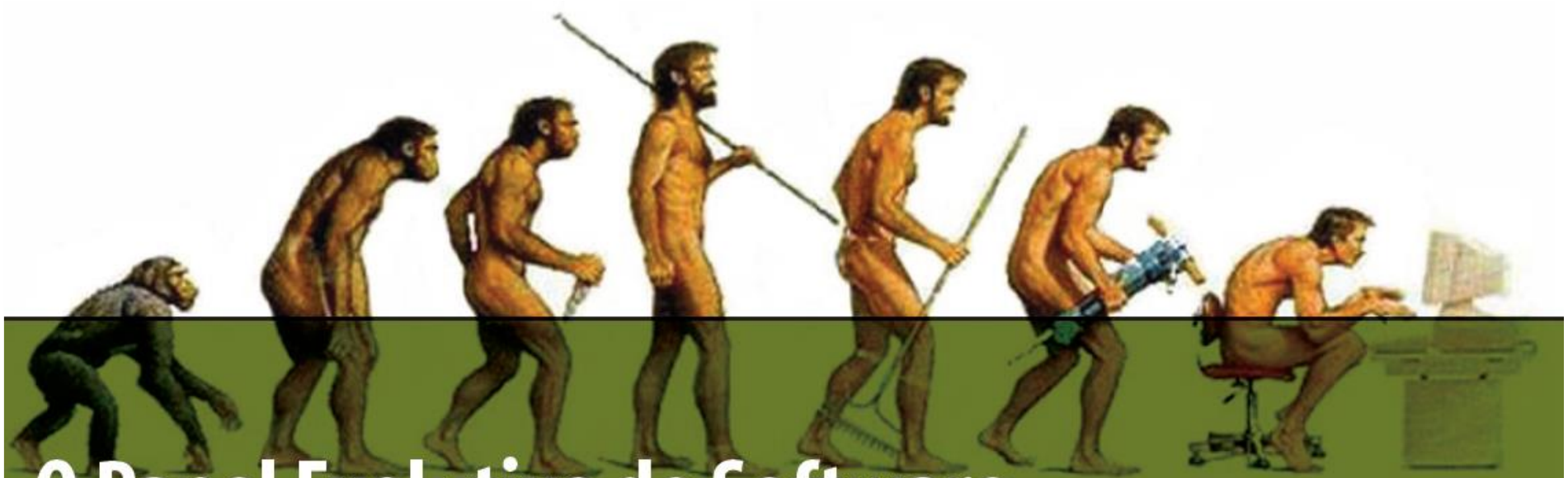
MANUTENÇÃO DE SOFTWARE

Evolução de Software

D.Sc. Jacilane Rabelo
jacilane.rabelo@ufc.br

Objetivo

- Explicar que a evolução é uma parte importante da engenharia de software e descrever os processo de evolução de software
- Compreender que a mudança é inevitável caso os sistemas de software devam permanecer úteis,
- Compreender os processos de evolução de software e as influências sobre esses processos



O Papel Evolutivo do Software

Exemplo: Em Teoria

1900

População: 48.369

Exemplo: Realidade

1900

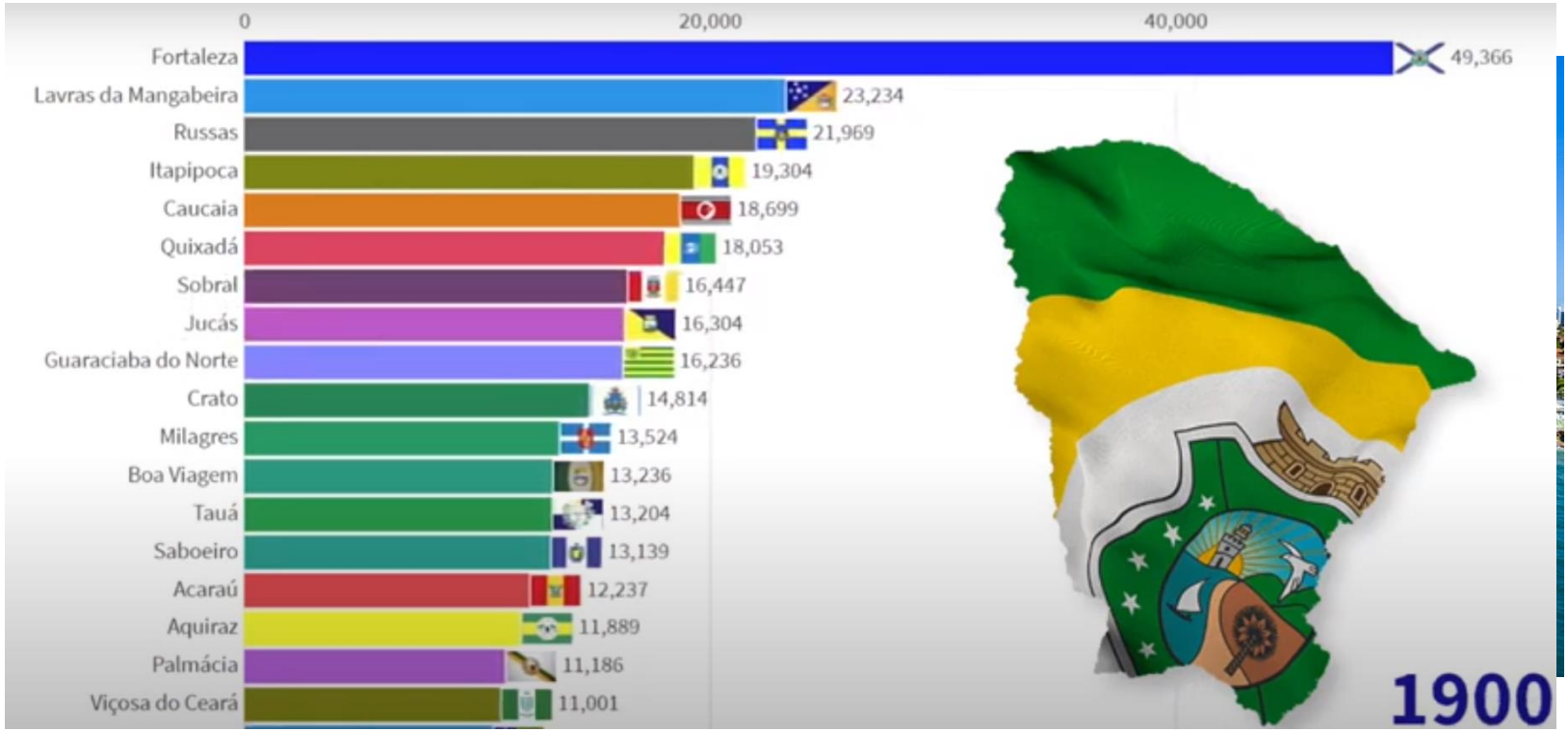
2020

População: 48.369

População: 2,698.300

Exemplo: Em Teoria

1900



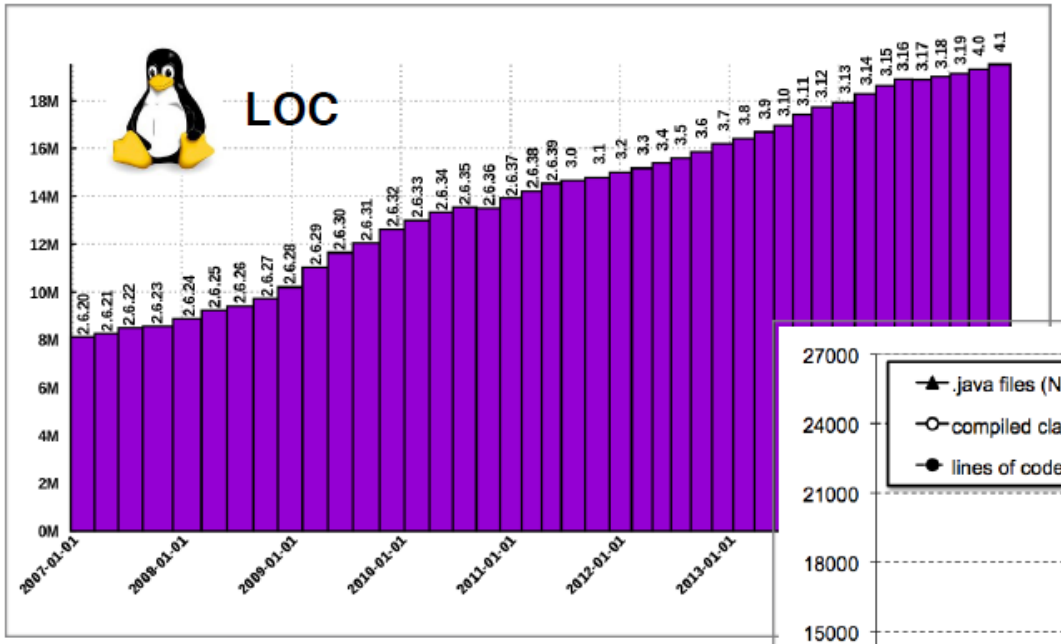
População: 656.675

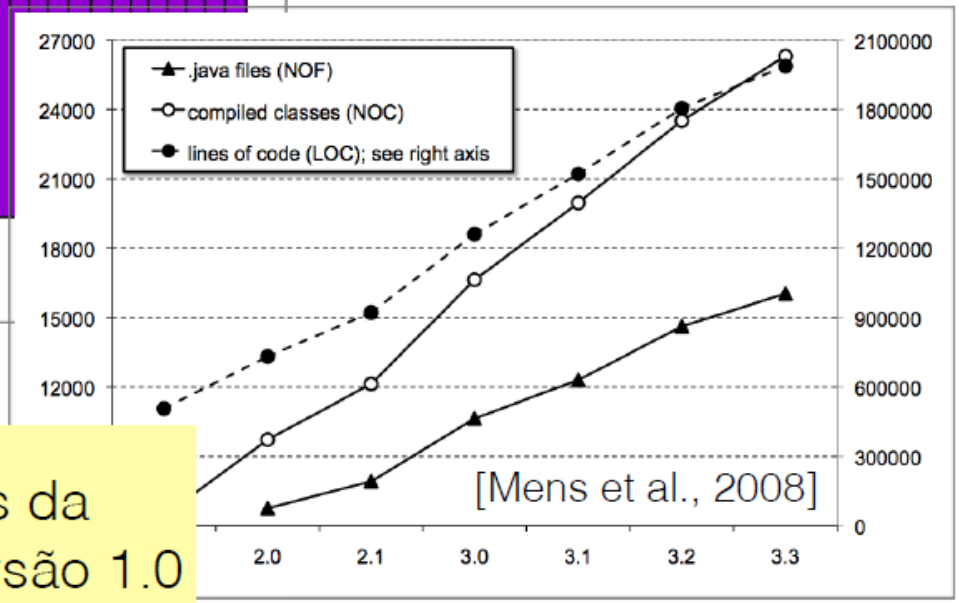
Exemplo: Realidade

2020

População: 9,306.254

Evolução de Software





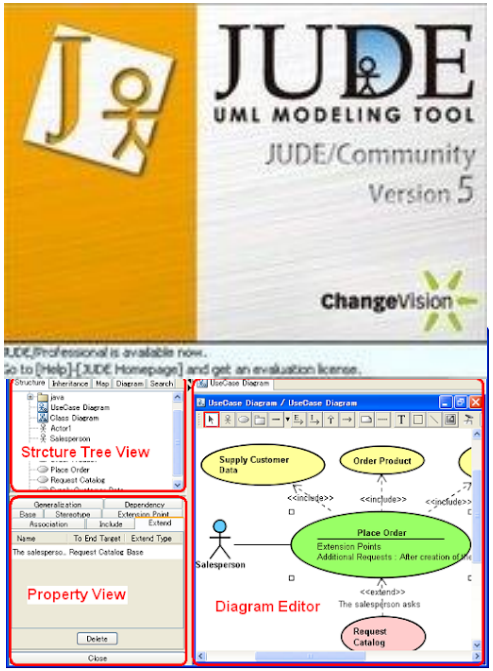
No Eclipse, 58% dos métodos da versão 2.0 não existiam na versão 1.0

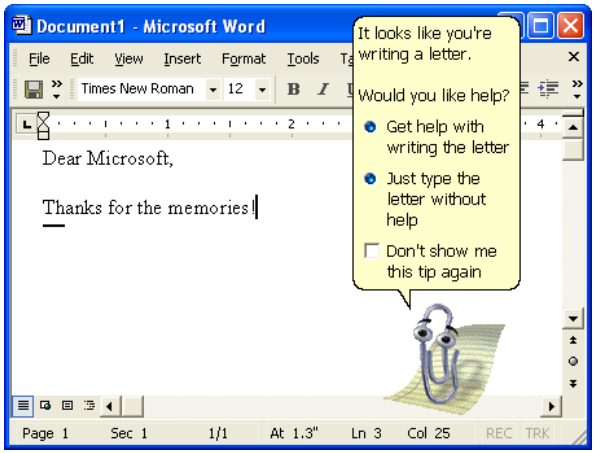
Evolução do Software

- Pense em qualquer produto de software que tenha lhe servido bem?

Evolução do Software

- Pense em qualquer produto de software que tenha lhe servido bem?





Evolução do Software

- Você utiliza todo o dia, mas vai ficando obsoleto
  - Apresenta problemas com frequência
  - Leva mais tempo para ser reparado
  - Já não representa a tecnologia mais recente utilizada

O que fazer?

Evolução do Software

- Você utiliza todo o dia, mas vai ficando obsoleto
  - Apresenta problemas com frequência
  - Leva mais tempo para ser reparado
  - Já não representa a tecnologia mais recente utilizada

O que fazer?

- Tenta remendá-lo?
- Tenta consertá-lo?
- Ampliar suas funcionalidades?

Evolução do Software

- Você utiliza todo o dia, mas vai ficando obsoleto
  - Apresenta problemas com frequência
  - Leva mais tempo para ser reparado
  - Já não representa a tecnologia mais recente utilizada

O que fazer?

- Tenta remendá-lo?
- Tenta consertá-lo?
- Ampliar suas funcionalidades?

Manutenção de Software

- Manutenção significa um **conjunto de modificações** realizadas no **software** que pode ocorrer durante o **desenvolvimento** ou **após a sua entrega**, ou seja, **durante a sua utilização**
- As modificações podem ser de várias formas e para atingir objetivos distintos. São utilizados para:
  - Correção de erros
  - Atualização do sistema
  - Aperfeiçoamento do software ou
  - Adaptação a uma nova realidade

- A manutenção de software, até pouco tempo, sempre foi considerada na etapa de desenvolvimento algo secundário, de pouco valor
- Era considerada por muitos como **uma fonte de gastos que comprometiam a criação de software**

Manutenção de Software

- Os sistemas eram elaborados **sem a preocupação de quem um dia eles precisariam sofrer alguma alteração** para se adequarem às novas necessidades do usuário
  - A manutenção era feita de **forma precária**
    - não existia um gerenciamento adequado para que fossem feitas as mudanças
  - Logo, as **novas mudanças poderiam gerar novos erros** que aumentariam ainda mais o tempo necessário para fosse feitas as correções desejadas pelo usuário

Evolução do Software

- A manutenção fica mais difícil à medida que os anos passam
  - Precisar reformá-lo
    - Mais funcionalidades
    - Melhor desempenho e mais confiabilidade
    - Manutenção mais fácil

Evolução do Software

- A manutenção fica mais difícil à medida que os anos passam
  - Precisar reformá-lo
    - Mais funcionalidades
    - Melhor desempenho e mais confiabilidade
    - Manutenção mais fácil

REENGENHARIA DE SOFTWARE

Redocumentar, refatorar, mudar a linguagem de programa para uma mais nova
- Não é mudar o software e sim sua estrutura!

Manutenção de Software

- Os sistemas eram elaborados **sem a preocupação de quem um dia eles precisariam sofrer alguma alteração** para se adequarem às novas necessidades do usuário
  - A **manutenção era feita de forma precária**,
    - não existia um gerenciamento adequado para que fossem feitas as mudanças
  - Logo, as **novas mudanças poderiam gerar novos erros** que aumentariam ainda mais o tempo necessário para fosse feitas as correções desejadas pelo usuário

Um problema chave para as organizações é **implementar e gerenciar a manutenção em sistemas legados**

Manutenção de Software

- Os sist**em a preocupação de quem um dia ele**ração para se adequarem às n
  - **ia precária**,
    - nã quado para que fossem feitas as mudanças
  - Logo, as **novas mudanças poderiam gera novos** aumentariam ainda mais o tempo necessário para f correções desejadas pelo usuário

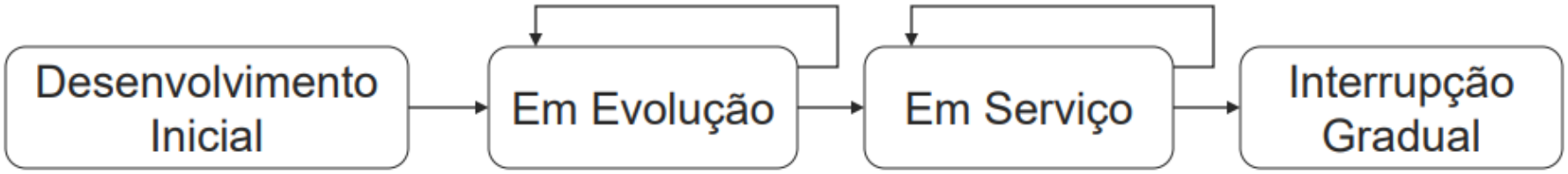
Esse cenário mudou atualmente?

Um problema chave para as organizações é implementar e gerenciar a manutenção em sistemas legados

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

Em Evolução vs. Em Serviço

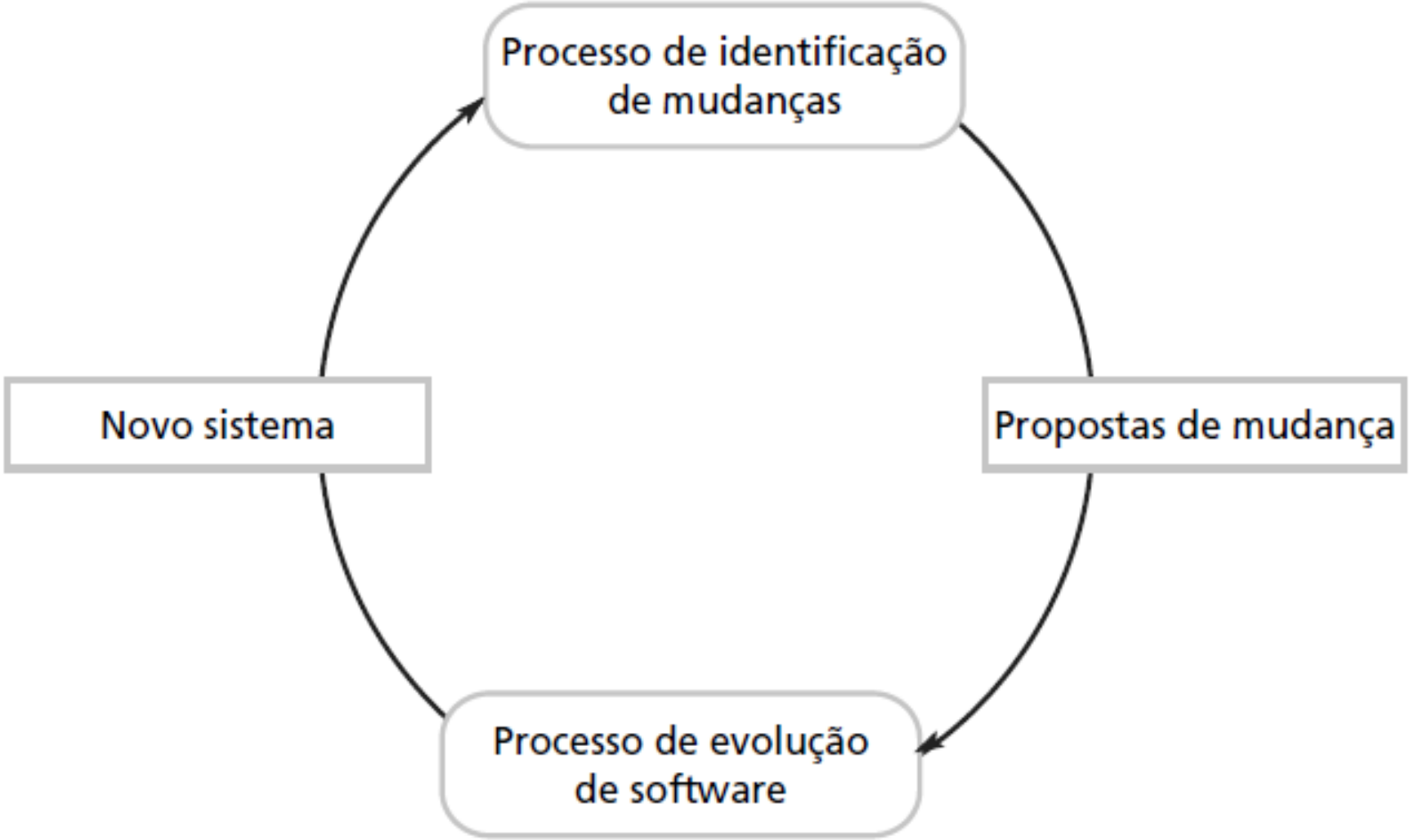


- **Em Evolução**
  - Mudanças significativas são feitas tanto na arquitetura quando nas funcionalidades
  - A estrutura tende a gradativamente se degradar
- **Em Serviço**
  - Apenas mudanças pequenas e essenciais ocorrem (software é pouco usado)

Processo de Evolução

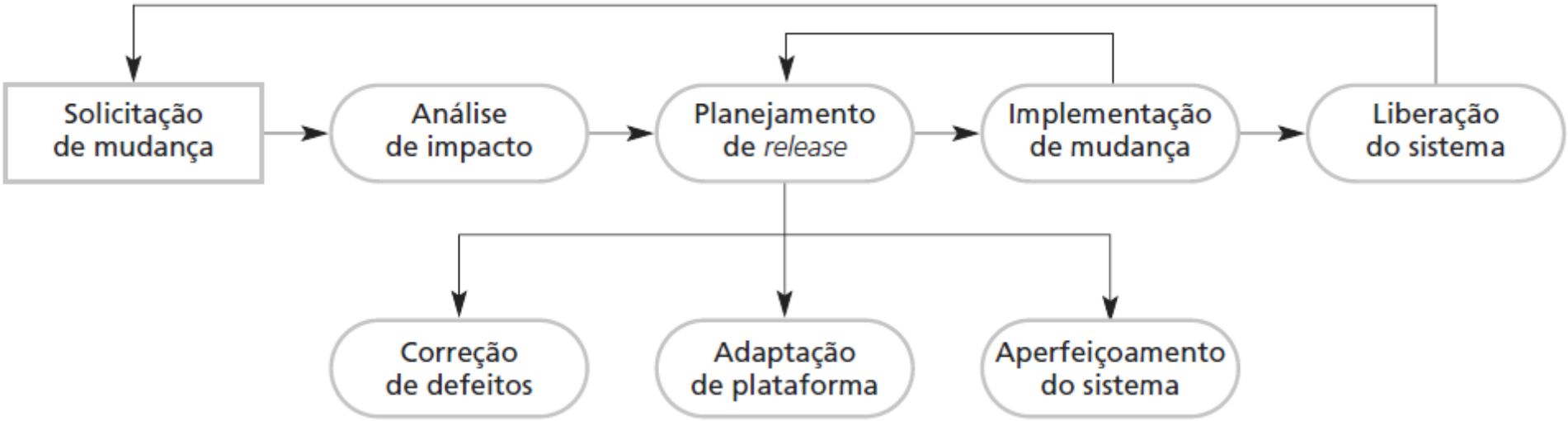
- A **evolução dos processos** de software pode variar dependendo ....
  - **tipo de software** que esteja sendo mantido
  - processos de desenvolvimento usados em uma organização
  - habilidades das pessoas envolvidas
- A evolução pode ser:
  - um **processo informal** em que as solicitações de mudança resultam, na maior parte, das conversas entre os usuários do sistema e desenvolvedores
  - um **processo formal** com documentação estruturada produzida em cada estágio do processo
- As propostas de mudança no sistema são os acionadores para a evolução

Processo de Identificação de Mudanças e de Evolução



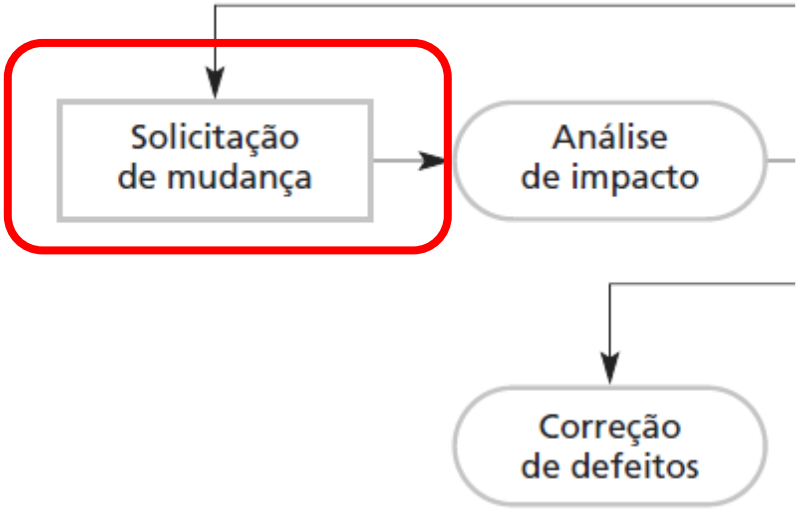
Modelo de Processo

- Cada solicitação de mudança é avaliada, planejada e implementada
  - O processo se repede em novas solicitações



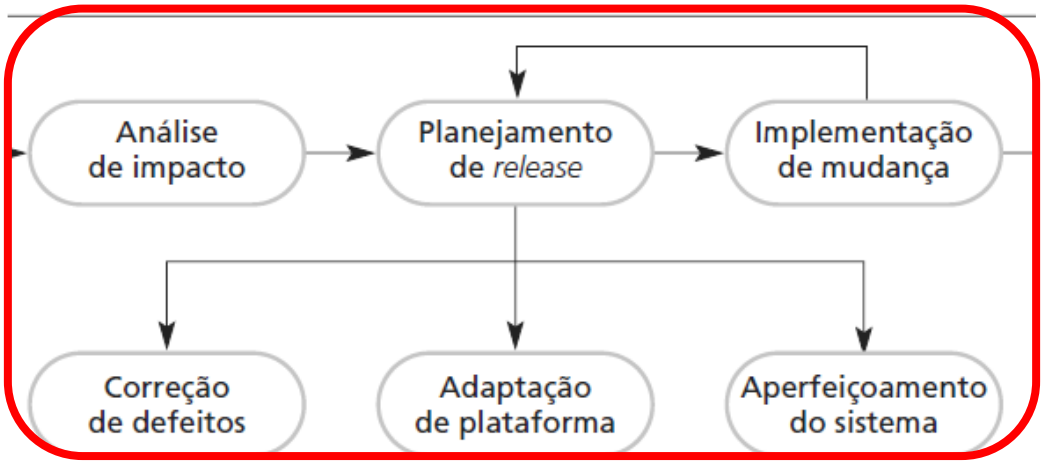
Pedidos de Mudança

- São feitos pelos utilizadores, cliente ou gerentes
- Na prática, alguns pedidos devem ser implementados urgentemente:
  - Reparar falhas
  - Mudanças no ambiente
  - Mudanças urgentes do negócio



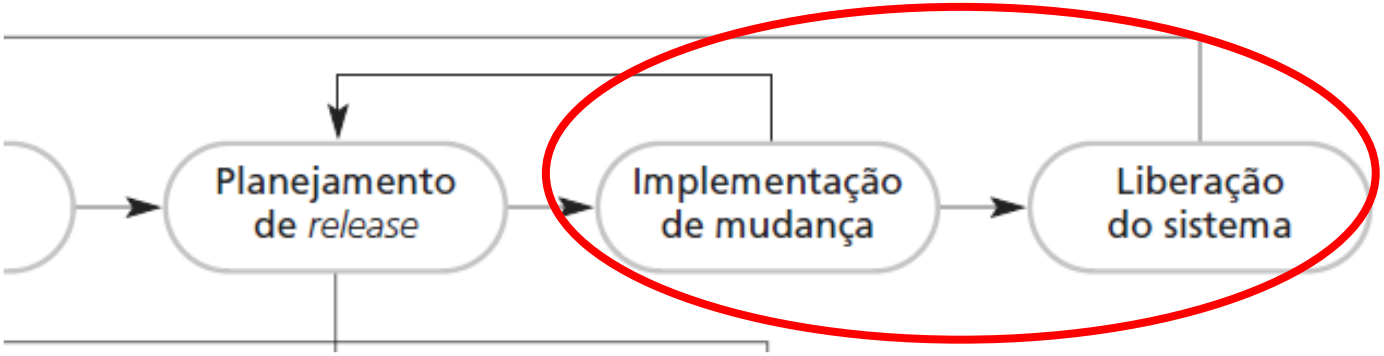
Análise e Planejamento

- **Análise de Impactos**
  - Avalia o custo e a porção do sistema afetado pela mudança solicitada
  - Decide se a mudança será aceita



- **Planejamento de Versões**
  - Caso sejam aceitas, diferentes tipos de mudanças podem ser agrupados para a próxima versão
  - Correção de defeitos, adaptações de plataforma e aperfeiçoamentos

Implementação e Liberação

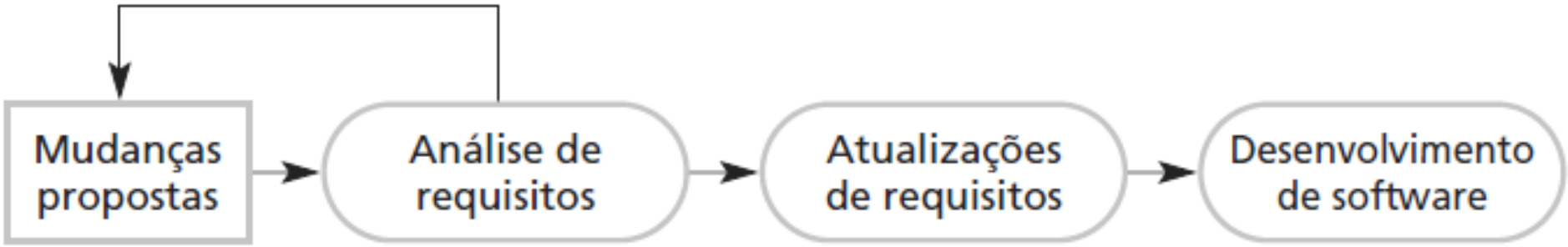


- **Implementação de Mudanças**
  - Geralmente, uma nova versão inclui um conjunto com várias mudanças solicitadas
- **Liberação do sistema**
  - Uma nova versão é liberada após validação com o cliente

O processo se repete com um novo conjunto de solicitação de mudanças

Implementação da Mudança

- A implementação da mudança deve atualizar a documentação correspondente
  - Especificação, modelos de projeto, implementação, testes, etc.



Compreensão de Programa

- A implementação da mudança envolve atividades semelhante ao desenvolvimento inicial do sistema
  - Uma diferença fundamental é **compreender o programa**
- Compreensão inclui
  - Entender como a **funcionalidade a ser alterada encontra-se implementada**
  - Avaliar **o impacto da mudança** em outras **partes do programa**

Previsão de Mudança

- A previsão de mudança preocupa-se em avaliar as partes do sistema que podem causar problemas e ter custos de manutenção altos
  - A **aceitação da mudança** depende da facilidade de manutenção dos componentes afetados pela mudança
  - **Implementar mudanças** degrada o sistema e reduz sua facilidade de manutenção
  - **Custos de manutenção** dependem do **número de mudanças** e os **custos de implementação** das mudanças dependem da **facilidade de manutenção dos componentes** do sistema

Previsão de Mudança

Que partes do sistema serão mais dispendiosas para se manter?

- Quais serão os custos manutenção durante o tempo de vida útil desse sistema?
- Quais os custos de manutenção desse sistema no próximo ano?

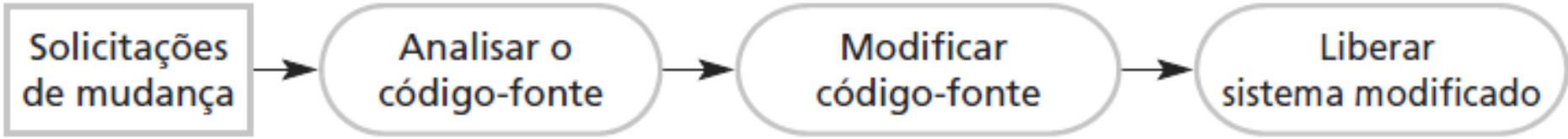
Que partes serão afetadas pelos pedidos de mudança?
Quantos pedidos de mudança podem ser esperados?

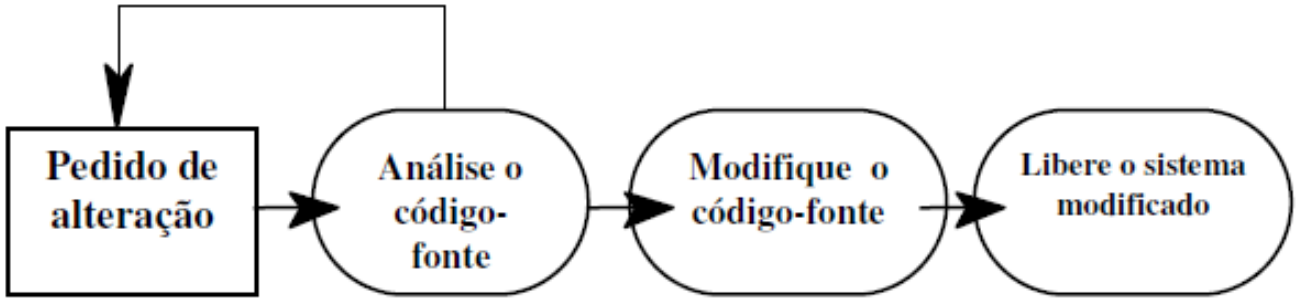
Situações Emergenciais

- Algumas **solicitações de mudanças podem ser urgente**, exemplo:
  - Se ocorrer um defeito grave que impede o funcionamento normal do sistema
  - Alterações do ambiente impedem o funcionamento do sistema
  - Alteração no negócio do cliente ou uma nova legislação
  - Entrada de um novo concorrente no mercado

Correção de Emergência

- Pode não haver tempo hábil para atualização da documentação
  - Ao invés de seguir as atividades “normais” de desenvolvimento, a alteração é feita diretamente no código





Refazer a Correção

- Código para correção de emergência geralmente tem baixa qualidade
- Portanto, a correção de emergência deveria ser idealmente re-implementada
  - E a documentação correspondente atualizada
- Na prática, este re-trabalho tem baixa prioridade e acaba esquecido

Refazer a Correção

- Código para correção de emergência geralmente tem baixa qualidade
- Portanto, a correção de emergência deveria ser idealmente re-implementada
  - E a documentação correspondente atualizada
- Na prática, este re-trabalho tem baixa prioridade e acaba esquecido

Sistemas Legados

Sistemas Legados

- **Antigos sistemas** que ainda **são essenciais para a empresa**
- Todos os **sistemas de software úteis inevitavelmente são modificados** quando as empresas passam por mudanças

Sistemas Legados

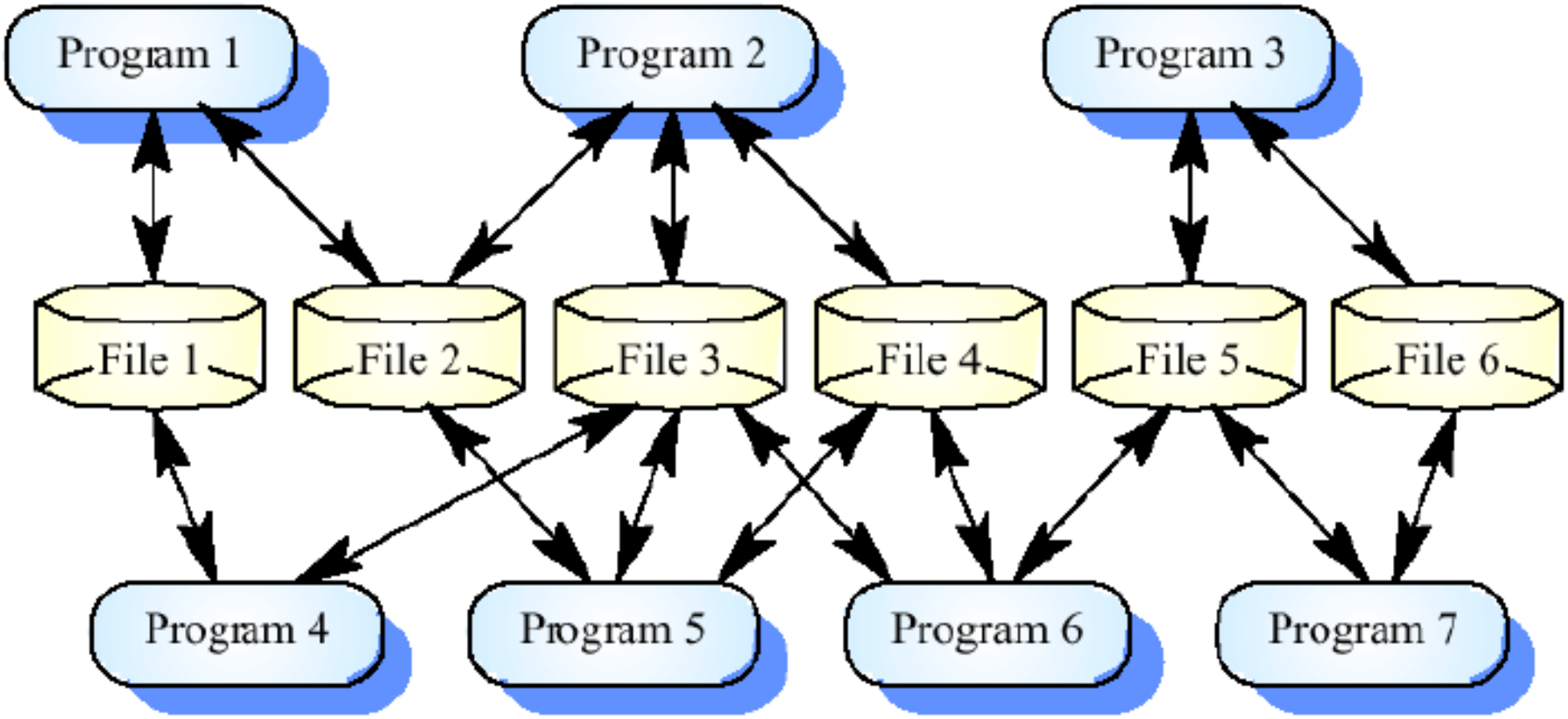
- **Antigos sistemas** que ainda **são essenciais para a empresa**
- Todos os **sistemas de software úteis inevitavelmente são modificados** quando as empresas passam por mudanças

- **Riscos de substituir um Sistema Legado**
  - Sistemas de legado, raramente, têm uma especificação completa
  - Os processos corporativos e o modo como os sistemas legados operam estão sempre entrelaçados
  - O sistema pode embutir regras corporativas que não estão documentadas formalmente em outro lugar
  - Desenvolvimento de software novo é arriscado e pode não ter êxito

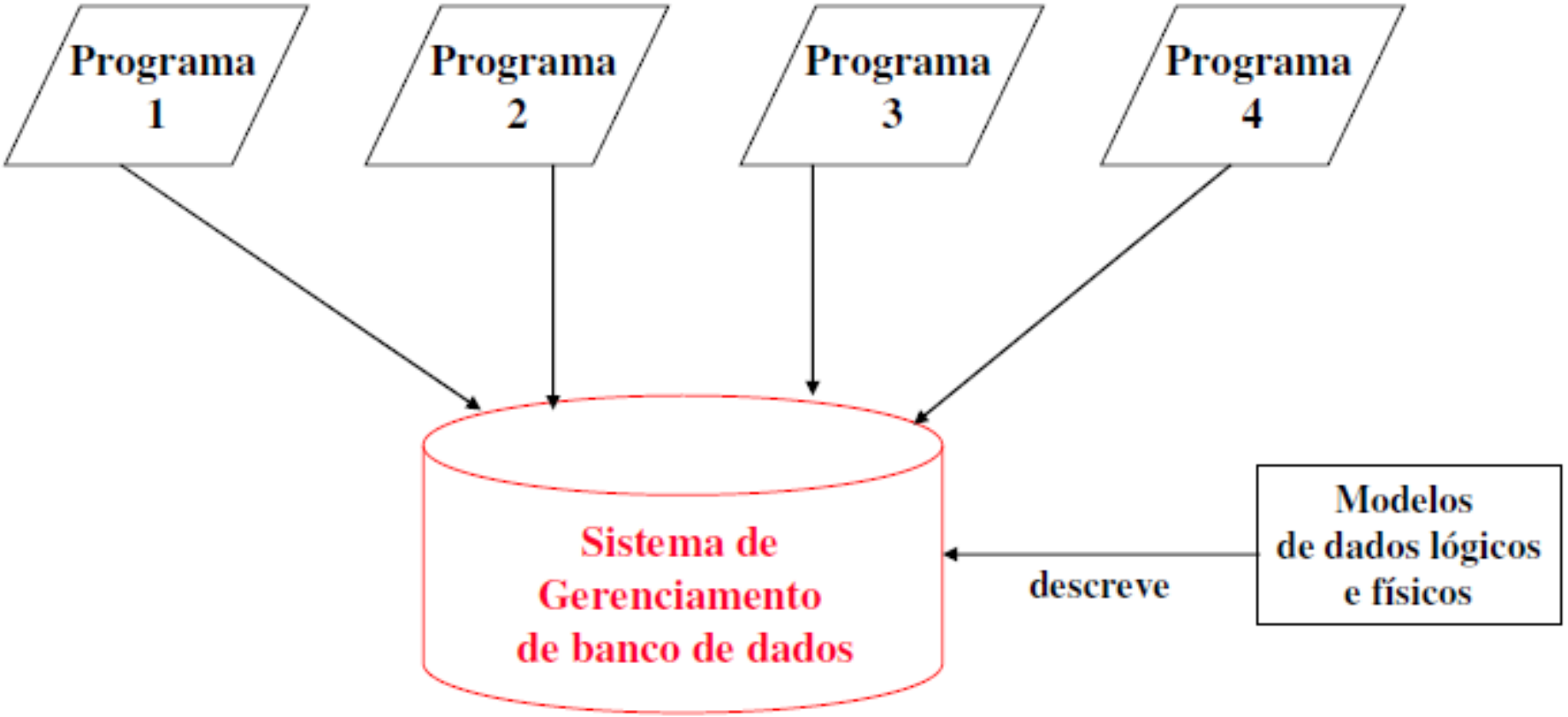
Mudanças em Sistema Legado

- Partes diferentes implementadas por diferentes equipes
- Linguagem de programação obsoleto
- A documentação de sistema está desatualizada
- A estrutura de sistema corrompida por muitos anos de manutenção (utilização de técnicas para economizar espaço ou aumentar a velocidade)
- Estruturas de arquivo usadas podem ser incompatíveis

Sistema Legados de Aplicação



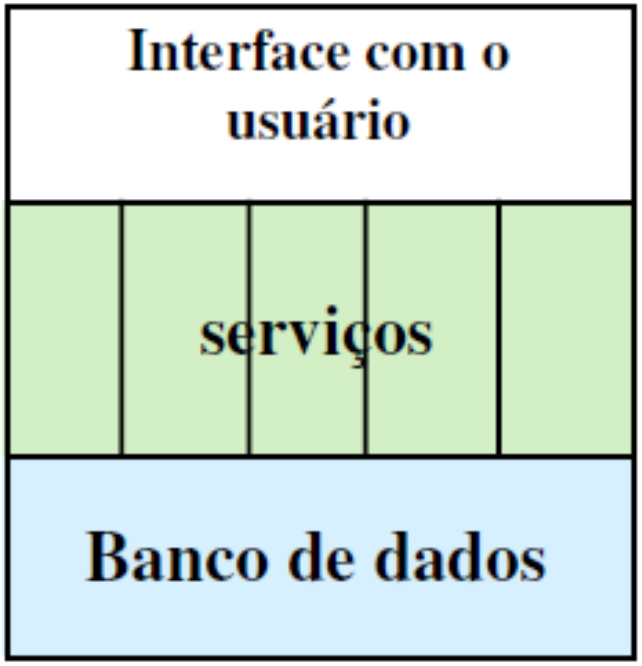
Sistema Centrados em Bancos de Dados



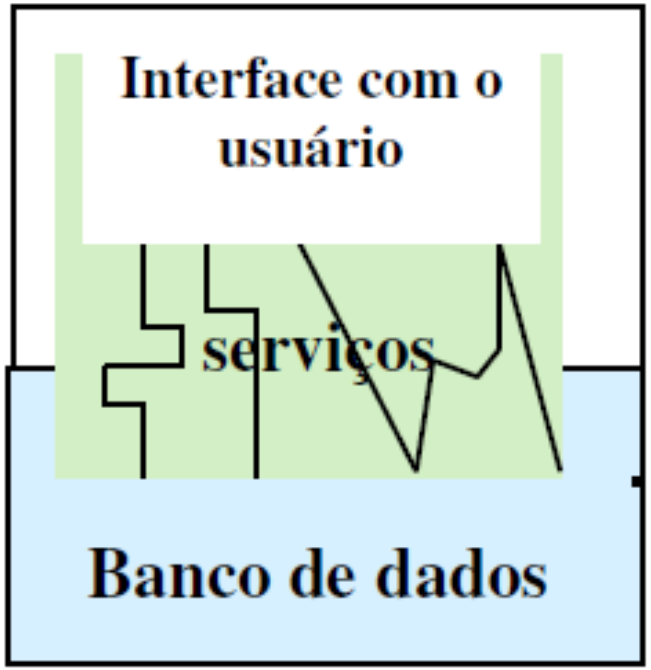
Avaliação dos Sistemas Legados

- Descartar completamente o sistema.
  - escolhida quando o sistema não estiver prestando uma contribuição efetiva aos processos
- Continuar a manter o sistema
  - sistema é necessário e sem grande número de modificações
- Transformar o sistema de alguma maneira para melhorar sua facilidade de manutenção
  - qualidade do sistema foi degradada por modificações regulares

Estrutura de Sistemas Legados



Modelo ideal



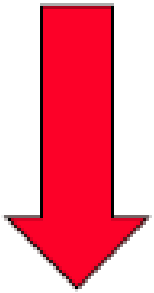
Sistema legados reais

Manutenção de Código Alienígena

- Programas com muitos fluxos de controle
- Módulos muito grandes
- Poucos linhas de comentários significativos
- Não existe nenhum outro elemento da configuração de software, além do código
- Nenhum membro do pessoal atual de manutenção trabalhou no desenvolvimento do programa
- Nenhuma metodologia de desenvolvimento foi aplicada

Manutenção de Código Alienígena

Solução



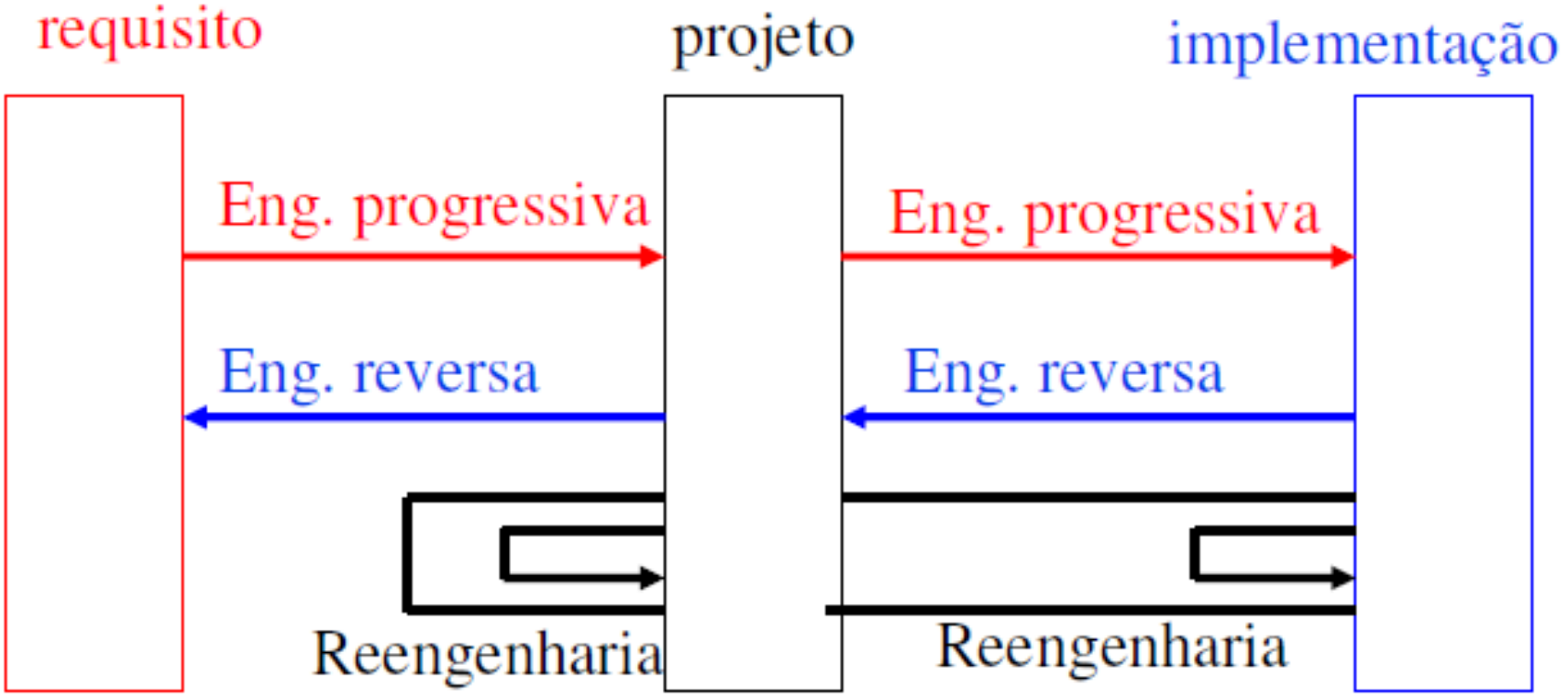
Engenharia Reversa e Reengenharia

Manutenção de Código Alienígena

- **ENGENHARIA REVERSA**
  - processo de análise de um software, partindo-se inicialmente da implementação para um nível mais alto de abstração

- **REENGENHARIA**
  - implica no exame e na alteração do software para reconstruí-lo em uma nova forma.

Manutenção de Código Alienígena



Dinâmica da Evolução (Leis de Lehman)

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

Leis de Lehman

1. Mudança Contínua
2. Complexidade Crescente
3. Auto-regulação
4. Estabilidade Organizacional
5. Conservação de Familiaridade
6. Crescimento Contínuo
7. Qualidade Declinante
8. Sistema de Feedback

Leis de Lehman

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

6. Crescimento Contínuo
7. Qualidade Declinante
8. Sistema de Feedback

Leis de Lehman

1. Mudança Contínua
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

7. Qualidade Declinante
8. Sistema de Feedback

Leis de Lehman

1. Mudança Contínua
2. Complexidade Crescente
3. **Autorregulação**

A **evolução de software** é um **processo autorregulável**

- 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

Leis de Lehman

1. Mudança Contínua
2. Complexidade Crescente
3. Autorregulação
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

Leis de Lehman

Durante a vigência de um sistema, a mudança incremental em cada release é aproximadamente constante

- 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

5. **Conservação de Familiaridade**
6. Crescimento Contínuo
7. Qualidade Declinante
8. Sistema de Feedback

Leis de Lehman

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

5. Conservação de Familiaridade
6. **Crescimento Contínuo**
7. Qualidade Declinante
8. Sistema de Feedback

Leis de Lehman

1. Mudança Contínua
2. Co
3. Au
4. Es
5. Co
6. Crescimento Contínuo
7. **Qualidade Declinante**
8. Sistema de Feedback

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

Leis de Lehman

1. Mudança Contínua
2. Complexidade Crescente

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

6. Crescimento Contínuo
7. Qualidade Declinante
8. **Sistema de Feedback**

O que é Manutenção de Software?

engenharia de SOFTWARE

Aplicabilidade das leis de Lehman

- Geralmente, as leis de Lehman parecem ser aplicáveis a sistemas customizados, grandes, desenvolvidos por grandes organizações.
  - ✓ Confirmado no início de 2000 pelo trabalho de Lehman sobre o projeto FEAST.
- Não está claro como devem ser modificadas para:
  - ✓ **Produtos de software devidamente embalados (rever tradução ???);**
  - ✓ Sistemas que incorporam um número significativo de componentes COTS;
  - ✓ Pequenas organizações;
  - ✓ Sistemas de médio porte.

O que é Manutenção de Software?

engenharia de SOFTWARE

Pontos importantes

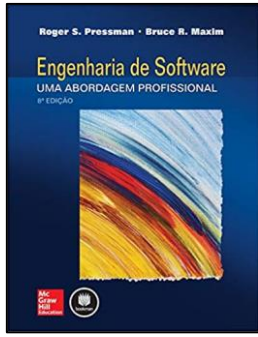
- O desenvolvimento e a evolução de software podem ser pensados como processo iterativo e integrado, que pode ser representado usando um modelo em espiral.
- Para sistemas customizados, os custos de manutenção de software geralmente excedem os custos de desenvolvimento de software.
- O processo de evolução do software é impulsionado pelas solicitações de mudança e inclui a análise do impacto da mudança, planejamento de release e implementação da mudança.
- As leis de Lehman, assim como a noção de que a mudança é contínua, descreve uma série de considerações derivadas de estudos de longo prazo, de evolução de sistema.

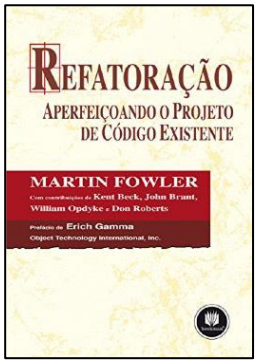
Exercício

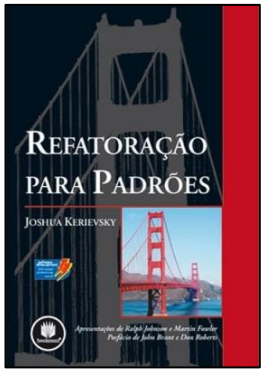
- Qual foi o último projeto de software em que você trabalhou? Foi um projeto comercial, um projeto de graduação ou um projeto pessoal? Escreva uma avaliação crítica do modelo de ciclo de vida com o qual você trabalhou. Foi bem estruturado ou ad hoc? Você trabalharia com um modelo diferente se iniciasse este projeto novamente?

Referências Bibliográficas

- Refatoração para padrões. KERIEVSKY, J.
- Refatoração: Aperfeiçoando O Projeto De Código Existente. FOWLER, MARTIN
- Software Maintenance Management: Evaluation And Continuous Improvement. APRIL, ALAIN; ABRAN, ALAIN
- IEEE Std 14764-2006, Software Engineering – Software Life Cycle Processes Maintenance.







- **Perguntas?**

E-mail: jacilane.rabelo@ufc.br