

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte  
Departamento Acadêmico de Gestão e Tecnologia da Informação  
Curso de Tecnologia em Análise e Desenvolvimento de Sistemas

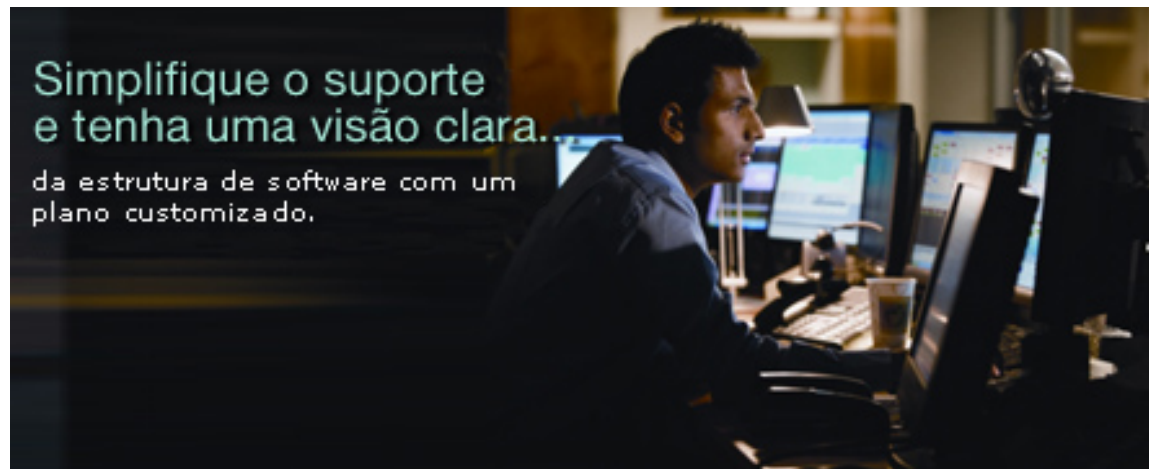
# Suporte, Treinamento e Manutenção de Software

Prof. Fellipe Aleixo ([fellipe.aleixo@ifrn.edu.br](mailto:fellipe.aleixo@ifrn.edu.br))

# SUPORTE E TREINAMENTO

Simplifique o suporte  
e tenha uma visão clara.

da estrutura de software com um  
plano customizado.



# Suporte

- Canal de comunicação pelo qual o cliente relata a ocorrência de um problema com o software, solicitando a sua solução
- Função do suporte:
  - Identificar a natureza do problema
  - Caso se trate de um “problema conhecido”, auxiliar ou executar a solução do problema
  - Caso contrário, fazer o encaminhamento do mesmo para a equipe de desenvolvimento

# Suporte

- O atendimento pode ser presencial ou remoto (telefone ou Internet)
- Pode ter diferentes níveis de disponibilidade (definidas em contrato)
  - Por exemplo: “24 por 7” = 24 horas, 7 dias por semana
  - Também pode definir um tempo máximo para resposta (também definido em contrato)

# Suporte

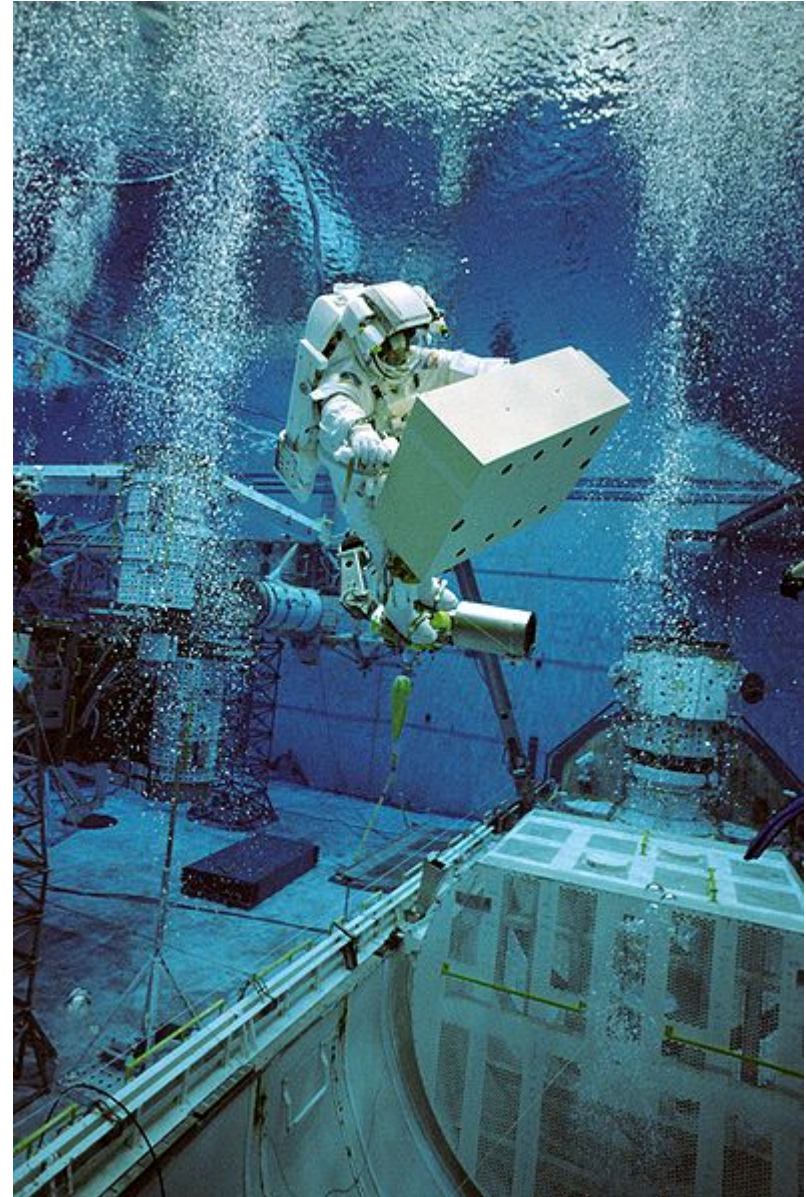
- É uma atividade de alto custo
- Só deve haver o comprometimento (contrato) se a equipe possuir as condições para tal
- O contato direto com o usuário exige habilidades especiais e treinamento específico

# Treinar Usuários Também é Função da Equipe de Desenvolvimento

- A equipe de desenvolvimento precisa entender:
  - Não vale de nada o tempo de planejamento e desenvolvimento na criação do software se ninguém irá usá-lo
  - As pessoas resistem à mudança e evitam se aventurar em áreas pouco familiares
- Daí a importância de incluir treinamento na fase de desenvolvimento
- Caso contrário os problemas permanecerão para a próxima etapa

# Treinamento

- Pode evitar:
  - Perda de tempo
  - Irritação e rejeição
  - Prejuízos
- Leva a os usuários finais a se sentirem parte do processo



# MANUTENÇÃO

É natural que um software sofra mudanças e evolua. Uma vez entrando em produção, defeitos são descobertos, o ambiente operacional muda, e novos requisitos dos usuários vêm a tona.





# Manutenção de Software

- Pode fazer parte do ciclo de desenvolvimento – período de “garantia”
- Envolve:
  - Melhoria e otimização do software
  - Reparo de defeitos
  - Inclusão de novas funcionalidades para melhorar a aplicabilidade e usabilidade do software

# Implicações no Desenvolvimento

- A manutenção se torna mais fácil se alguns aspectos forem levados em consideração no desenvolvimento:
  - Consistência entre projeto e implementação
  - Utilização de padrões (projeto e código)
  - Organização da documentação
  - Adoção de algum mecanismo para **documentar** e **rastrear** os defeitos e deficiências

# Necessidade de Manutenção

- As mudanças no sistema podem envolver ações corretivas e não-corretivas
- É realizada para:
  1. Corrigir falhas
  2. Melhorar o projeto
  3. Implementar melhorias
  4. Definir interfaces com outros sistemas
  5. Adaptar o sistema para poder utilizar diferentes hardwares, softwares, meios de comunicação e dar suporte a novas características,
  6. Migrar software legado
  7. “Aposentar” o software

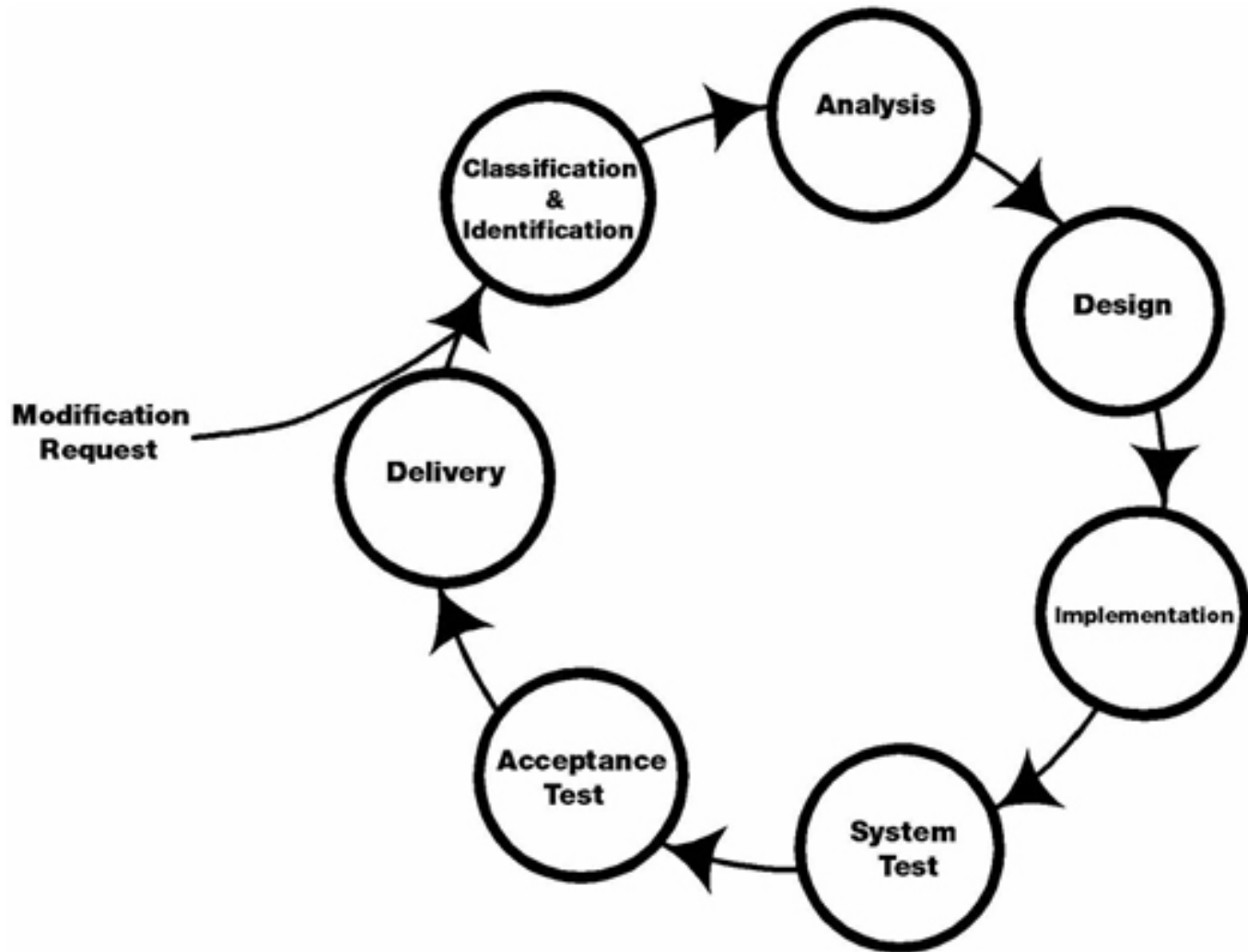
# Necessidade de Manutenção

- As ações de manutenção possuem quatro características chave:
  1. Manter o controle das funcionalidades do dia a dia do software
  2. Manter o controle das modificações do software
  3. Evoluir as funcionalidades já existentes
  4. Prevenir que a performance do software caia para níveis inaceitáveis

# Categoria de uma Manutenção

- **Reativa**
  - **Corretiva** → modificação do software, realizada após a descoberta de problemas
  - **Adaptativa** → modificação do software em virtude de mudanças no ambiente operacional
- **Proativa**
  - **Perfectiva** → modificações para melhorar a performance ou manutenibilidade
  - **Preventiva** → modificação para detectar e corrigir “erros” antes de se tornarem “falhas”

# Processo de Manutenção



# Técnicas de Manutenção

- **Compreensão do sistema** → gastar tempo lendo e entendendo o software e sua documentação
- **Reengenharia** → examinar e alterar o software para repensar o mesmo em uma nova estrutura, para em seguida implementar essa nova estrutura
- **Engenharia reversa** → analisar o software para identificar os componentes e os relacionamentos – representações de mais alto nível de abstração
- **Separação estática** → Identificar as áreas do código afetadas por alguma variável

# Planejar...Antes de Tudo



- Cada ação a ser desempenhada pela equipe, deve ser precedida do seu planejamento
  - Estratégia principal
  - Soluções de contorno
  - Acordar com os usuários
  - Comunicar as decisões a todos



Corpo de Conhecimento em Engenharia de Software  
(*Software Engineering Body of Knowledge*)

# **SWEBOK**




- Disponível na Internet a versão HTML
  - <http://www.computer.org/portal/web/swebok/htmlformat>

# SWEBOK



# SWEBOK – Tabela de Conteúdo

- **COPYRIGHT**
- **FOREWORD**
- **ASSOCIATE EDITORS**
- **INDUSTRIAL ADVISORY BOARD**
- **PANEL OF EXPERTS**
- **REVIEW TEAM**
- **PREFACE**
- **CHAPTER 1: INTRODUCTION TO THE GUIDE**
- **CHAPTER 2: SOFTWARE REQUIREMENTS**
- **CHAPTER 3: SOFTWARE DESIGN**
- **CHAPTER 4: SOFTWARE CONSTRUCTION**
- **CHAPTER 5: SOFTWARE TESTING**
- **CHAPTER 6: SOFTWARE MAINTENANCE** 
- **CHAPTER 7: SOFTWARE CONFIGURATION MANAGEMENT**
- **CHAPTER 8: SOFTWARE ENGINEERING MANAGEMENT**
- **CHAPTER 9: SOFTWARE ENGINEERING PROCESS**
- **CHAPTER 10: SOFTWARE ENGINEERING TOOLS AND METHODS**
- **CHAPTER 11: SOFTWARE QUALITY**
- **CHAPTER 12: RELATED DISCIPLINES OF SOFTWARE ENGINEERING**
- **APPENDIX A: KNOWLEDGE AREA DESCRIPTION SPECIFICATIONS FOR THE IRONMAN VERSION OF THE GUIDE TO THE SOFTWARE ENGINEERING BODY OF KNOWLEDGE**
- **APPENDIX B: EVOLUTION OF THE GUIDE TO THE SOFTWARE ENGINEERING BODY OF KNOWLEDGE**
- **APPENDIX C: ALLOCATION OF IEEE AND ISO SOFTWARE ENGINEERING STANDARDS TO SWEBOK KNOWLEDGE AREAS**
- **APPENDIX D: CLASSIFICATION OF TOPICS ACCORDING TO BLOOM'S TAXONOMY**