Engenharia de Software I

Profª. Me. Cynara Leão Garcia

cynara.garcia@unicesumar.edu.br

Verificação e Validação

Verificação e Validação

- Objetivos da verificação e validação
  - Mostrar que o software atende a sua especificação
  - Mostrar que o software atende as necessidades do cliente

- Teste é a principal técnica de V&V
  - Técnicas de inspeção e revisão também são usadas

UniCesumar

Verificação

- O objetivo é verificar se o software atende aos requisitos funcionais e não funcionais especificados

- Verificação inclui da realização de testes para encontrar erros

- Pergunta principal

  - Estamos construindo o produto da maneira correta?

UniCesumar

Validação

- A inexistência de erros não mostra a adequação operacional do sistema
  - Deve ser feita a **validação** com o cliente

- A validação procura assegurar que o sistema atenda as expectativas e necessidades do cliente

- Pergunta principal
  - Estamos construindo o produto correto?

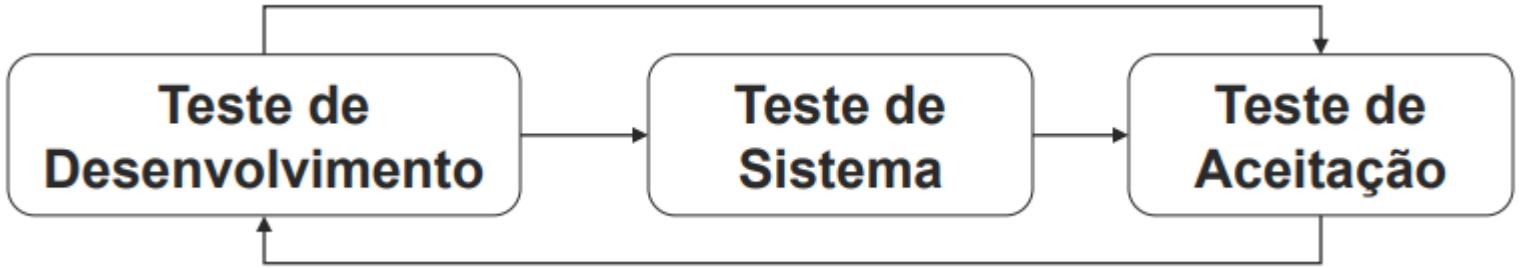
UniCesumar

Estágios de Teste

- Teste de Desenvolvimento
  - Os componentes são testados pelas pessoas que os desenvolvem
  - Ferramentas, como o JUnit, são usadas para re-executar os testes
- Teste de Release (Sistema)
  - O sistema é testado por uma equipe independente antes da entrega ao cliente
- Teste de Usuário (Aceitação)
  - O sistema é testado com dados do cliente

UniCesumar

Processo de Teste



- Objetivos das atividades

  - Teste de Desenvolvimento: descobrir erros no início do processo

  - Teste de Sistema: encontrar erros pela interações entre componentes

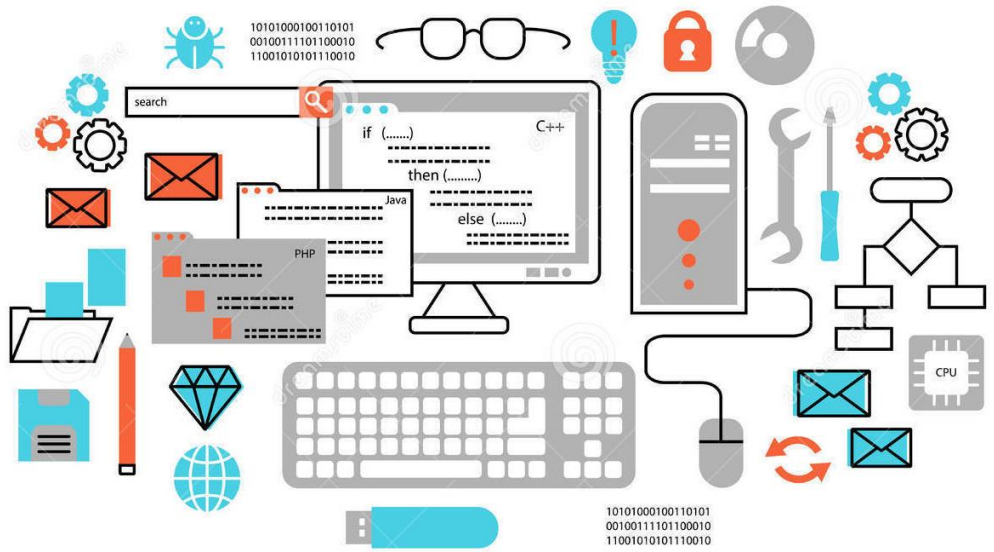
  - Teste de Aceitação: garantir que o sistema satisfaz as necessidades dos usuários

UniCesumar

Desenvolvimento Incremental

- Cada incremento é testado enquanto é desenvolvido
  - Testes são baseados nos requisitos do incremento

- Em XP, os testes são escritos antes de se iniciar a implementação
  - Desenvolvimento Dirigido por Testes (TDD)



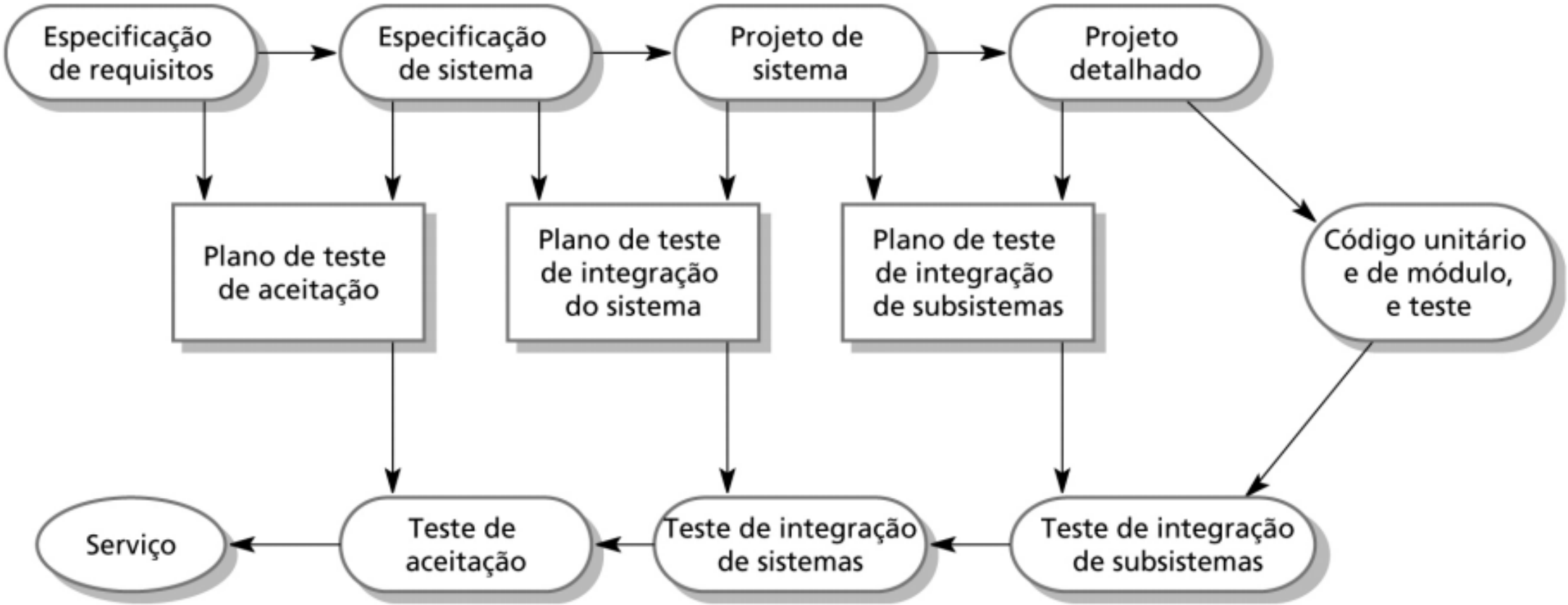
Desenvolvimento Tradicional

- Verificação e validação deve ocorrer durante e depois do desenvolvimento

- O Modelo V ilustra as atividades de testes durante o desenvolvimento
  - Começa na especificação de requisitos
  - Revisões de arquitetura e projeto
  - Inspeções e testes de código

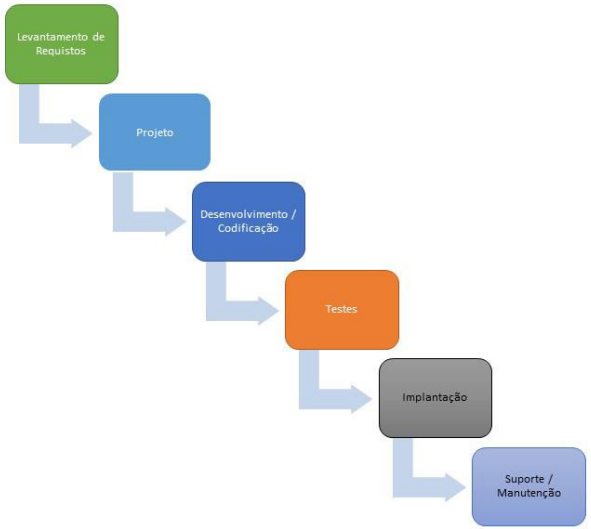
UniCesumar

Modelo V


Especificação de requisitos
Especificação de sistema
Projeto de sistema
Projeto detalhado
Plano de teste de aceitação
Plano de teste de integração do sistema
Plano de teste de integração de subsistemas
Código unitário e de módulo, e teste
Serviço
Teste de aceitação
Teste de integração de sistemas
Teste de integração de subsistemas

Características do Modelo V

- Modelo que integra o desenvolvimento aos testes
- É fortemente baseado no Modelo Cascata
- Os planos de testes são derivados das atividades de desenvolvimento



UniCesumar

Teste Alfa e Teste Beta

- Teste Alfa α

  - Testes de aceitação que ocorrem antes do sistema ser entregue ao cliente

  - É feito até que o cliente aceite que o sistema seja entregue

- Teste Beta β

  - Entrega o sistema a um conjunto de usuários (potenciais clientes)

  - Os usuários reportam os erros encontrados aos desenvolvedores

UniCesumar

Inspeção de Software

Inspeção de Software

"Testes podem somente revelar a presença de defeitos, não a ausência"

Dijkstra

- Testes fazem parte do processo de verificação e validação (V&V)
  - Devem ser usados em conjunto com a verificação estática (inspeção)

UniCesumar

Níveis de Confiabilidade

- Verificação e Validação buscam estabelecer a confiança de que o software está pronto para ser usado

- O nível de confiabilidade depende
  - Da finalidade do software
  - Das expectativas dos usuários
  - Do ambiente de mercado

Finalidade do Sistema

- Sistema Crítico
  - O nível de confiança é muito maior
  - O software deve ser confiável

- Protótipo
  - O nível de confiança é menor
  - É aceitável que o software possa falhar

Expectativa de Usuários

- Usuários podem já estar acostumados com software de baixa qualidade

- Usuários tendem a aceitar falhas quando os benefícios superam as desvantagens

  - A tolerância a falhas dos usuários diminuí a medida que o software amadurece

UniCesumar

Ambiente de Mercado

- Sistemas comerciais devem levar em conta os programas concorrentes

- Em um ambiente competitivo
  - O sistema pode ser liberado mais cedo pelo pioneirismo
  - V&V pode não ter sido bem feita

- Quando clientes não querem pagar caro pelo produto
  - Eles geralmente aceitam alguns defeitos

UniCesumar

Teste de Software vc. Inspeção de Software

- Teste é uma técnica dinâmica de V&V
  - Consiste em executar uma implementação com dados de teste

- Pode ser usada para avaliar os requisitos não funcionais
  - Desempenho, confiabilidade, segurança, etc.

UniCesumar

Teste de Software

- Teste de software buscam por erros ou anomalias em requisitos funcionais e não funcionais

- Classificação de testes pelo objetivo

  - **Teste de Validação:** mostrar que um programa faz o que é proposto a fazer

  - **Teste de Defeito:** descobrir os defeitos do programa antes do uso

UniCesumar

Teste de Validação

- Pretende mostrar que o software atende aos seus requisitos
  - Faz o que o cliente deseja
- Um teste bem sucedido mostra que o requisito foi implementado
- Refletem o uso esperado do software

Teste de Defeito

- Destinado a revelar defeitos no sistema

- Um teste de defeitos bem sucedido é aquele que revela defeitos no sistema

- Os casos de teste podem ser obscuros

  - Não precisam refletir exatamente como o sistema é normalmente usado

UniCesumar

Resultados e Relatórios

- Resultados de teste

  - Saídas que somente podem ser previstas por pessoas que conhecem o domínio de negócio do sistema

- Relatório de teste

  - Pode ser feito de forma manual, seguindo um formulário específico

  - Pode ser automatizado comparando os resultados esperados às saídas dos testes

UniCesumar

Outros tipos de teste

- Teste de Integração;
- Teste de Sistema;
- Teste de Aceitação;
- Teste de Desempenho;
- Teste de Estresse.

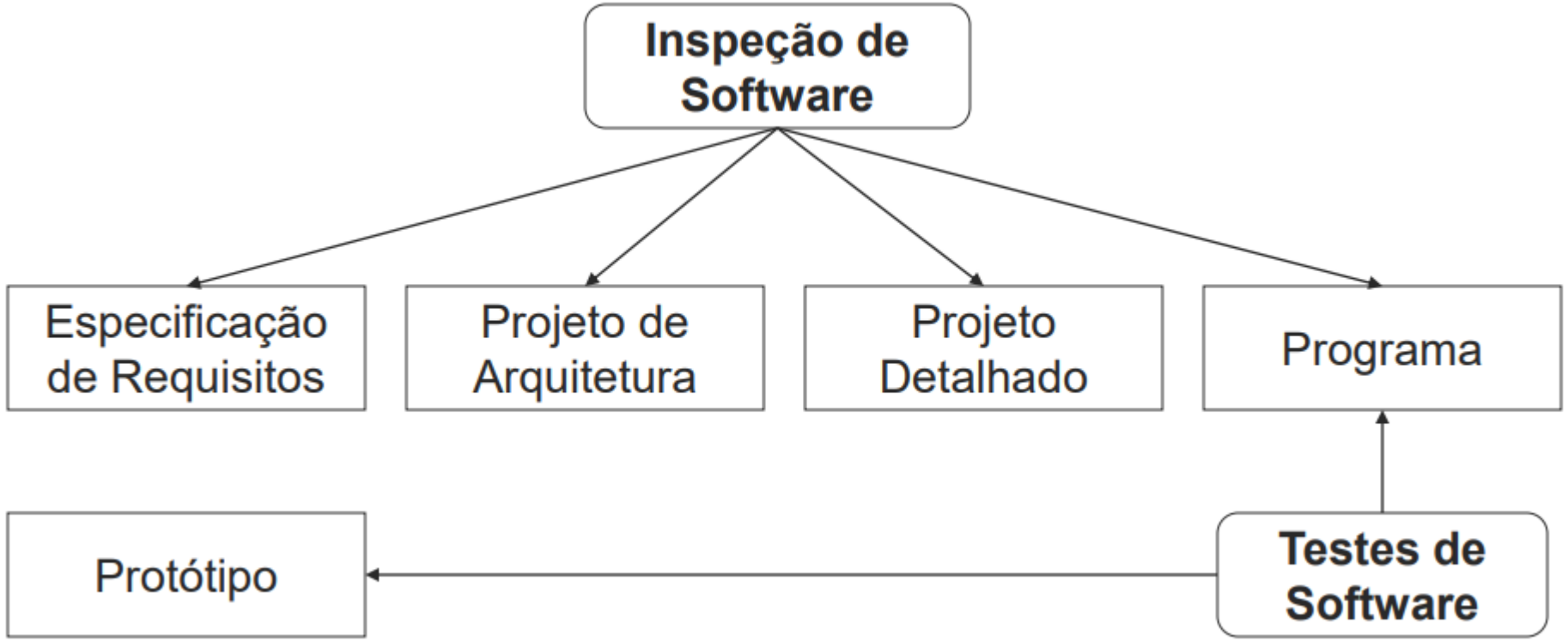
UniCesumar

Teste de Software vc. Inspeção de Software

- Técnica estática de V&V
  - Não precisa executar o programa
- Pode ser usada em qualquer atividade de desenvolvimento
  - Requisitos, projeto, código fonte, etc.
- Pode ser semi-automatizada por análise estática
  - Análise não automatizada é cara

UniCesumar

Verificação Estática e Dinâmica



Vantagens da Inspeção

- Muitos defeitos diferentes podem ser descobertos em uma única inspeção
  - Um teste revela um defeito e oculta vários
- Versões incompletas do sistema podem ser inspecionadas
- Permite encontrar problemas em outros atributos de qualidade do software
  - Uso de um algoritmo mais eficiente, padrão de projeto, etc.

UniCesumar

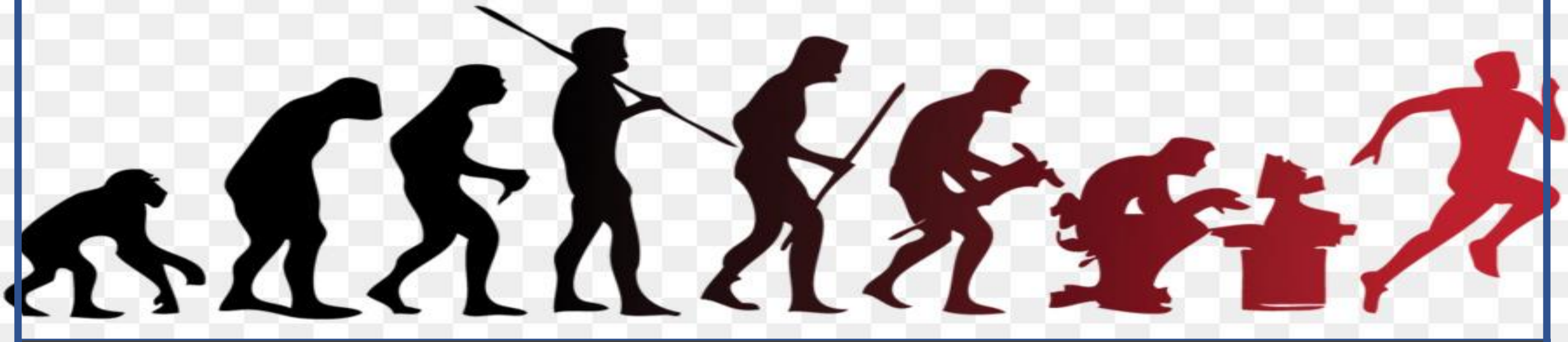
Limitações de Inspeção

- Inspeção não é adequada
  - Para descobrir defeitos nas interações
  - Para demonstrar se o software é útil
  - Para verificar requisitos não funcionais, como desempenho, segurança, etc.

- Inspeção é uma técnica cara

- Ela não permite validar com o cliente
  - Somente verifica a correspondência entre a especificação e o software

UniCesumar



Evolução de Software

Evolução de Software

- Depois de implantados, sistemas devem inevitavelmente mudar para permanecerem úteis

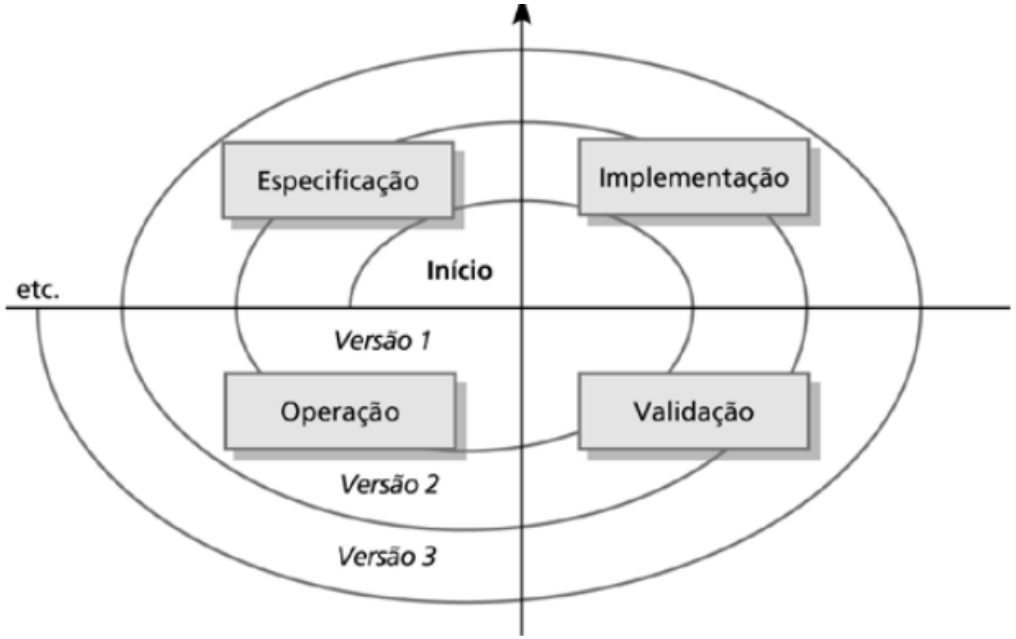
- Mudanças no ambiente requerem atualizações do sistema

  - Ao implantar um sistema modificado, este sistema causa uma mudança no ambiente

Evolução de Software

- Sistemas de software geralmente têm vida útil longa
  - Organizações são dependentes dos sistemas que custaram milhões

- O Modelo Espiral reflete os ciclos contínuos e ininterruptos de desenvolvimento e evolução

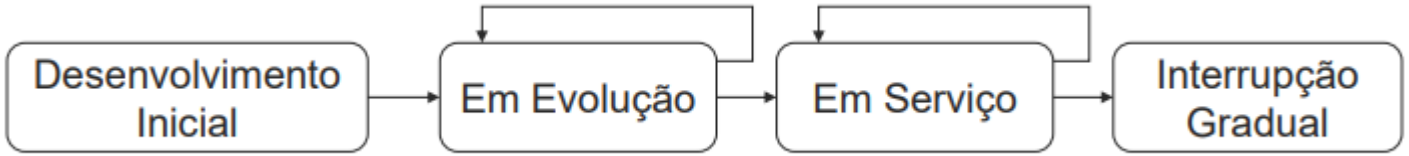


Fases de um sistema

- Desenvolvimento Inicial
  - Primeira solicitação do cliente
- Em Evolução
  - Sistema é intensamente usado
- Em Serviço
  - Sistema é pouco usado
- Interrupção gradual
  - Empresa considera a substituição do sistema

UniCesumar

Em Evolução vs. Em Serviço



- **Em Evolução**

  - Mudanças significativas são feitas tanto na arquitetura quando nas funcionalidades

  - A estrutura tende a gradativamente se degradar

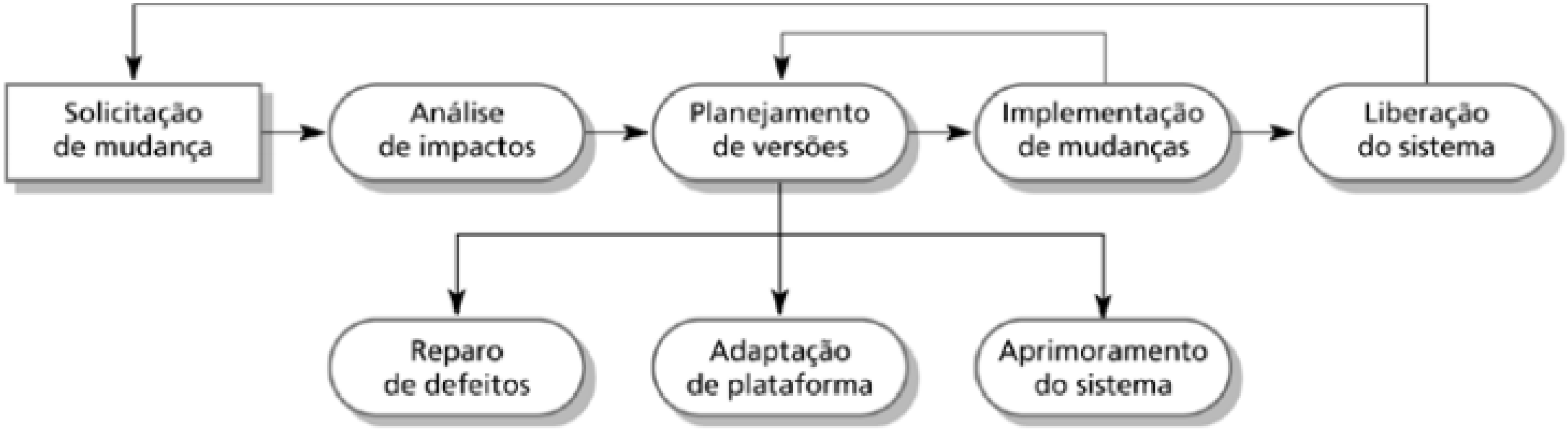
- **Em Serviço**

  - Apenas mudanças pequenas e essenciais ocorrem (software é pouco usado)

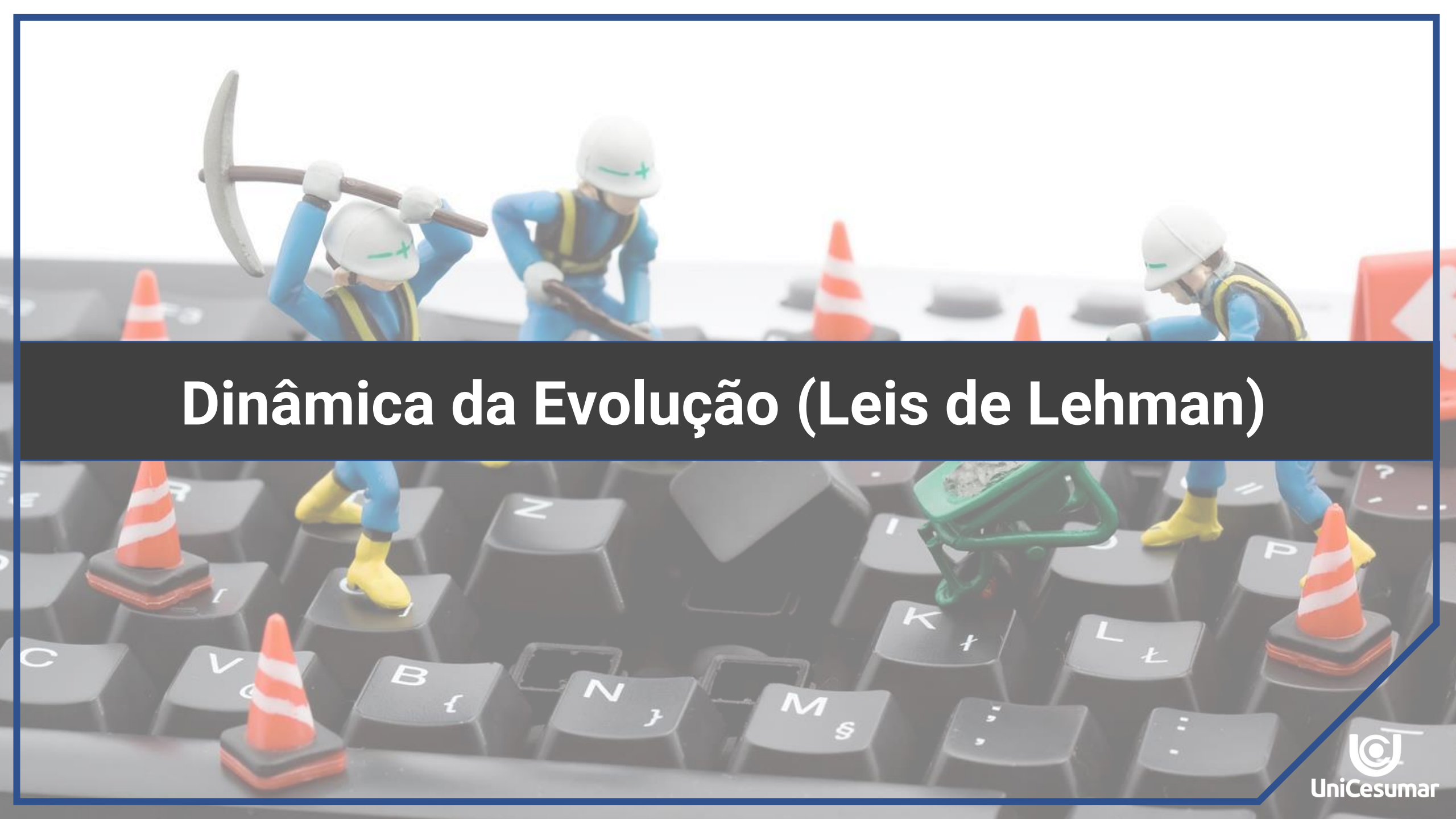
UniCesumar

Processo de Evolução

- Processo Informal;
- Processo Formal.



UniCesumar



Dinâmica da Evolução (Leis de Lehman)

UniCesumar

Dinâmica da Evolução (Leis de Lehman)

- O crescimento e a evolução de vários sistemas de grande porte foram examinados

- Objetivo
  - Definir uma teoria unificada para evolução de software

- Resultados
  - Um conjunto de oito leis que “governam” a evolução de sistemas

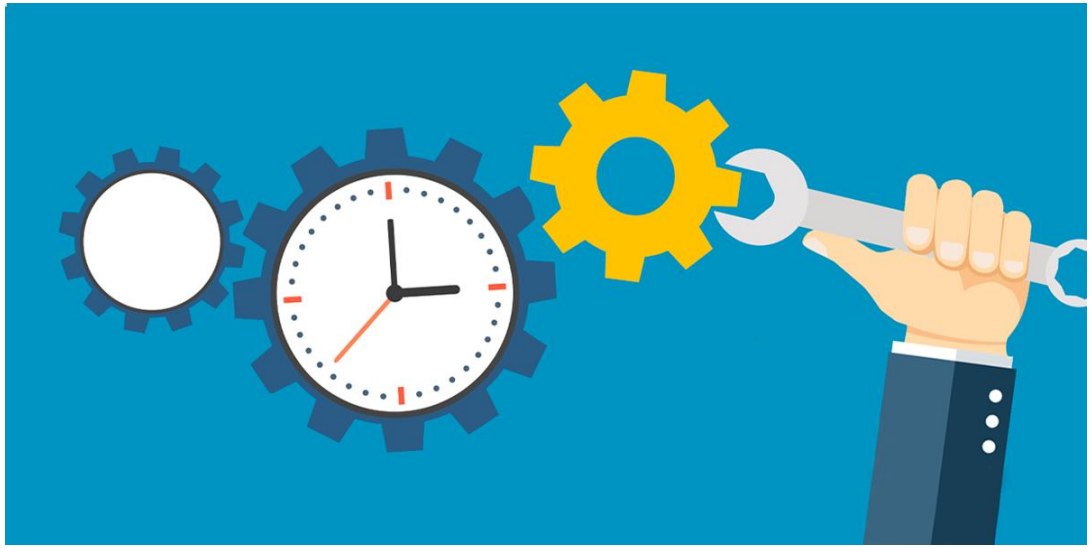
UniCesumar

1 – Mudança Contínua

- Sistemas devem ser continuamente adaptados ou eles se tornam progressivamente menos satisfatórios

  - O ambiente muda, novos requisitos surgem e o sistema deve ser modificado

  - Após o sistema modificado ser reimplantado, ele muda o ambiente



2 – Complexidade Crescente

- A medida que um sistema evolui, sua complexidade aumenta, a menos que seja realizado esforço para mantê-la ou diminuí-la
  - Manutenções preventivas são necessárias
  - Precisa-se de recursos extras para implementar as mudanças

UniCesumar

3 – Auto-Regulação

- A evolução de software é um processo auto-regulável

  - Atributos do sistema como tamanho, tempo entre versões e número de erros reportados é quase invariável em cada versão do sistema

- Uma grande alteração inibe futuras grandes alterações

  - Consequência de fatores estruturais e organizacionais

UniCesumar

4 – Estabilidade Organizacional

- Durante o ciclo de vida de um programa, sua taxa de desenvolvimento é quase constante

  - Independe de recursos dedicados ao desenvolvimento do sistema

  - Alocação de grandes equipes é improdutivo, pois o *overhead* de comunicação predomina

5 – Conservação de Familiaridade

- Durante o ciclo de vida de um sistema, mudanças incrementais em cada versão são quase constantes

  - Um grande incremento em uma release leva a muitos defeitos novos

  - A release posterior será dedicada quase exclusivamente para corrigir os defeitos

- Ao orçar grandes incrementos, deve-se considerar as correções de defeitos

UniCesumar

6 – Crescimento Contínuo

- O conteúdo funcional de sistemas devem ser continuamente aumentado para manter a satisfação do usuário

  - A cada dia, o usuário fica mais descontente com o sistema

  - Novas funcionalidades são necessárias para manter a satisfação do usuário

7 – Qualidade Declinante

- A qualidade de sistemas parecerá estar declinando a menos que eles sejam mantidos e adaptados às modificações do ambiente

8 – Sistema de Feedback

- Os processos de evolução incorporam sistemas de feedback com vários agentes e loops

  - Estes agentes, loops e feedback devem ser considerados para conseguir melhorias significativas do produto

Engenharia de Software I

UniCesumar