

# Engenharia de Software

## **Introdução**

**Tópicos sobre Eng. Sw ...**

GRD/2010/2

**Prof. Luís Fernando Garcia**

**LUIS@GARCIA.PRO.BR**

# Engenharia de Software

- Onipresença/DEPENDÊNCIA de “computadores”
- Computador “=” **Software**
- Aspectos POSITIVOS
- **Aspectos NEGATIVOS**
- (<http://www5.in.tum.de/~huckle/bugse.html>)

# Engenharia de Software



- Início em 24:30
- Final em 30:30

# Engenharia de Software

## SOFTWARE HORROR STORIES



[My Home Page](#) [Comp. Risks](#) [Verification Course](#) [Submit a Story!](#)

The time is now **U.S.N.O.**

1. The Mars Climate Orbiter crashed in September 1999 because of a "silly mistake": wrong units in a program. [Story](#) [Story](#) [Report](#)
2. The 1988 shooting down of the Airbus 320 by the USS Vincennes was attributed to the cryptic and misleading output displayed by the tracking software. [Story](#) [More](#)
3. Death resulted from inadequate testing of the London Ambulance Service software. [Story](#)
4. Several 1985-7 deaths of cancer patients were due to overdoses of radiation resulting from a race condition between concurrent tasks in the Therac-25 software. [Report](#) [Report](#) [Story](#) [More](#) [More](#) [More](#)
5. Errors in medical software have caused deaths. Details in B.W. Boehm, "Software and its Impact: A Quantitative Assessment," *Datamation*, 19(5), 48-59(1973).
6. An Airbus A320 crashes at an air show. [Story](#)
7. A China Airlines Airbus Industrie A300 crashes on April 26, 1994 killing 264. Recommendations include software modifications. [Summary](#)
8. The British destroyer H.M.S. Sheffield was sunk in the Falkland Islands war. According to one report, the ship's radar warning systems were programmed to identify the Exocet missile as "friendly" because the British arsenal includes the Exocet's homing device and allowed the missile to reach its target, namely the Sheffield. From "The development of software for ballistic-missile defense," by H. Lin, *Scientific American*, vol. 253, no. 6 (Dec. 1985), p. 48.
9. An error in an aircraft design program contributed to several serious air crashes. From P. Naur and B. Randell, eds., *Software Engineering: Report on a Conference Sponsored by the NATO Science Committee*, Brussels, NATO Scientific Affairs Division, 1968, p. 121.

[http://www.cs.tau.ac.il/  
~nachumd/verify/horror.html](http://www.cs.tau.ac.il/~nachumd/verify/horror.html)

# Engenharia de Software



Bielefeld University - Faculty of Technology  
**Networks and Distributed Systems**  
Research group of Prof. Peter B. Ladkin, Ph.D.

- Home
- People & Info
- Publications
- Research & Projects
- Why-Because Analysis
- Lectures
- Compendium CRICA
- System Safety Society
- Bieleeschweig Workshops
- RVS Blog
- Photo Gallery
- Contact & Impressum

## Computer-Related Incidents with Commercial Aircraft

### The Incidents and Accidents

- [My Favorite CRICA Nightmare](#)
- [TAM A320 runway overrun accident, Sao Paulo Congonhas airport, Brazil, July 2007](#)
- [B737-800/Embraer Legacy midair collision, Amazon jungle, Brazil, September 2006](#)
- [B777, anomalous flight behavior and partial Loss of Control, off Perth, W. Australia, August 2005](#)
- [A320, runway overrun on landing, Taipei-Sungshan airport, Taiwan, October 2004](#)
- [Unknown FRW aircraft type, Byzantine failures in Flight Control System \(FCS\), n.d.](#)
- [A320, braking problem on touchdown, Cardiff, August 2003](#)
- [British Mediterranean Airways, A320, false navigation data on non-precision approach, Addis Ababa, March 2003](#)
- [Singapore Airlines, B747-400, Primary Flight Display information loss, January 2003](#)
- [B717, in-flight engine shutdown due to electronic fault, Launceston, Australia, October 2002](#)
- [Tu154m and B757, Midair Collision, Überlingen, Lake Constance, Germany, July 2002](#)
- [B717, dual electronic Flight Management System failure, nr. Launceston, Australia, December 2001](#)
- [American Airlines Flight 587, In-Flight Break-Up, New York City, November 2001](#)
- [Air Transat Flight 236, The Azores Glider, August 2001](#)
- [A330, landing difficulties, Melbourne, Australia, August 2001](#)
- [A320 cross-wired sidestick incident, Frankfurt am Main, March 2001](#)
- [Comair, Embraer Brasilia turboprop, Primary Flight Display information loss, 19 March 2001](#)
- [Iberia A320, hard landing, Bilbao, Spain, February 2001](#)
- [Beach 300, CFIT, Blumberg near Donaueschingen, Germany, October 2000](#)
- [Air France Concorde SST, Fire and Loss of Control, Paris, France, July 2000](#)
- [Air New Zealand Flight 60, B767, false glideslope capture, Faleola, Samoa, July 2000](#)
- [A340/A330 airprox under RVSM, North Atlantic Track E, October 2000](#)
- [Various aircraft, airprox caused by TCAS RA manoeuvring, Trasadingen, September 2000](#)
- [British Airways/Korean Air near miss, Chinese airspace, 28 June 1999](#)
- [High-Intensity Electromagnetic Radiated Fields: TWA 800, and Swissair 111 \(1996 and 1998\)](#)
- [Air UK Leisure A320 braking problem and runway overrun, Ibiza, May 1998](#)

# Engenharia de Software

- “Software”
  - Abstrato ... Intangível
  - Produto “complicado” ... “diferente”
  - Sem limitações/“leis da física”
  - Complexo ...

# Engenharia de Software

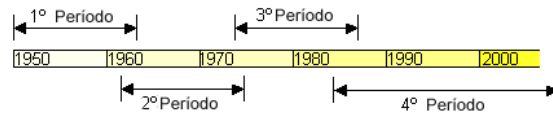


Figura I.1 Evolução do Software

## 1º Período:

- orientação para lotes; e
- distribuição limitada.

## 3º Período:

- sistemas distribuídos;
- "inteligência" embutida;
- baixo custo de hardware; e
- impacto do consumo.

## 2º Período:

- multiusuário;
- tempo real;
- banco de dados; e
- produto software.

## 4º Período:

- poderosos sistemas de mesa;
- tecnologias orientada a objetos;
- sistemas especialistas;
- redes neurais artificiais;
- computação paralela; e
- redes de computadores.

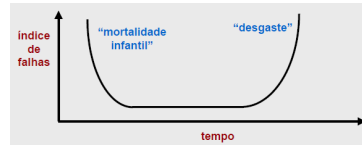
# Engenharia de Software

- **Questões a pensar:**
- **Construção**
  - Processos + Ferramentas + Atores
- **Efeitos/conseqüências**
- **Custo (\$)**
- **Tempo**

# Engenharia de Software

- Questões a pensar 2:

- SW é desenvolvido, não “fabricado”
- SW não desgasta (?)
- SW sob encomenda ...  
Indústria montagem/componentes
- Natureza “MUTÁVEL” do SW



# Engenharia de Software

- Como “desenvolver” software?

- Enfoque “ARTESANAL” (informal)
- Enfoque ENGENHARIA

# *Engenharia de Software*



# *Engenharia de Software*

- Enfoque ARTESANAL
- “Na falta de padrões expressivos, uma nova indústria, como a de software, passa a depender de FOLCLORE” (Tom de Marco)
- CTRL-C/CTRL-V ...
- F8 .. Tentativa e erro ...

# *Engenharia de Software*

---

- Enfoque ENGENHARIA
- DISCIPLINA ... OK!
- EQUIPES .. OK!
- ADAPTABILIDADE ... ??
- AGILIDADE ...??

# *Engenharia de Software*

---

- “O estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um software que seja confiável e que funcione eficientemente em máquinas reais” ... Fritz Bauer, 1969

## *Engenharia de Software*

---

- “Aplicação de uma abordagem sistemática, disciplinada e quantificável para o desenvolvimento, operação e manutenção do software” ...  
IEEE

## *Engenharia de Software*

---

- “Engenharia é a aplicação sistemática de conhecimentos científicos na criação e construção de soluções com um bom custo-benefício para a resolução de problemas práticos da sociedade” ... SEI

# Engenharia de Software

## O que é Engenharia de Software ?

Engenharia de Software é uma área do conhecimento da informática voltada para a especificação, desenvolvimento e manutenção de sistemas de software aplicando tecnologias e práticas de ciências da computação, gerencia de projetos e outras disciplinas, objetivando organização, produtividade e qualidade na produção de software.

# Engenharia de Software

- *Conferência da NATO (1968) – Crise de Software*

### *Problemas detectados:*

- *Cronogramas não observados.*
  - *Projetos abandonados.*
  - *Módulos que não operam corretamente quando combinados.*
  - *Programas que não fazem exatamente o que era esperado.*
  - *Sistemas tão difíceis de usar que são descartados.*
  - *Sistemas que simplesmente param de funcionar.*
- 
- *Passados mais de 40 anos, o que mudou? ☺*

# Engenharia de Software



**SWEBOK**  
IEEE Computer Society

## Guide to the Software Engineering Body of Knowledge

Get the 2004 SWEBOK Guide

- » [HTML](#) (free)
- » [PDF](#)
- » [Book](#)

**SWEBOK**

## Guide to the Software Engineering Body of Knowledge

2004 Version

Executive Editors  
Alain Abran, École de technologie supérieure  
James W. Moore, The MITRE Corp.

Editors  
Pierre Bourque, École de technologie supérieure  
Robert Dupuis, Université du Québec à Montréal

IEEE  
COMPUTER  
SOCIETY

IEEE

MITRE

NIST

Raytheon

SAP

BOEING

Canadian Council of Professional Engineers  
Conseil canadien des ingénieurs

ARC-CMRC

IVÉ

Construx

Advancing software project success

Rational

the software development company

Project managed by:

UQAM

École de technologie supérieure

# Engenharia de Software

**Table 1** The SWEBOK Knowledge Areas (KAs)

Software requirements
Software design
Software construction
Software testing
Software maintenance
Software configuration management
Software engineering management
Software engineering process
Software engineering tools and methods
Software quality

# Engenharia de Software

**Table 2** Related disciplines

- |  |   |
|--|---|
| <ul style="list-style-type: none"><li>• Computer engineering</li><li>• Computer science</li><li>• Management</li><li>• Mathematics</li></ul> | <ul style="list-style-type: none"><li>• Project management</li><li>• Quality management</li><li>• Software ergonomics</li><li>• Systems engineering</li></ul> |
|--|---|

# Responsabilidade e ética

The screenshot shows the ACM website page for the Software Engineering Code of Ethics and Professional Practice. The page features the ACM logo and navigation menu on the left, and the main content area on the right. The main content area includes a title, a description of the code, and a preamble.

Association for Computing Machinery  
Advancing Computing as a Science & Profession

you are here: home → about → software engineering code of ethics and professional practice

ACM myACM

- Home
- Special Interest Groups (SIGs)
- Publications
- Membership
- Digital Library
- Educational Activities
- Online Books & Courses
- Career & Job Center
- Chapters
- Conferences
- Calendar of Events
- Awards
- Public Policy
- Online Communities

## Software Engineering Code of Ethics and Professional Practice

Software Engineering Code of Ethics and Professional Practice (Version 5.2) as recommended by the Engineering Ethics and Professional Practices and jointly approved by the ACM and the IEEE-CS as the engineering.

- [Short Version](#)
- [Full Version](#)

### Software Engineering Code of Ethics and Professional Practice (Short Version)

#### PREAMBLE

The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that details of how these aspirations change the way we act as software engineering professionals. Without tedious; without the details, the aspirations can become high sounding but empty; together, the aspiratic

Software engineers shall commit themselves to making the analysis, specification, design, development, to respected profession. In accordance with their commitment to the health, safety and welfare of the public, Principles:

1. **Público.** Engenheiros de Software devem atuar consistentemente com os interesses públicos.
2. **Clientes e empregados.** Engenheiros de Software devem atuar de modo a atender os melhores interesses dos seus clientes e empregados, consistentemente com os interesses públicos.
3. **Produto.** Engenheiros de Software devem assegurar que seus produtos e modificações relacionadas atendam os melhores padrões profissionais possíveis.
4. **Julgamento.** Engenheiros de Software devem manter a integridade e independência nos seus julgamentos profissionais.
5. **Administração.** Administradores e líderes de Engenharia de Software devem aderir e promover uma abordagem ética ao gerenciamento do desenvolvimento e manutenção de software.
6. **Profissão.** Engenheiros de Software devem desenvolver a integridade e reputação da profissão consistentemente com os interesses do público.
7. **Coleguismo.** Engenheiros de Software devem ser justos e dispostos a auxiliar seus colegas.
8. **Identidade.** Engenheiros de Software devem participar do aprendizado de suas vidas valorizando a prática da sua profissão e devem promover uma abordagem ética à prática da profissão.”

## *Responsabilidade e ética*

---

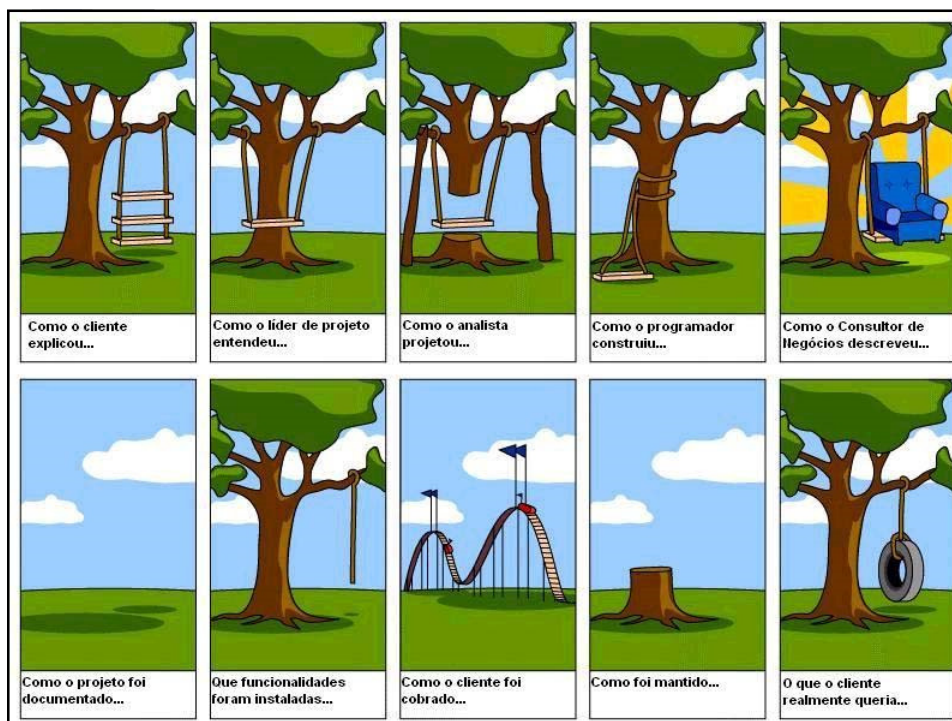
- Dilemas/Pontos de vista divergentes
- Discordar políticas da alta gerência?
- Relatar problemas com o software?
- Liberação de sw sem testes?
- SW militares/nucleares?

## *Engenharia de Software*

---

- FOCO ... OBJETIVO FINAL

**QUALIDADE**



## Qualidades Desejáveis

### Qualidades Desejáveis

- **Questão do desenvolvimento de SW**
- **Desenvolvimento x Fabricação**
- **Produto de Fácil Mudança ? Maleabilidade ?**
- **Sim → Código**
- **Não → Requisitos, análise, projeto, código, testes, etc ...**

## *Qualidades Desejáveis*

---

### ***Qualidades Desejáveis***

- Sim → Código
- Não → Requisitos, análise, projeto, código, testes, etc ...
  
- Sim → “manutenção”
- Não → Novos requisitos
  
- SW – desenvolvimento HUMANO
- SW – desenvolvimento não-fábrica ...

## *Qualidades Desejáveis*

---

### ***Qualidades Desejáveis***

- Classificação
  
- Internas – (desenvolvedor)
- Externas – (usuário)
  
- Produto
- Processo

## *Qualidades Desejáveis*

---

### ***Corretude/Correção***

- Estiver de acordo com a especificação
- Funcionalmente corretos
  
- Questão – Especificação? Sim? não? Em parte?
  
- Algoritmos de alto nível
- Bibliotecas padronizadas
- Testes

## *Qualidades Desejáveis*

---

### ***Confiabilidade***

- Confiável = usuário puder depender dele
- Relativo – depende do usuário
- Produtos não confiáveis não sobrevivem ...
  
- Questões?
- Confiar <> desgaste?
- Confiar ? Por quanto tempo?
- Foco nos Testes !

## *Qualidades Desejáveis*

---

### ***Robustez***

- **Robusto = comportamento razoável**
- **Capacidade de recuperar-se de erros/problemas não previstos – falha HD, queda luz, etc ...**
- **Depende**
  - Tipo do SW
  - Área de aplicação

## *Qualidades Desejáveis*

---

### ***Performance/Desempenho***

- **Forma utilização recursos computacionais**
- **Afeta usabilidade**
- **Questão:**
  - Otimizar SW – ignorar recursos (ex. Windows Vista)
  - Economizar no SW – economizar recursos
- **Depende do Propósito da Aplicação**

## *Qualidades Desejáveis*

---

### ***Amigabilidade***

- **Facilidade de utilização (por Usuários!!)**
- **Relativa**
  - Depende do nível de conhecimento
  - Depende do PERFIL do usuário
- **Interface Homem-Computador**
- **Foco → ADAPTAÇÃO.**

## *Qualidades Desejáveis*

---

### ***Manutenabilidade***

- **Alterações após liberação**
- **Envolve mais de 60% do custo TOTAL projeto SW**
- **Corretiva**
- **Perfectiva**
- **Adaptativa**

## *Qualidades Desejáveis*

---

### **Manutenabilidade**

- **Divide-se em:**
- **Reparabilidade**
  - Permitir correção defeitos com quantidade limitada de trabalho
  - EX – Carros – índice de reparabilidade
  - Utilização de componentes PADRÕES

## *Qualidades Desejáveis*

---

### **Manutenabilidade**

- **Divide-se em:**
- **Evolutanabilidade**
  - Capacidade de ser evoluído ...
  - **Depende do estágio do SW**
    - INICIAL – mais fácil
    - DEPOIS TEMPO – mais difícil

## *Qualidades Desejáveis*

---

### ***Reusabilidade***

- **REAPROVEITAMENTO de SW**
  - Código
  - Projeto
  - Tudo ...
- **Difícil de obter a posteriori**
- **Foco – Componentização extrema !**

## *Qualidades Desejáveis*

---

### ***Portabilidade***

- **Execução em diferentes Plataformas (HW/SW)**
- **Questão:**
  - **Específico – mais adequado/rápido/confiável**
  - **Portável - ??**
- **Questão de codificação**

## *Qualidades Desejáveis*

---

### ***Entendabilidade***

- Capacidade do SW ser “entendido”
- Depende da “complexidade”
- Foco em Metodologia
- Foco em Padrões de desenvolvimento
- Foco em Componentização
- Questão do GURU. ?

## *Qualidades Desejáveis*

---

### ***Interoperabilidade***

- Coexistir/cooperar com outros SW
- Microsoft Office
- Conceito de Sistemas ABERTOS

## *Qualidades Desejáveis*

---

### ***Produtividade***

- **Qualidade do PROCESSO de desenvolvimento do SW**
- **Envolve QUALIDADE DE SOFTWARE**
- **Envolve GERÊNCIA DE PROJETOS**
- **CMMi**
- **MPS.BR**

## *Qualidades Desejáveis*

---

### ***Timeliness***

- **Entrega no PRAZO !**
- **Questão:**
- **Mais “engenharia de sw”**
  - **Mais tempo**
  - **Mais custo**
- **Pelo menos nos primeiros momentos**

## Qualidades Desejáveis

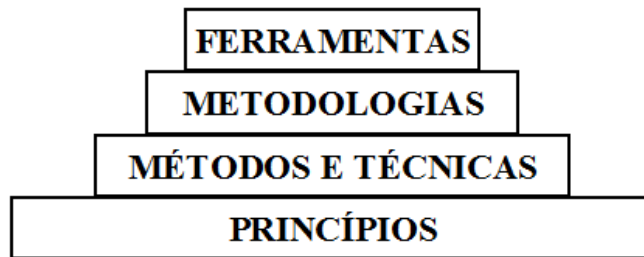
Qualidades Representativas	Produto	Processo	Interna	Externa
Corretude	X			X
Confiabilidade	X	X	X	X
Robustez	X	X	X	X
Performance ou Desempenho	X	X	X <sup>3</sup>	X
Amigabilidade	X			X
Verificabilidade	X		X	
Manutenibilidade	X	X	X	
Reusabilidade	X	X	X	
Portabilidade	X		X	
Entendabilidade	X		X	X
Interoperabilidade	X		X	
Produtividade		X	X	
<i>Timeliness</i>		X	X	
Visibilidade	X	X	X	X <sup>4</sup>

Classificação das qualidades do *software*

## Princípios

- **Objetivo = SUCESSO no desenvolvimento**
- **Envolve tanto PRODUTO quanto PROCESSO**
- **GENÉRICOS**
- **INDEPENDENTES de linguagem/BD/SO e etc...**
  
- **declarações gerais e abstratas que descrevem as propriedades desejadas dos processos de desenvolvimento e dos produtos de *software* ...**

## *Princípios*



## *Princípios*



## *Princípios*

---

### **Rigor e Formalismo**

- **Desenvolvimento SW artesanal**
  - CRIATIVO
  - INSPIRAÇÃO
- **RIGOR**
- **Complemento à criatividade ...**

## *Princípios*

---

### **Rigor e Formalismo**

- **Desenvolvimento SW artesanal**
  - CRIATIVO
  - INSPIRAÇÃO
- **RIGOR → FORMALISMO**
- **Complemento à criatividade ...**

## *Princípios*

---

### **Rigor e Formalismo**

- **Abordagem rigorosa e sistemática**
- **Programação = atividade rigorosa ...**
- **Depende:**
  - *Caso*
  - *Aplicação*
- *“Mais rigoroso = mais lento = mais caro”*

## *Princípios*

---

### **Separação de Preocupações**

- **tratar individualmente de diferentes aspectos de um problema de forma a concentrar esforços separadamente.**
- *Senso COMUM...*

## *Princípios*

---

### **Separação de Preocupações** **Preocupações**

- *Regras de negócio*
  - *Interface*
  - *Robustez*
  - *Desempenho*
- 
- *Diferentes VISÕES do SW*

## *Princípios*

---

### **Separação de Preocupações**

**A única forma de dominar a complexidade do projeto é separar as preocupações e decisões do projeto. Primeiramente alguém deve tentar isolar problemas que estão menos relacionados com outros.**

## *Princípios*

---

### **Modularização**

- **Dividir complexidade ...**
- **Modular x Monolítico**

## *Princípios*

---

### **Modularização**

- **O principal benefício da modularização é que ela permite que o princípio da separação de preocupações seja aplicado em duas fases. Primeiramente quando tratarmos dos detalhes de cada módulo isoladamente (ignorando detalhes dos outros módulos) e, posteriormente, quando tratarmos das características globais de todos os módulos incluindo seus relacionamentos, o que possibilita interligá-los para formar um sistema íntegro e coeso.**

## *Princípios*

---

### **Modularização**

- **Bastante COESÃO**
- **Pouca INTERRELAÇÃO**

## *Princípios*

---

### **Abstração**

- **A abstração é um processo pelo qual identificamos os aspectos importantes de um fenômeno ignorando seus detalhes.**
- **Depende do propósito da abstração.**

## *Princípios*

---

### **Abstração**

- **A abstração acompanha todo e qualquer processo de implementação ou programação. As linguagens de programação que utilizamos nada mais são do que construções abstratas para representar ou interagir com o *hardware***

## *Princípios*

---

### **Antecipação de Mudanças**

- *A habilidade do software em poder evoluir ...*
- *Prevista ANTES de desenvolver ...*
  
- *Questão de novos HW ..*
- *Questão da área FINANCEIRA ...*

## *Princípios*

---

### **Generalização**

- *Toda vez que você for solicitado para resolver um determinado problema tente, primeiramente, se focar na descoberta de um problema mais geral que possa existir por trás do problema em questão ...*

## *Princípios*

---

### **Generalização**

- *Soluções generalizadas, por outro lado, podem ser mais custosas em termos de velocidade de execução, requisitos de memória e/ou tempo de desenvolvimento do que soluções que são “feitas sob medida” para o problema original.*

## *Princípios*

---

### **Incrementabilidade**

- *A incrementabilidade é o princípio que busca a perfeição ou a obtenção dos objetivos através de passos que evoluem (ou são incrementados) ao longo do tempo*