

TÓPICO 01 - INTRODUÇÃO AO CÓDIGO LIMPO

Clean Code - Professor Ramon Venson - SATC 2025.2

A crise do software

Em 1968, a OTAN (Organização do Tratado do Atlântico Norte) criou um projeto para discutir e implementar padrões de desenvolvimento e especificar, pela primeira vez, o termo Engenharia de Software

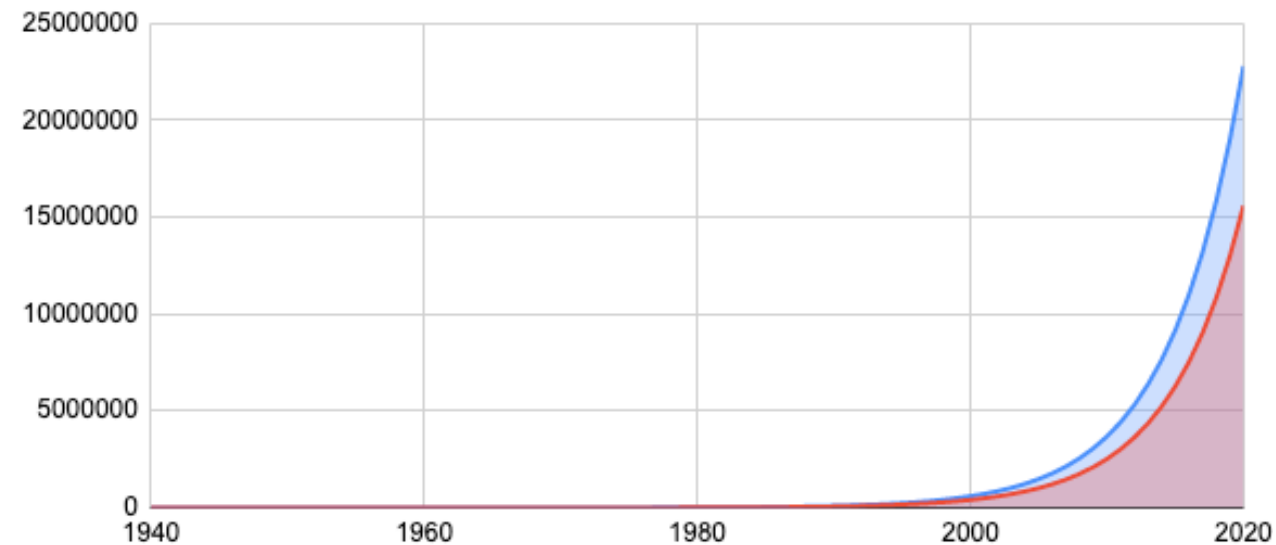
Aumento da demanda e da complexidade

Demanda crescente de novos softwares, impulsionada pelo estabelecimento de novas redes de computadores

Total programmers and Software Developers



■ Total programmers ■ Software Developers



Mesmos métodos e força de trabalho

Necessidade de criar novos métodos e ferramentas de desenvolvimento que suportassem a demanda

Caso Therac-25

Na década de 80, a Atomic Energy of Canada Limited (AECL) desenvolveu uma série de máquinas de radioterapia chamadas Therac



6 mortes entre 1985 e 1987



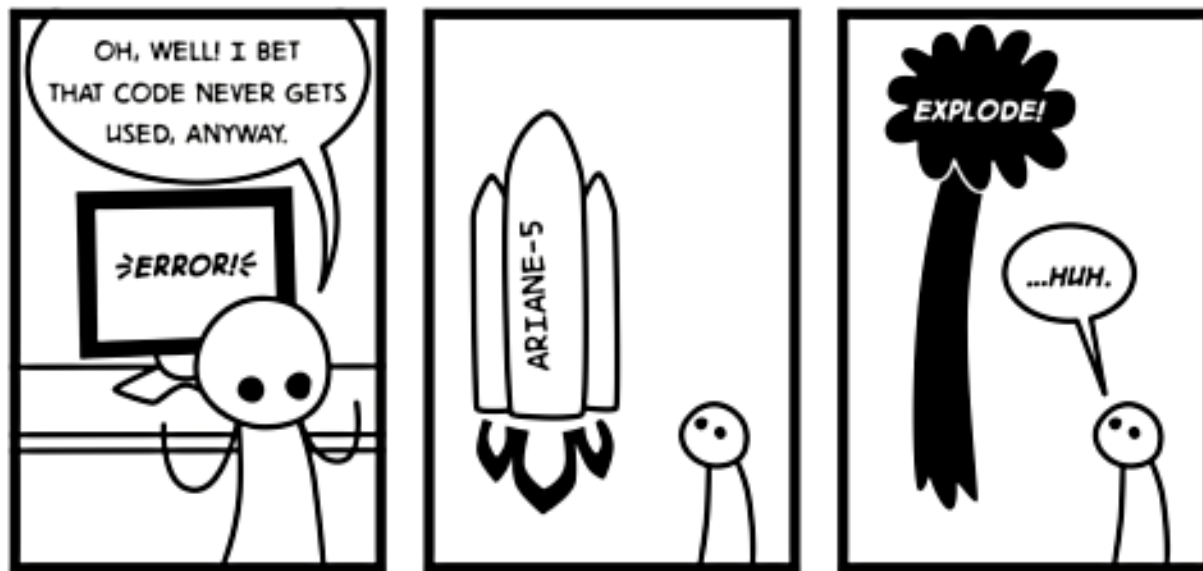
Falhas de Software

- Design ruim
- Sem testes
- Sem mensagens de erro
- Excesso de confiança

Caso Ariane 5

Em 1996, o foguete **Ariane 5** explodiu após 39 segundos de voo e causou um **prejuízo de 370 milhões de dólares**.





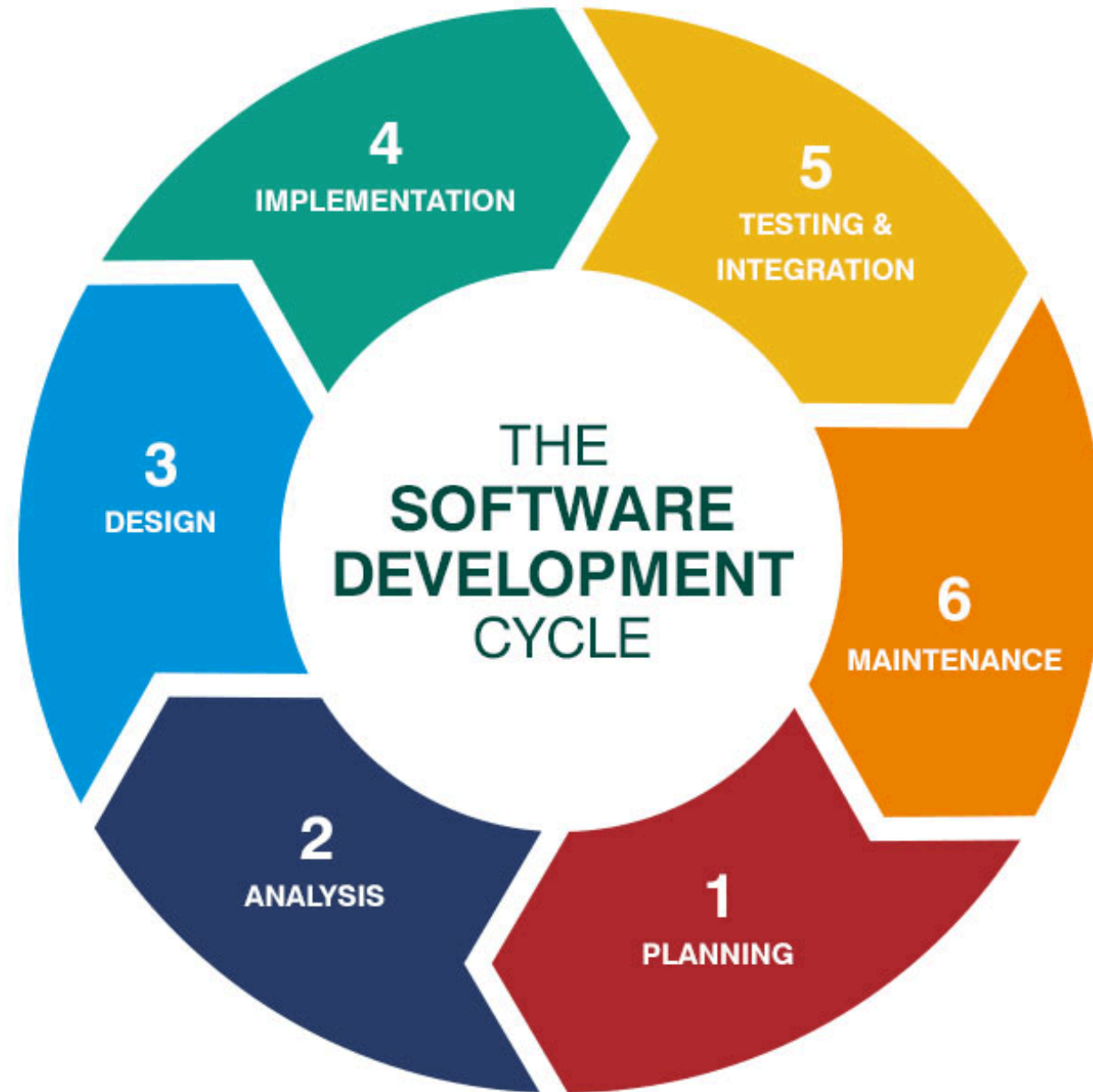
Causa: Overflow

O sistema do foguete tentava converter uma variável de 64 bits em uma de 16 bits, resultando em um overflow do resultado em tempo de execução.

Que outras falhas poderiam ser evitadas?

Como evitar falhas?

- Desenvolvimento Ágil
- Testes Automatizados
- Revisão de Código
- Arquitetura Limpa
- Programação Defensiva

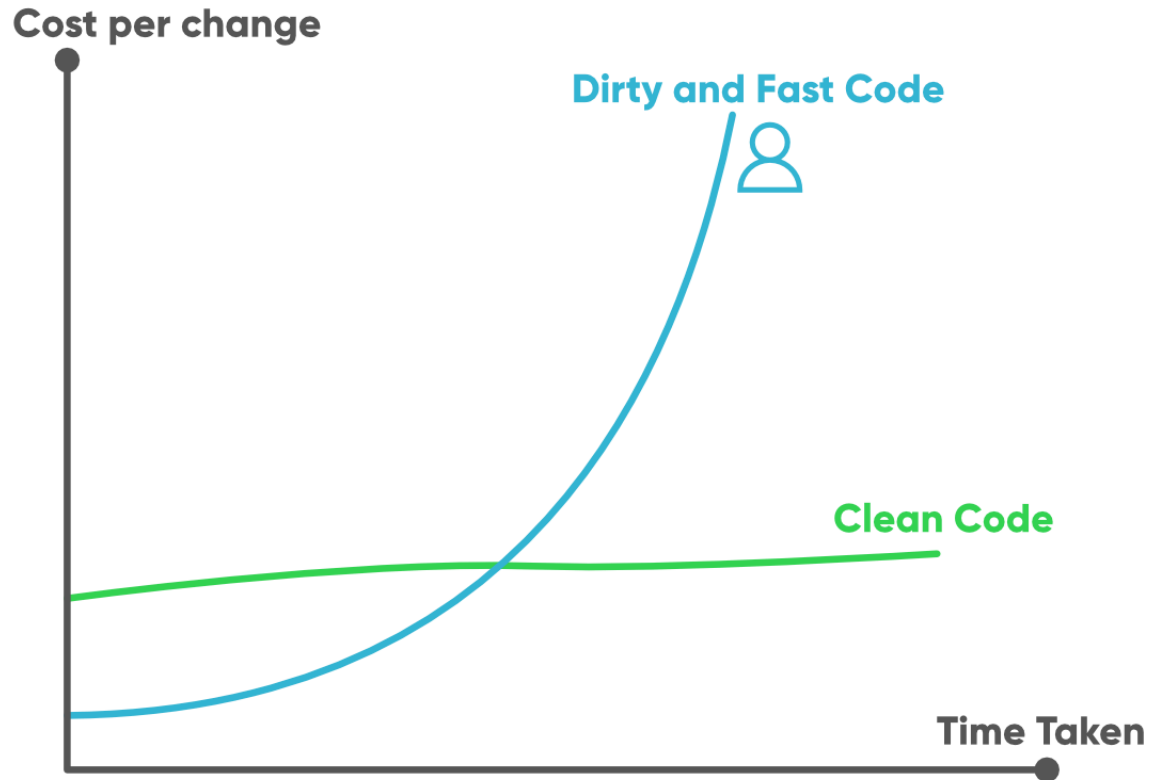


Impactos do Código Limpo

- Redução de custos
- Maior produtividade
- Menos vulnerabilidades
- Mais facilidade na manutenção
- Escalabilidade de equipes

Código Limpo

- Legível (Readability)
- Mantível (Maintainability)
- Testável (Testability)
- Extensível (Extensibility)
- Elegante (Beauty)



Barreiras do Código Limpo

- Ignorância
- Teimosia
- Arrogância
- Falta de Tempo

É sua responsabilidade como programador defender seu código

Robert C. Martin

Débito Técnico

O Débito Técnico é o resultado de uma implementação sem planejamento, mas que é adicionado para atender a uma demanda imediata.

Com o tempo, o débito técnico se acumula e pode levar a problemas de performance, segurança e escalabilidade.



Analogia do Débito Bancário

O Débito Técnico é como um empréstimo bancário. Você pode pegar um empréstimo para comprar algo que precisa agora, mas terá que pagar juros no futuro.

O débito técnico as vezes compensa?



Clean Code

A Handbook of Agile Software Craftsmanship



Robert C. Martin



Foreword by James O. Coplien

Práticas de Código Limpo

Livro escrito por Robert C. Martin
cunhou o termo **Código Limpo**.

É uma referência e não um dogma.

Nomenclaturas

Existem apenas duas coisas difíceis em ciências da computação: Invalidação de cache e dar nome as coisas - Phil Karlton, Netscape developer

Name	Label
e3	Have you bought a...
m_e3_0	No vehicles
m_e3_1	Bicycle
m_e3_2	Motorbike
m_e3_3	Tuk-tuk?
m_e3_4	Other vehicle?

```
9 create or replace table COMPANY_NAME.somedata AS
8 -- ...
7 create or replace table COMPANY_NAME.moredata AS
6 -- ...
5 create or replace table COMPANY_NAME.moremoredata AS
4 -- ...
3 create or replace table COMPANY_NAME.datadata AS
2 -- ...
1 create or replace table COMPANY_NAME.somemoredata AS
10 -- ...
```

```
private int  
{  
    try  
    {  
        bool ehLoja = new  
  
        int lojaId = 0;  
  
        if (!ehLoja)  
        {
```



Descodando | Programação e Tec... · 21m ...

Não sei vocês, mas eu tenho costume de dar nome de pessoas para as variáveis

```
1  ✓ def soma(carlos, maria):  
2      ana = carlos + maria  
3      return ana  
4  
5  joaquim = soma(5, 7)  
6  
7  print("A soma é:", joaquim)
```


Funções e Comentários

Uma função bem escrita vale mais do que mil comentários. Um comentário pode mal escrito pode valer por uma função mal escrita.

0 references

```
public bool IsEven(int foo) {  
    if (foo < 0) {  
        foo *= -1;  
    }  
    for(int i = 0; i <=foo; i += 2)  
    {  
        if (i == foo) {  
            return true;  
        }  
    }  
    for (int i = 1; i <= foo; i += 2)  
    {  
        if (i == foo)  
        {  
            return false;  
        }  
    }  
    return (foo % 2 == 0);  
}
```

```
def date_warning(): #warn students that there book is not yet returned
    #for a day or two or more
    borrow_records = []
    borrow_records.append(get_borrow_data()) #Appending the loaded json to to be incremented
    for x in borrow_records: #First increment
        for b in x: #Second increment Note: Should have use the json dumps or json loads
            current_datetime = datetime.now() #Get the current time and date
            ret_date = b['date_returned'] #return date
            ret_time = b['time_returned'] #return time

            return_stat = b['return_status'] #return status boolean true or false
            #return_stat is only true if a book is returned and false if not

            date_time_ret = f'{ret_date} {ret_time}' #Combine both into a string

            #turn date_time_ret into a strptime formats
            initial_ret = datetime.strptime(date_time_ret, "%Y/%m/%d %I:%M:%p")
            current_datetime = datetime.now() #Get current time and date

            #Calculate the total amount of hours to be calculated and turned into fines
            current_data = (current_datetime - initial_ret).total_seconds() / 3600
            if current_data != 0 and return_stat == False: #Sending a message if the return_stat = false
                #And the current_data !=0 means that if its 0 hence it still has time left to be returned
                print("Please return the book")
```



```
19 group by 1;
```

Expected indent of 4 spaces. (LT02)

[View Problem \(⌘F8\)](#) No quick fixes available

```
23 select generate_series(  
24     date_trunc('hour', now()) - '1 day'::interval,  
25     date_trunc('hour', now()),  
26     '1 hour'::interval  
27 ) as hour  
28 )
```

```
29  
30 select
```

If you are looking for
an image, it was
probably deleted.

Tratamento de Erros

O primeiro bug de software registrado foi, literalmente, um inseto preso em um computador em 1947, documentado por Grace Hopper.



```
public static void main(String[] args) {  
    try {  
        InputStream file = new FileInputStream(new File(args[0]));  
    }  
    catch (FileNotFoundException e){  
        System.out.println("Could not locate file, please try again");  
    }  
}
```

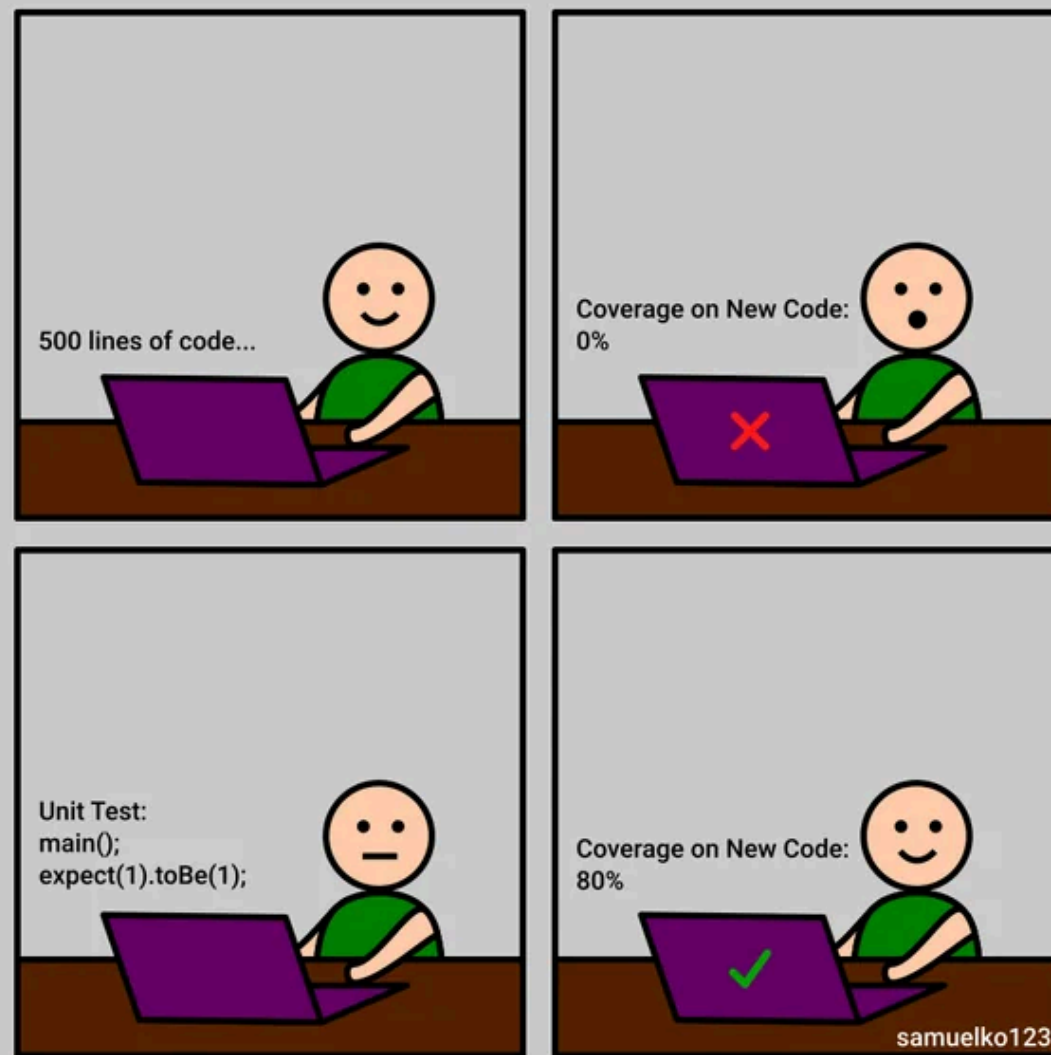
```
public static void main(String[] args) throws Exception {  
    InputStream file = new FileInputStream(new File(args[0]));  
}
```


Testes Unitários

Testes unitários são testes que testam uma unidade de código isoladamente.

Esses testes são escritos antes do código, e são executados automaticamente a cada atualização.





Escrever código limpo é uma arte,
não uma ciência.

Material de Apoio

- [Clean Code \(PT/EN\)](#)
- [As falhas numéricas que podem causar desastres](#)