

Engenharia de Software

09 - Evolução de Software

Prof. Msc. Francisco de Assis de Souza Rodrigues

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

"Os sábios acumulam conhecimento, mas a boca do tolo é uma ruína iminente." **Provérbios 10:14**

Estudar ajuda a tomar decisões mais sábias e evita erros desnecessários na vida.

🎯 Objetivos

- **Compreender a Importância da Evolução:** Entender que os sistemas de software devem se adaptar e evoluir para se manterem úteis, e que a modificação é parte integrante da Engenharia de Software.

- **Identificar e Contextualizar Custos:** Reconhecer que a maior parte dos custos de software está relacionada à evolução e manutenção (entre 60% e 90% dos custos totais).

- **Definir Sistemas Legados:** Compreender o conceito de sistema legado, incluindo seus componentes sociotécnicos e a importância para o negócio.

UniFAP

🎯 Objetivos

- **Avaliar Estratégias de Gerenciamento:** Aplicar a matriz de valor de negócio vs. qualidade para decidir se um sistema legado deve ser descartado, mantido, reprojetado ou substituído.

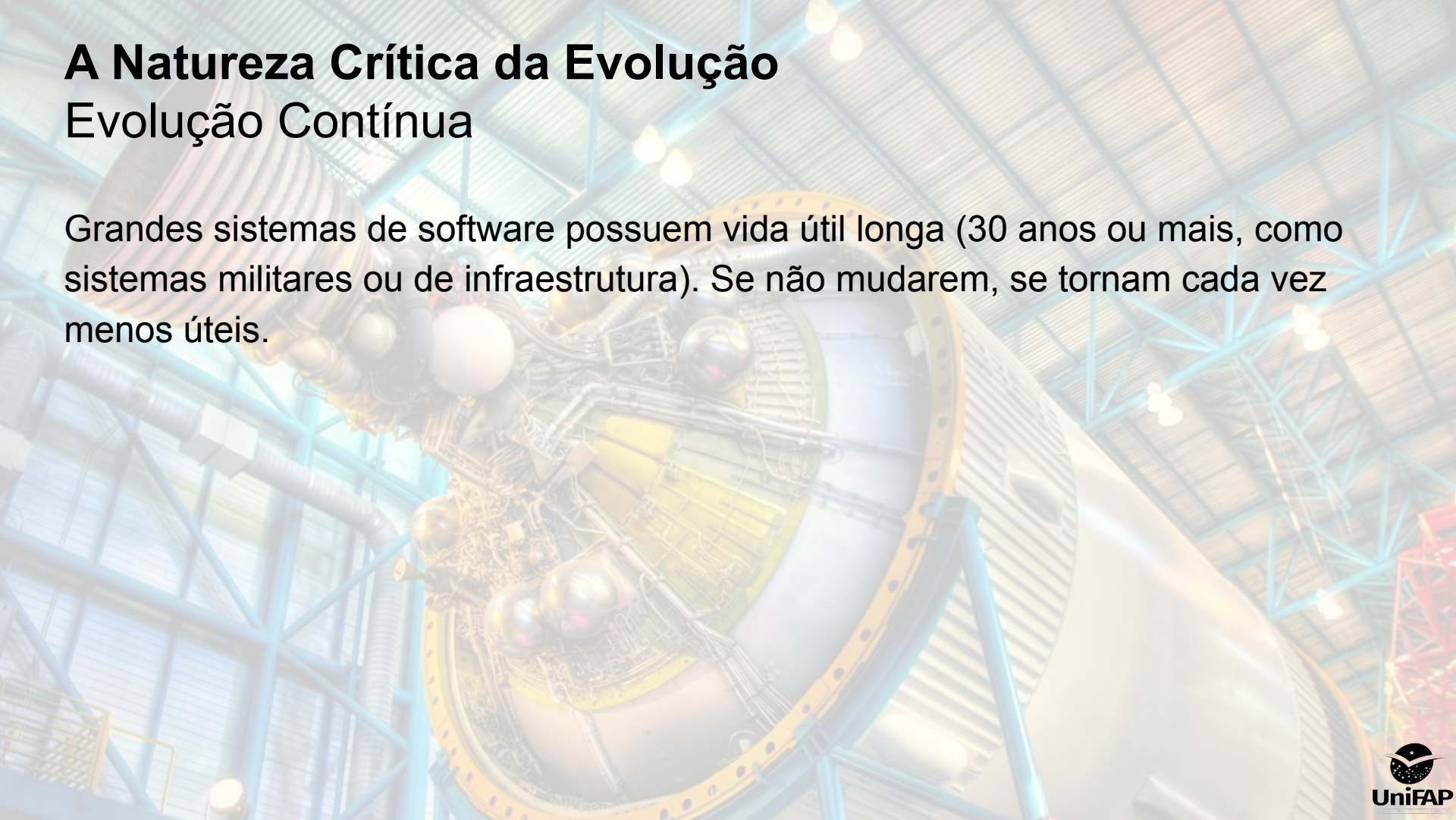
- **Diferenciar Manutenção e Reengenharia:** Distinguir os tipos de manutenção e descrever as técnicas de reengenharia e refatoração para melhorar a manutenibilidade

UniFAP

A Natureza Crítica da Evolução

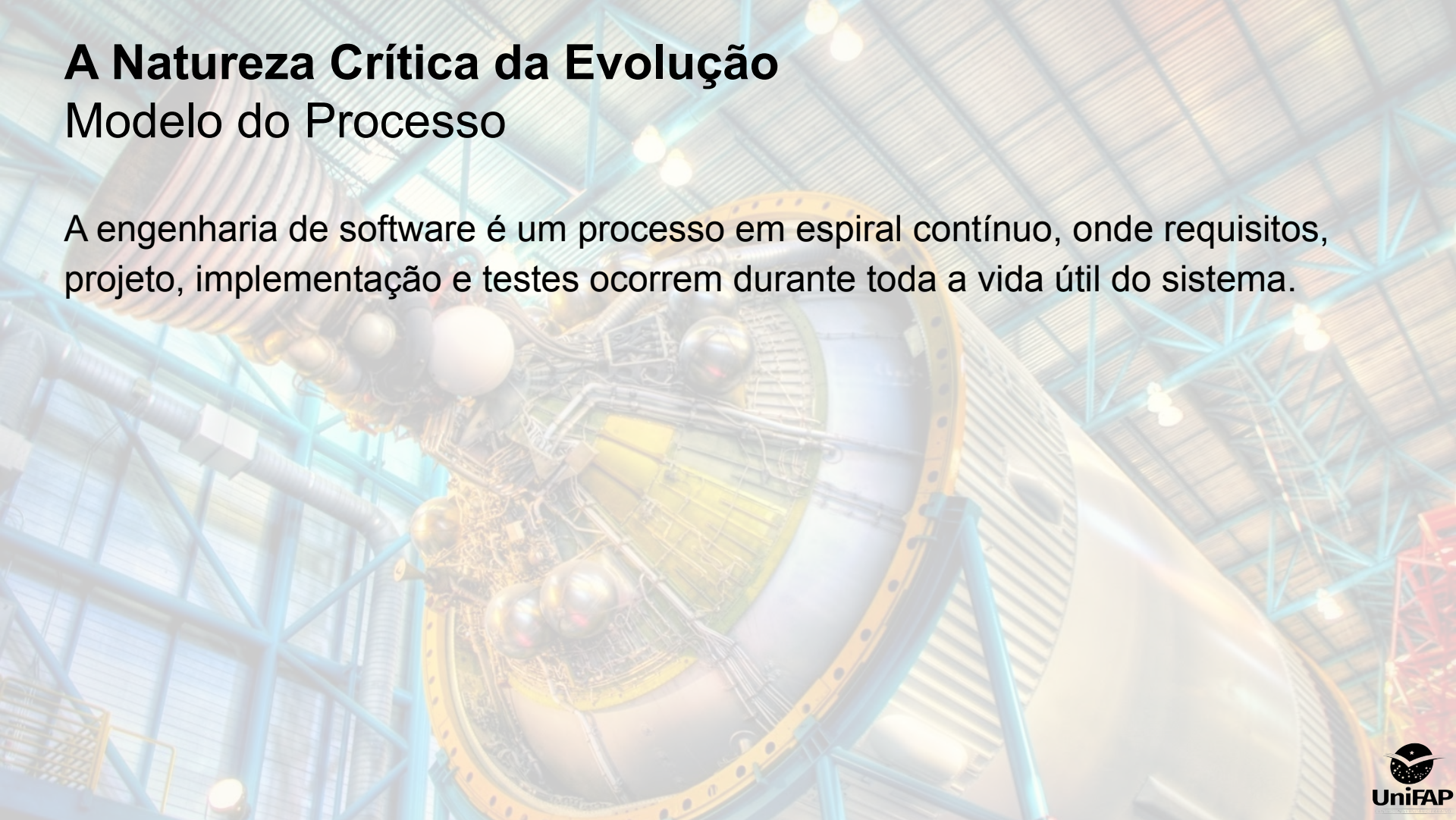
Evolução Contínua

Grandes sistemas de software possuem vida útil longa (30 anos ou mais, como sistemas militares ou de infraestrutura). Se não mudarem, se tornam cada vez menos úteis.



A Natureza Crítica da Evolução
Modelo do Processo

A engenharia de software é um processo em espiral contínuo, onde requisitos, projeto, implementação e testes ocorrem durante toda a vida útil do sistema.



A Natureza Crítica da Evolução
Custos (Perspectiva SI/ADS)

A maior parte dos custos de grandes empresas é gasta na manutenção de sistemas existentes, não no desenvolvimento de novos. O custo da evolução é significativo, especialmente em sistemas corporativos que são parte de um 'sistema de sistemas'.

UniFAP

A Natureza Crítica da Evolução

Processo de Evolução

É impulsionado por propostas de mudança (formais ou informais), que disparam a análise de mudança, o planejamento de lançamento, a implementação e o lançamento do sistema.

UniFAP

Manutenção de Software e Custos

Manutenção vs. Desenvolvimento

A manutenção de software é geralmente aplicada a software personalizado e ocorre quando há uma descontinuidade entre a equipe de desenvolvimento e a equipe responsável pela evolução do sistema. A manutenção requer atividades extras, como a compreensão do programa.

UniFAP

Manutenção de Software e Custos

Tipos de Manutenção

1. Reparo de Defeitos: Correção de erros.

2. Adaptação ao Ambiente: Mudança para operar em um novo ambiente (hardware, sistema operacional).

3. Acréscimo de Funcionalidade: Adição de novas características ou requisitos.
Destaque: Este é o tipo mais caro e consome a maior parte do esforço de manutenção.

UniFAP

Manutenção de Software e Custos

Fatores de Custo

Compreensão do Sistema: O novo time de manutenção gasta tempo entendendo o código e as decisões de projeto originais.

Degradação da Estrutura: As alterações ao longo do tempo corrompem a estrutura original do programa, tornando futuras mudanças mais difíceis e caras (Leis de Lehman).

Documentação Inconsistente: Em reparos emergenciais (como correções de segurança urgentes), a documentação pode não ser atualizada, levando à inconsistência entre código e documentação

UniFAP

Gerenciamento de Sistemas Legados

Sistemas Legados

- **Definição:** Sistemas mais antigos, baseados em tecnologias obsoletas, mas que continuam cruciais para a operação do negócio.
- **Natureza Sociotécnica:** Um sistema legado não é apenas o software. Ele inclui hardware antigo, software de apoio (ex: SO, BD), dados de aplicação, e processos de negócio e regras organizacionais que estão incorporadas nele.

Estratégias de Gerenciamento:

- A decisão de gestão (ADS/SI) é baseada na avaliação do Valor de Negócio e da Qualidade do Sistema

UniFAP

Gerenciamento de Sistemas Legados

Sistemas Legados

Qualidade do Sistema	Baixa Qualidade	Alta Qualidade
Alto Valor de Negócio	**Reengenharia ou Substituição Evolucionária**	**Manutenção Normal**
Baixo Valor de Negócio	**Descarte** (Se não for necessário)	**Descarte** (Se os custos de suporte forem altos)

UniFAP

Gerenciamento de Sistemas Legados

Sistemas Legados

- **Avaliação da Aplicação:** Fatores como a compreensibilidade do código, a qualidade da documentação e a duplicação de dados afetam a qualidade.

- **Avaliação do Ambiente:** Fatores como a estabilidade do fornecedor (se faliu, o sistema pode não ser mais suportado) e a taxa de falha do hardware influenciam a decisão.

Melhores Práticas para Manutenibilidade

Soluções para a Degradação do Código

Reengenharia de Software: Processo de reestruturar e redocumentar o software para torná-lo mais fácil de entender e mudar, sem alterar a funcionalidade.

- *Vantagens (para o negócio):* Menos arriscada e mais barata do que a substituição.
- *Processo:* Envolve análise, engenharia reversa para recuperar a documentação, reengenharia do código-fonte e, possivelmente, reengenharia de dados.
- *Observação:* A reengenharia pode envolver o desenvolvimento de serviços adaptadores para que o sistema legado interopere com novos sistemas.

UniFAP

Melhores Práticas para Manutenibilidade

Soluções para a Degradação do Código

Refatoração: Processo de fazer pequenas melhorias em um programa para refinar sua estrutura e reduzir a complexidade, agindo como uma manutenção preventiva.

- **Princípio:** Modificar o programa para melhorar sua inteligibilidade, sem adicionar funcionalidade.
- **Prática:** É frequentemente realizada em conjunto com o desenvolvimento ágil (como parte do desenvolvimento dirigido por testes - TDD).

Atividades

Discussão em Grupo (Estudo de Caso):

Analisar um cenário de mudança urgente (ex: falha grave de segurança em um sistema legado crítico). Discutir o processo de reparo de emergência (que prioriza a correção imediata em detrimento da documentação) e os riscos de longo prazo da inconsistência resultante.

Debater se as técnicas ágeis (como o Scrum) são adequadas para a evolução de sistemas. Argumentar que as práticas ágeis (como TDD e teste de regressão automatizado) são úteis para modificações, mas a transição de um time de desenvolvimento ágil para um time de manutenção pode ser problemática devido à perda de conhecimento implícito.

UniFAP


UniFAP


OBRIGADO
Arigato
Spasibo
Obrigado
Dankon
Maake
Multumesc
Ua Tsaug Rau Koj
Mochchakkeram
Kiitos
Merci
Raibh Maith Agat
Chokrane
Grazie
Terma Kasih
Asante
Mamana
Kia Ora
THANK YOU
Jusparxar
Welalin
Spasibo
Arigato

UniFAP

Bibliografia

SOMMERVILLE, Ian. Engenharia de software. 10. ed. São Paulo, SP: Pearson, 2018. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 15 ago 2025.

https://plataforma.bvirtual.com.br/Acervo/Publicacao/168127



Referências

CHATGPT. Expansão conceitual sobre integração entre COBIT, ITIL e PMBOK. OpenAI, 2025. Disponível em: https://chat.openai.com/. Acesso em: 11 ago. 2025.