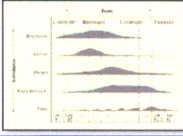
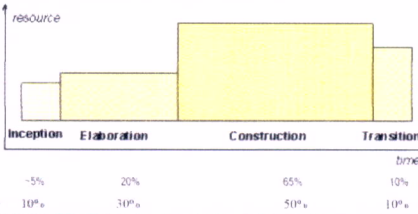


J964 : Engenharia de Software - 2019-1

RUP Fases



- **Concepção**
 - ♦ Estabelece os casos de negócio a serem atendidos pelo sistema.
 - ♦ No início, levanta, de forma genérica e pouco precisa, o escopo do projeto. O objetivo é ter uma visão inicial do problema, estimar esforço e prazos e determinar se o projeto é viável e merece uma análise mais profunda.
- **Elaboração**
 - ♦ Adquire uma compreensão do domínio do problema e da arquitetura do sistema.
 - ♦ Levanta a maior parte dos requisitos.
 - ♦ Em uma primeira iteração alguns requisitos, de maior risco e valor arquitetural, podem ser especificados em detalhes, implementados e servirem como base de avaliação junto ao usuário e desenvolvedores para o planejamento da próxima iteração. Ao fim da fase, 90% dos requisitos podem ter sido levantados em detalhes, o núcleo do sistema pode ter sido implementado com alta qualidade, os principais riscos podem ter sido tratados, podendo-se fazer estimativas mais realistas.
- **Construção**
 - ♦ Desenvolve o projeto, programação e teste do sistema.
 - ♦ Implementa, de forma iterativa, os elementos e prepara para a implantação.
- **Transição**
 - ♦ Realiza testes finais e implanta o sistema no ambiente de operação.

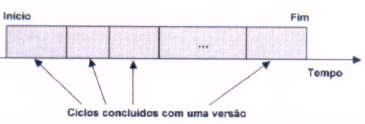


	Inception	Elaboration	Construction	Transition
Effort	~5%	20%	65%	10%
Schedule	10%	30%	50%	10%

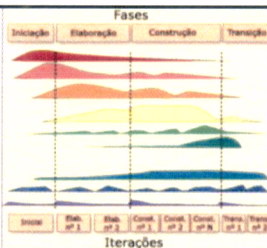
Engenharia de Software - Prof. Elvira Uchôa

5. Modelos de Ciclo de Vida - Slide 53

RUP Observações



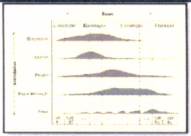
- O processo unificado consiste da repetição de uma série de **ciclos** durante o desenvolvimento de um sistema.
 - ♦ Cada **ciclo** é concluído com uma **versão do produto** pronta para entrega. Essa **versão** é um conjunto relativamente completo e consistente de artefatos, possivelmente incluindo manuais e um módulo executável do sistema, que pode ser usado pelos usuários.
 - ♦ Cada **entrega** do software em um ciclo agrega mais valor ao produto em relação ao ciclo anterior.
- Cada **ciclo** consiste de **quatro fases sequenciais**: concepção, elaboração, construção e transição.
 - ♦ Uma passagem pelas quatro fases produz uma **versão do software**.
 - ♦ A menos que o produto "desapareça", sua **próxima versão** será desenvolvida, repetindo a mesma sequência de fases de concepção, elaboração, construção e transição.
- Cada **fase** é subdividida em **iterações**.
 - ♦ Apesar de parecer um **modelo em cascata**, na verdade cada **fase** é composta de uma ou mais **iterações**, o que se assemelha a um **modelo em espiral**.
 - ♦ Estas **iterações** são em geral curtas (1-2 semanas) e abordam algumas **poucas funções** do sistema. Isto reduz o impacto de mudanças, pois quanto menor o tempo, menor a probabilidade de haver uma mudança neste período para as funções em questão.
 - ♦ Em cada **final de fase**, é executada uma avaliação para determinar se os objetivos da fase foram alcançados. Uma avaliação satisfatória permite que o projeto passe para a próxima fase.



Engenharia de Software - Prof. Elvira Uchôa

5. Modelos de Ciclo de Vida - Slide 54

J964 : Engenharia de Software - 2019-1

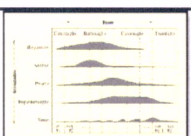


RUP

Atividades

- O Processo Unificado define **atividades** ou **disciplinas** para o desenvolvimento de software.
- **Disciplinas** organizam **papéis**, **ações** e **artefatos** em fluxos de trabalho. Elas são chamadas de workflows .
- Cada **fase** do processo é executada através de **atividades ou disciplinas**.
- **Atividades** podem ser executadas em qualquer **fase**, embora algumas sejam mais frequentes do que outras, de acordo com o objetivo da fase.
- Existem seis workflows principais e três workflows de apoio, identificados no processo.
- O Processo Unificado foi projetado com **base na UML** e, por isso, a descrição dos workflows faz referência aos modelos da UML.


Engenharia de Software - Prof. Elvira Uchôa 5. Modelos de Ciclo de Vida - Slide 55



RUP

Atividades

- **Workflows de engenharia (principais)**
 1. **Modelagem de negócio**: Os processos de negócios são representados por meio de casos de uso de negócios.
 2. **Requisitos**: Os atores que interagem com o sistema são identificados e os casos de uso são especificados em detalhes para representar os requisitos do sistema.
 3. **Análise e projeto (design)**: Um modelo de projeto é criado e documentado usando modelos de arquitetura, modelos de componentes, modelos de objeto e modelos de sequência.
 4. **Implementação**: Os componentes do sistema são implementados e estruturados em subsistemas de implementação. A geração automática de código a partir dos modelos de projeto ajuda a acelerar esse processo.
 5. **Teste**: O teste é um processo iterativo realizado em conjunto com a implementação. O teste de sistema é efetuado após o término da implementação.
 6. **Implantação**: Uma versão (*release*) do produto é criada, entregue aos usuários e instalada no local de trabalho.
- **Workflows de apoio**
 1. **Gerenciamento de configuração e mudanças**: Este workflow de apoio gerencia as mudanças do sistema.
 2. **Gerenciamento de projetos**: Este workflow gerencia o desenvolvimento do sistema.
 3. **Ambiente**: Este workflow está relacionado à disponibilização de ferramentas apropriadas para a equipe de desenvolvimento de software.



```

graph TD
    Modelagem[Modelagem de Negócio] --> Requisitos[Requisitos]
    Requisitos --> Analise[Análise e Design]
    Analise --> Implementacao[Implementação]
    Implementacao --> Teste[Teste]
    Teste --> Implantacao[Implantação]
    Implantacao --> Avaliacao[Avaliação]
    Avaliacao --> Planejamento[Planejamento]
    Planejamento --> Modelagem
    Planejamento --> Gerenciamento[Gerenciamento de Configuração e Mudança Ambiente]
    Gerenciamento --> Avaliacao
    
```

Engenharia de Software - Prof. Elvira Uchôa 5. Modelos de Ciclo de Vida - Slide 56

J964 : Engenharia de Software - 2019-1

Programação em par: enquanto um escreve o código, o outro monitora falhas, realiza testes, faz sugestões e planeja próximas ações.

Metodologia Ágil
EXtreme Programming (XP)

Sommerville, 2011

O processo ágil mais **amplamente utilizado**, originalmente proposto por Kent Beck

Processo XP

- Começa com a criação de "histórias de usuários" (cenários que são usadas como base para decidir a funcionalidade que deve ser incluída em um incremento do sistema)
- Equipe ágil avalia cada história e atribui um **custo**
- As histórias são agrupadas em um **incremento de entrega**
- Um **compromisso** é assumido para uma data de entrega
 - Depois, a "velocidade do projeto" relativa ao primeiro incremento é usado para ajudar a definir as datas de entrega subsequentes para outros incrementos
- Os programadores trabalham em pares e desenvolvem testes **ANTES** de escreverem o código.
- A prática de integração contínua envolve a geração frequente de versões (builds ou executáveis) do sistema, assim como execução dos testes automatizados sobre as versões geradas.
- A prática de refatoração envolve modificação interna do código do sistema, mas sem modificar seu comportamento externo.
 - Todos os desenvolvedores devem refatorar o código continuamente, assim que encontrarem oportunidades de melhorias de código.
 - Os desenvolvedores trabalham em todas as áreas do sistema, de modo que não se desenvolvam ilhas de expertise. Todos os desenvolvedores têm responsabilidade em relação ao código; qualquer um pode mudar qualquer coisa.

Engenharia de Software - Prof. Elvira Uchôa

5. Modelos de Ciclo de Vida - Slide 91

Metodologia Ágil

Agile Method

Abordagem ágil

equipes pequenas

mesmo espaço físico

pequenos incrementos

PO/Cliente presente

colaboração

negociação

Engenharia de Software - Prof. Elvira Uchôa

5. Modelos de Ciclo de Vida - Slide 92