado na experiência deles.

Processo Unificado (PU)

Características do PU

Dirigido por casos de uso
Ter os casos de uso como entrada (fonte) para a maioria das atividades do processo

Centrado na arquitetura
Motivado a desenvolver o produto com base em uma arquitetura de software

Processo Unificado (PU)

Características do PU

Iterativo e incremental
Dividir o projeto em partes gerenciáveis, de forma a incrementar as funcionalidades continuamente até o final da construção do produto

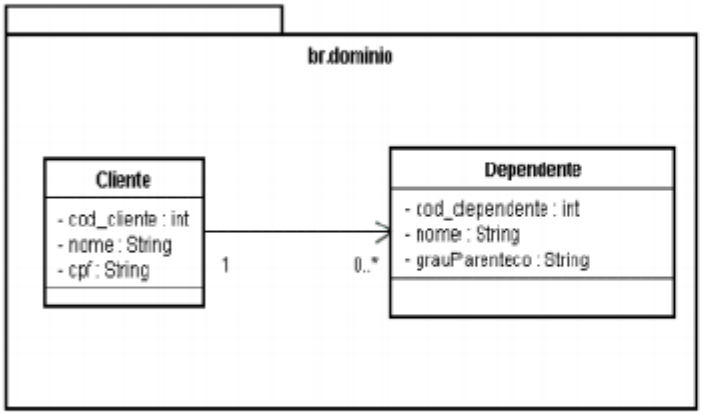
Focado no Risco
O Processo Unificado requer que a equipe do projeto concentre-se em enfrentar os Riscos mais críticos no início do ciclo de vida do projeto.

Processo Unificado (PU)

Onde entra a **UML**?

O PU usa UML como linguagem de modelagem

Qual a razão de utilizar linguagens gráficas?




```
package br.dominio;
public class Cliente {
    private int cod_cliente;
    private String nome;
    private String cpf;
    private Dependente[] dependente;
}
```

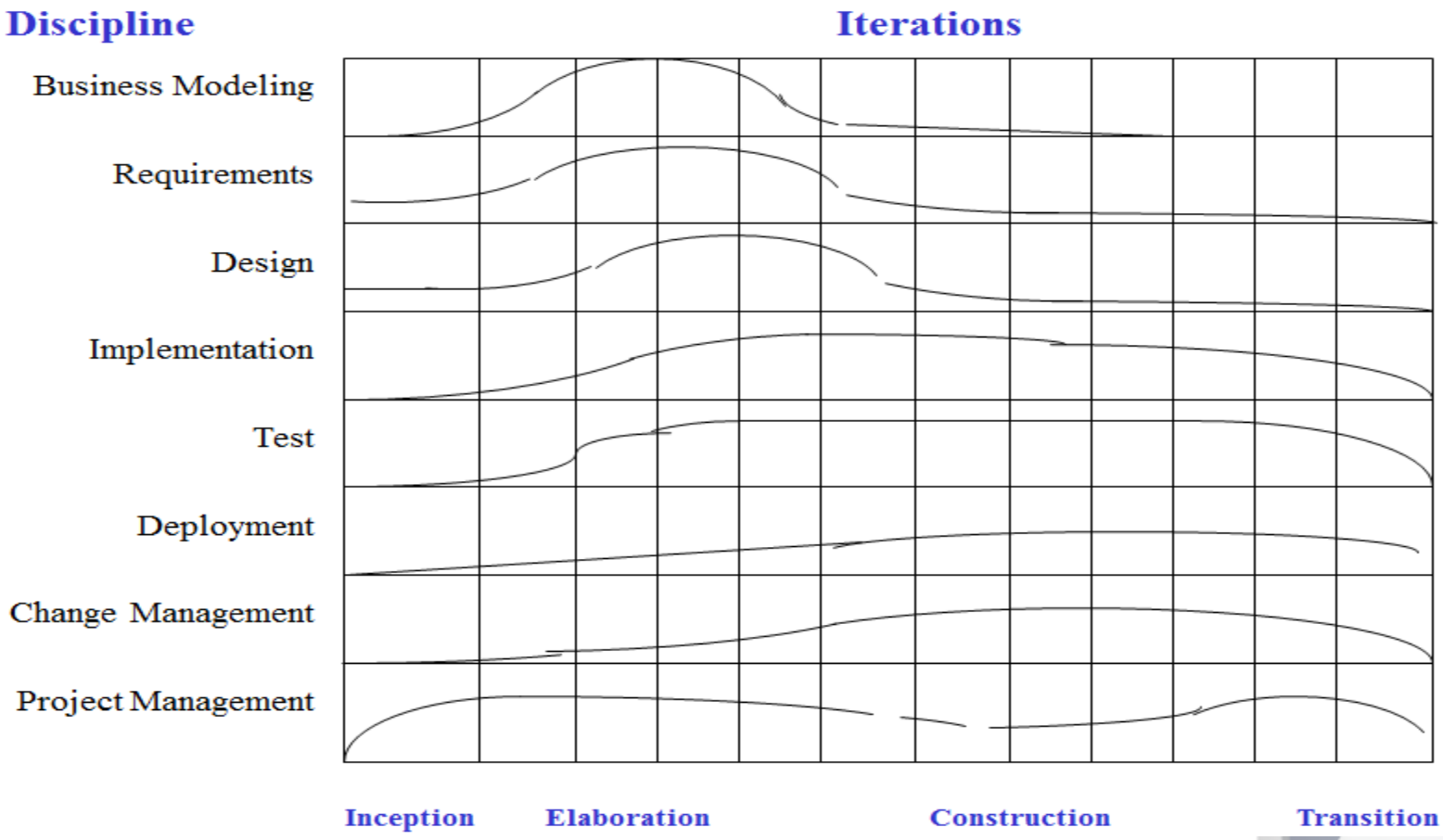
```
package br.dominio;
public class Dependente {
    private int cod_dependente;
    private String nome;
    private String grauParenteco;
}
```

Faculdade IMPACTA TECNOLOGIA

Processo Unificado (PU)

- ✓ A **Unified Modeling Language** (**UML**) é uma linguagem de modelagem de terceira geração.
- ✓ Não é uma metodologia de desenvolvimento
- ✓ Auxilia a visualizar desenho e a comunicação entre objetos, facilitando o entendimento de aspectos complexos inerentes ao sistemas
- ✓ É uma família de notações gráficas que ajuda na descrição e no projeto de sistemas de software

Processo Unificado (PU)



Processo Unificado (PU)

Características do PU - Casos de uso

- ✓ Sequência de ações que são executadas por um ou mais atores e pelo próprio sistema
- ✓ Produz um ou mais resultados de valor para um ou mais atores
- ✓ São expressos sob a perspectiva dos usuários do sistema

Processo Unificado (PU)

Características do PU - Casos de uso

- ✓ Expressos em língua natural, intuitivamente óbvios para o leitor
- ✓ Permitem maior compreensão dos requisitos do que outros documentos típicos
- ✓ Permitem um alto grau de rastreamento de requisitos com outros artefatos
- ✓ Meio simples de decompor os requisitos dos usuários em pedaços menores que permitam alocação de trabalho de sub-equipes

Processo Unificado (PU)

Características do PU - Arquitetura

É o sistema de programas (software) que realizam as transformações de dados em negócios, e pode incorporar também pessoas (transformadores, usuários) e os procedimentos de negócios automatizados.

Pressmann

Processo Unificado (PU)

Características do PU - Arquitetura

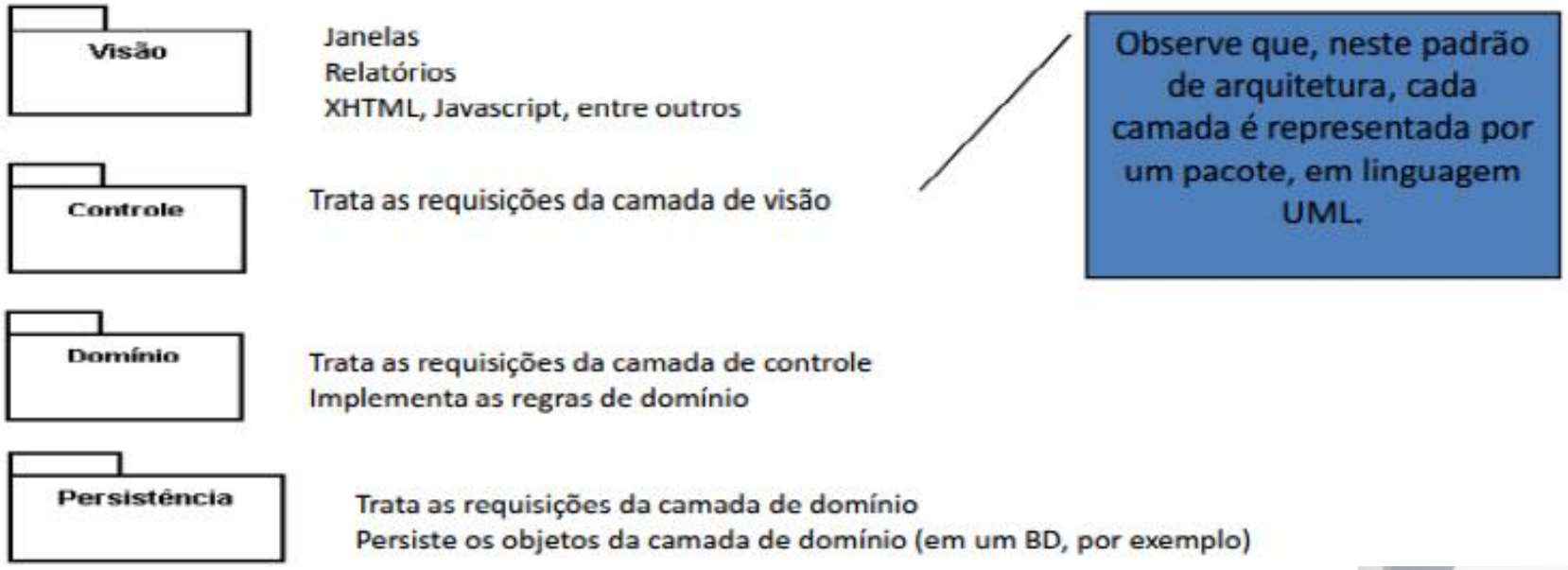
- ✓ Entender a visão global, simplificando o entendimento de sistemas complexos
- ✓ Organizar o esforço de desenvolvimento, dividindo o software em porções discretas
- ✓ Facilitar as possibilidades de reuso, facilitando o reuso de componentes dentro das porções discretas
- ✓ Facilitar a evolução do sistema
- ✓ Dirigir casos de uso, fornecendo condições de sempre adicionar mais casos de uso

Processo Unificado (PU)

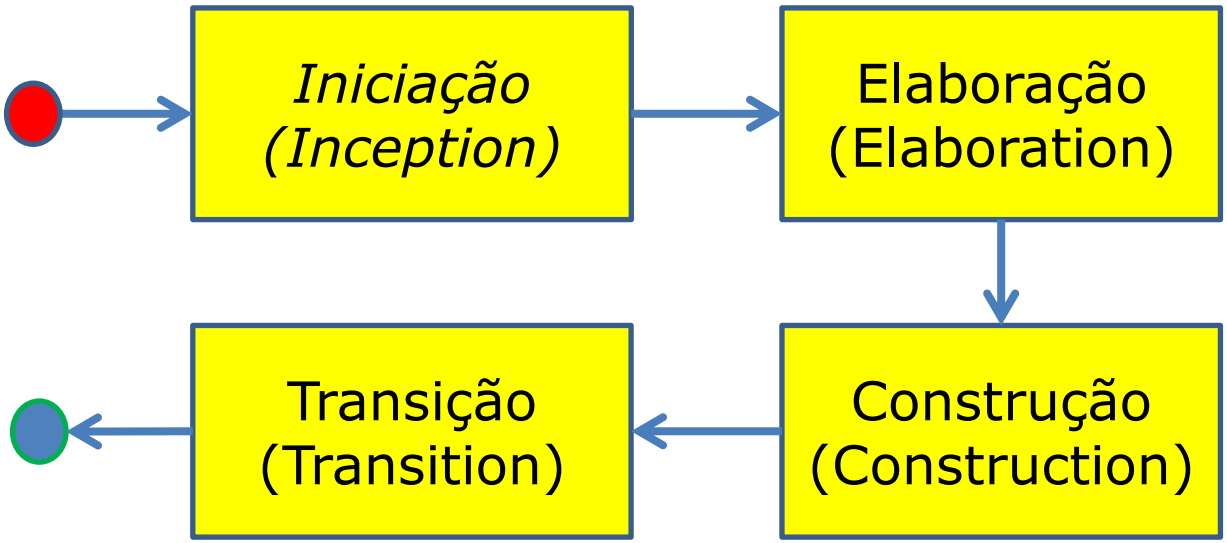
Características do PU

Padrão Arquitetural: Camadas

Organizar a estrutura lógica de um sistema em camadas distintas de responsabilidades precisas e relacionadas



PU – Ciclo de Vida



Ciclo de Vida - Fases do PU

- ✓ Concepção ou iniciação (Inception)
- ✓ Elaboração (Elaboration)
- ✓ Construção (Construction)
- ✓ Transição (Transition)

Ciclo de Vida - Fases do PU

Concepção ou iniciação (*Inception*)
Visão aproximada, casos de negócios, escopo e estimativas vagas

Elaboração (*Elaboration*)
Visão refinada, implementação iterativa da arquitetura central, resolução dos altos riscos, identificação da maioria dos requisitos e estimativas mais realistas

Ciclo de Vida - Fases do PU

Construção (*Construction*)
Implementação iterativa dos elementos restantes de menor risco e mais fáceis e preparação para a implementação

Transição (*Transition*)
Testes beta e implantação

Fluxo de Trabalho

- ✓ Cada fluxo indica um conjunto de atividades e vários tipos de membros que as executam
- ✓ No **processo unificado** existem 5 fluxos de trabalho: requisito, análise, projeto, implementação e teste
- ✓ Os fluxos permeiam as 4 fases do processo unificado
- ✓ **Fluxos de trabalho** é sinônimo de **disciplinas** (oriundo do RUP)

Fluxo de Trabalho (Disciplinas)

- ✓ Requisitos
- ✓ Análise
- ✓ Projeto
- ✓ Implementação
- ✓ Teste

Requisitos

- ✓ Visa definir e fechar o escopo (requisitos de alto nível) a ser desenvolvido

Análise

- ✓ Visa construir o modelo de análise, que ajuda os desenvolvedores a refinar e estruturar os requisitos funcionais

Projeto

- ✓ Visa a construir o modelo de projeto, o qual descreve as realizações físicas dos casos de uso.
- ✓ Visa também o modelo de instalação, que define a organização física dos sistema

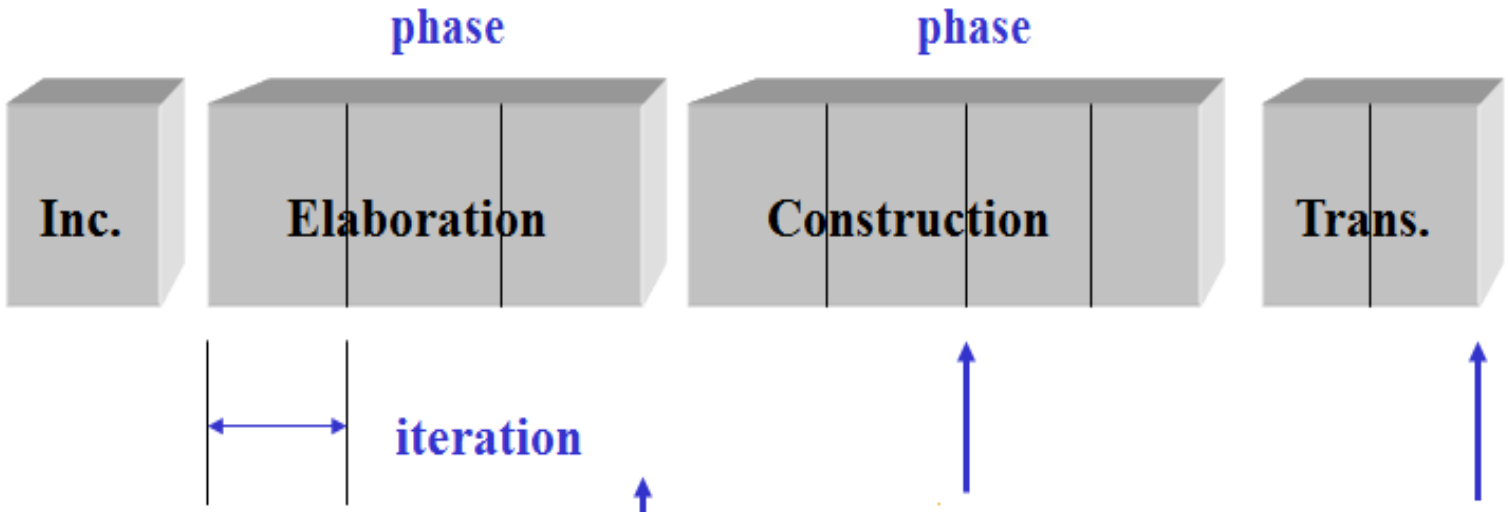
Implementação

- ✓ Visa a construir modelo de implementação, que descreve como os elementos do modelo de projeto são empacotados em componentes de software

Teste

Visa a construção do modelo de teste que descreve como os testes de integração e de sistema exercitarão componentes executáveis a partir do modelo de implementação

Fases x Iterações



Marco de Referência
Ponto de término de uma Iteração no qual ocorre alguma decisão significativa.

Versão
Um subconjunto estável executável do produto final.

Incremento
Diferença entre as entregas de 2 etapas subseqüentes.

Entrega Final
O sistema é entregue para uso.

Iterações x Fluxos de Trabalho

- ✓ As iterações podem conter os 5 fluxos de trabalho (R.A.P.I.T.)
- ✓ As iterações do início do processo normalmente ficam nos fluxos de requisitos e análise
- ✓ Estas iterações podem chegar ao fluxo de projeto, mas raramente ao fluxo de implementação e teste
- ✓ As iterações do final do processo de desenvolvimento normalmente executam os fluxos de implementação e teste

Iterações e Incrementos

Cada fase é divida em iterações

1 iteração, usa os 5 fluxos de trabalho (disciplinas)

O resultado da execução dos fluxos é um incremento (É uma versão do sistema que contém funcionalidades adicionada ou melhorada em comparação com a anterior)

Atividades de cada Iteração

- ✓Planejar iteração
- ✓Executar as disciplinas (fluxos de trabalhos)
- ✓Fazer análise ao término da iteração
- ✓Descartar os riscos que o incremento tratou
- ✓Revisar o plano do projeto
- ✓Ir para a próxima iteração, se existir

Artefatos

É qualquer porção significativa de informação interna ou externa que desempenhe um papel no desenvolvimento do sistema

Trabalhadores

É um papel que um indivíduo pode desempenhar no projeto em um dado momento

Diferença entre trabalhador e ator?

- ✓ Atores utilizam o sistema e podem participar do desenvolvimento, normalmente fornecendo informações sobre o cliente
- ✓ Trabalhadores somente participam do desenvolvimento do sistema

Disciplinas

- ✓ Determina um conjunto coerente de atividades a serem executadas para cumprir objetivos
- ✓ A disciplina é especificada por um fluxo de atividades, determinando os artefatos de entrada e saída de cada atividade deste fluxo bem como os respectivos níveis de detalhe de cada artefato

Pós-Graduação

Engenharia de Software

Faculdade IMPACTA TECNOLOGIA

Obrigada

Marta Fuzioka

mrtfuzioka@uol.com.br