

# Prof. esp. Thalles Canela

- **Graduado:** Sistemas de Informação - Wyden Facimp
- **Pós-graduado:** Segurança em redes de computadores - Wyden Facimp
- **Professor (contratado):**
- **Pós-graduação:** Segurança em redes de computadores - Wyden Facimp
- **Professor (Efetivado):**
- **Graduação:** Todo núcleo de T.I. - Wyden Facimp
- **Tech Lead na Motoca Systems**

## Redes sociais:

- **Linkedin:** <https://www.linkedin.com/in/thalles-canela/>
- **YouTube:** <https://www.youtube.com/aXR6CyberSecurity>
- **Facebook:** <https://www.facebook.com/axr6PenTest>
- **Instagram:** [https://www.instagram.com/thalles\\_canela](https://www.instagram.com/thalles_canela)
- **Github:** <https://github.com/ThallesCanela>
- **Github:** <https://github.com/aXR6>
- **Twitter:** <https://twitter.com/Axr6S>

# APLICAÇÕES, CRITÉRIOS E CATEGORIAS DE LINGUAGENS

# Compreendendo o problema

- Missão: desenvolver aplicativo simples para Android
  - Três telas, trocadas com deslizamento lateral



- Não tem elementos 3D, não faz cálculos
- Não tem cadastro ou interações complexas

# Compreendendo o problema

- Missão: desenvolver aplicativo simples para Android
  - Três telas, trocadas com deslizamento lateral
  - Não tem elementos 3D, não faz cálculos
  - Não tem cadastro ou interações complexas



# Compreendendo o problema

- Missão: desenvolver aplicativo simples para Android

**O desenvolvedor escolheu C# usando o  
Unity 3D para levar a tarefa a cabo**



**Será que foi uma  
boa escolha?**

# Compreendendo o problema

- Missão: desenvolver aplicativo simples para Android

**No caso, a aplicação ficou gigante (70MB  
ao invés do ideal, 2MB)**



# Compreendendo o problema

- Missão: desenvolver aplicativo simples para Android

**Escolha infeliz de ferramentas e linguagem**

**Mas quantos não cometem esse mesmo erro?**

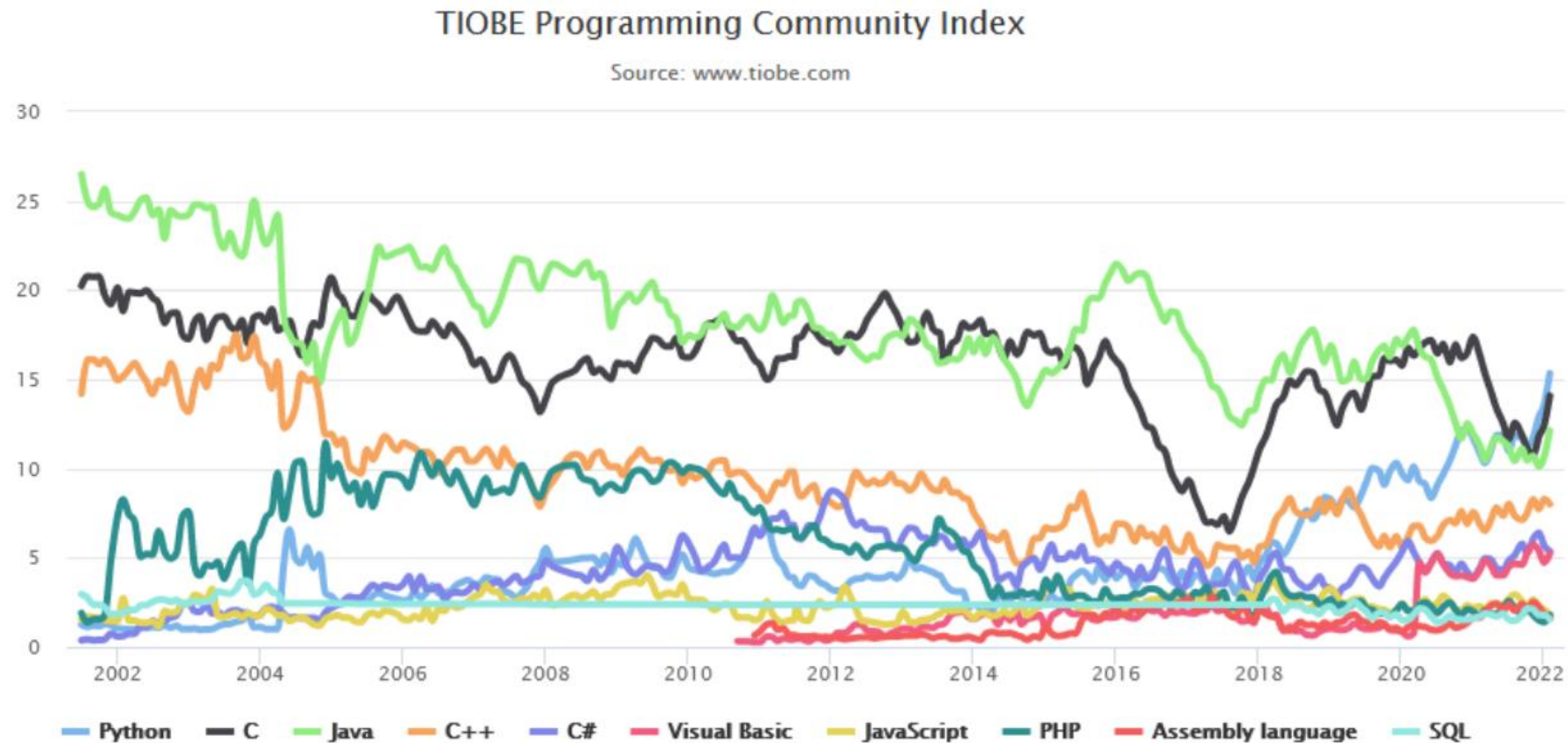
Linguagem	Posição	Rating
Python	1	15,33%
C	2	14,08%
Java	3	12,13%
C++	4	8,01%
C#	5	5,37%
Visual Basic	6	5,23%
JavaScript	7	1,83%

Linguagem	Posição	Rating
PHP	8	1,79%
Assembly	9	1,60%
SQL	10	1,55%
Go	11	1,23%
Swift	12	1,18%
R	13	1,11%
MATLAB	14	1,03%

<https://www.tiobe.com/tiobe-index/> , Fevereiro de 2022



# Compreendendo o problema



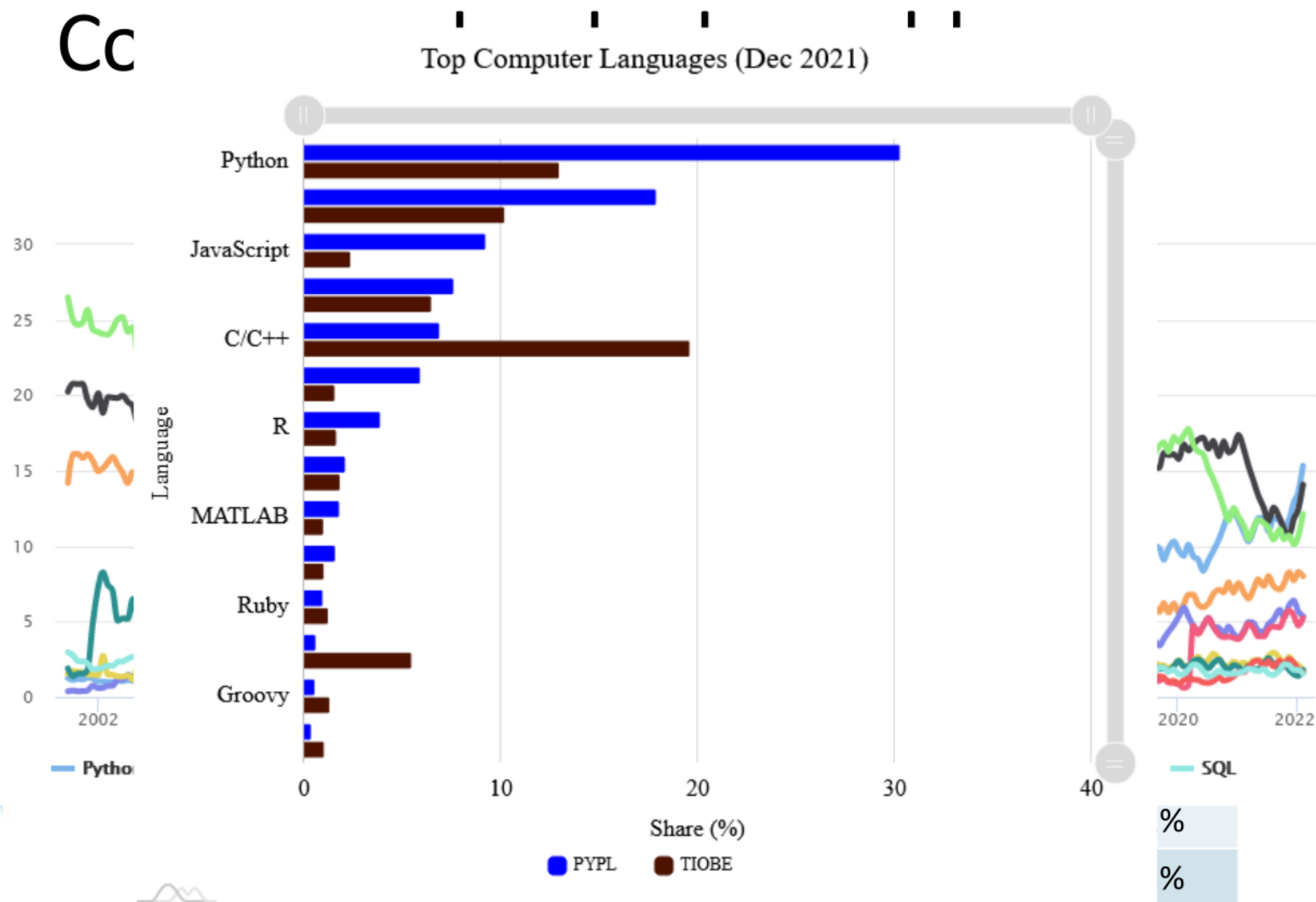
Visual Basic	6	5,23%
JavaScript	7	1,83%

R	13	1,11%
MATLAB	14	1,03%

<https://www.tiobe.com/tiobe-index/> , Fevereiro de 2022



# Cc



<https://www.tiobe.com/tiobe-index/> , Fevereiro de 2022

# Objetivos

- Conhecer os grandes domínios de aplicações e suas características
- Compreender os fatores que influenciam a arquitetura das linguagens e os principais paradigmas
- Conhecer os critérios para escolha de linguagens



# EVOLUÇÃO DAS LINGUAGENS:

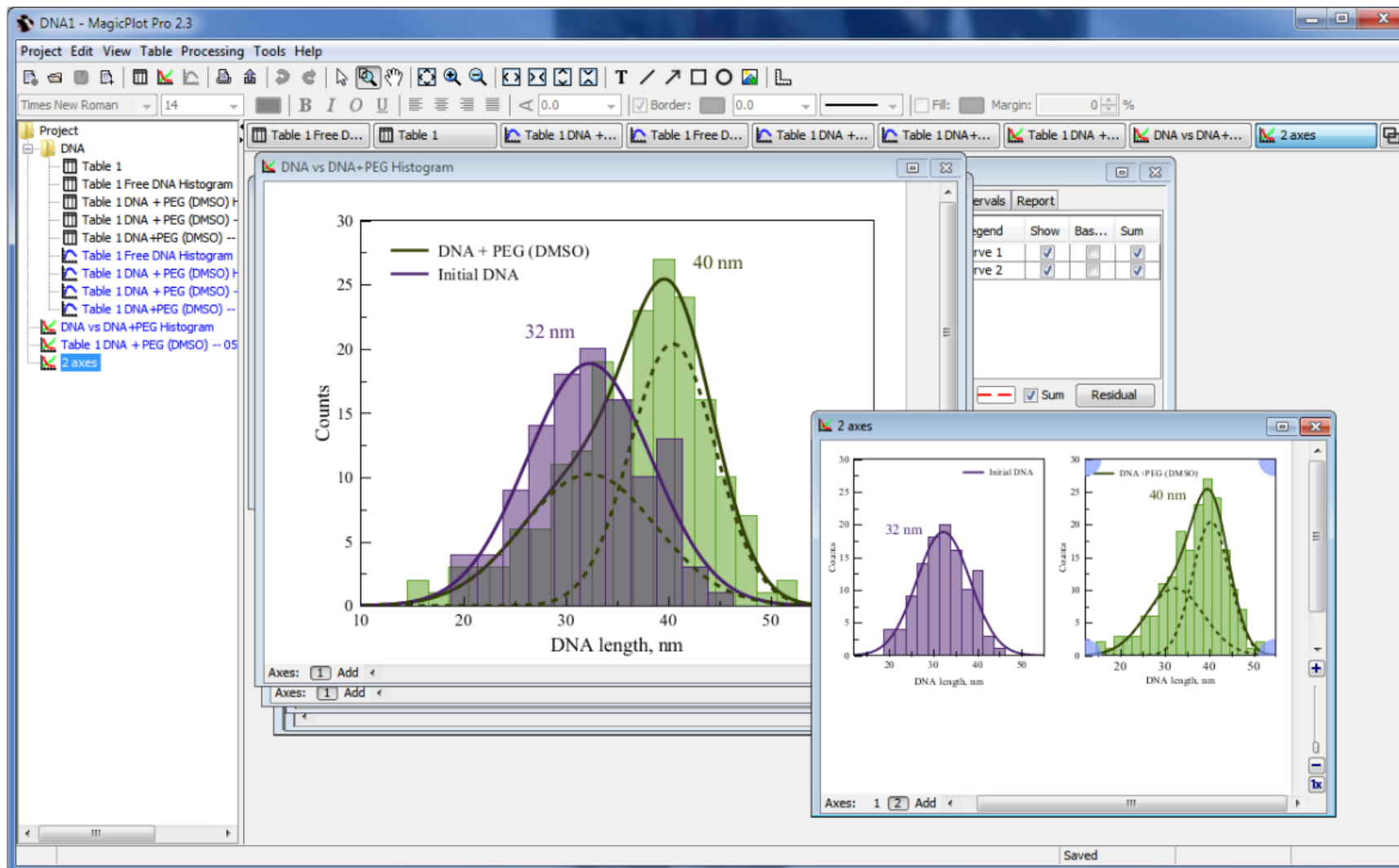
- Por Quê?

# Linha do Tempo

- Evolução das Linguagens
  - Início LM & Assembly
  - 195x/6x: FORTRAN, ALGOL 60, COBOL, Lisp
  - 197x: Pascal, Smalltalk, C, BASIC, Prolog
  - 198x: C++, Object Pascal, Objective C
  - 1991: VisualBASIC, Oak, Python
  - 1995: PHP, Ruby, Java
  - 2001: C#

Por quê?

# Observe...



# Observe...

Alterar Registros de Clientes Físicos

Nome:

Endereço:

Rua:

Cidade:

Número da Rua:

Cep:

Estado:

CPF:

RG:

Telefone Fixo:

( ) - -

Telefone Celular:

( ) - -

E-mail:

Data do Cadastro:

/ /

Observação:

Selecionar Foto

NOME	ENDEREÇO	RUA	CIDADE	NÚMERO	CEP	ESTADO	CPF	RG
VANESSA DA SILVA	AVISA SANDRO ANDRE D...	RUA BOM LOUGAR	BAIMINAS	540025	28955-022	MARANHÃO	025.450.025-...	0012001254
ERIQUE SOUZA FILHO	CONQUISTA DA VITORIA	PEDRO BADEIRA DO FRA...	NUBISFREIRE	04598785	45210-021	PARÁ	456.525.245-...	0024569850
GILNEI FILHO DULTRA CO...	SÃO JOSE DE CONSALVES	AMADARENA DO CITIO	SANTA ELIZA BEIRÃO	0055455	15498-000	PARANÁ	063.626.526-...	5025212001
VANIA DA SILVA BARUA	SAO CRISTOVAM ELENCO	BECO D VALE	PINHEIROS DIAS	545002	45978-054	PARAÍBA	225.454.202-...	0251254002
VAGNER DILTRA BOLSON...	VIRGINIA VELHA	TATARA DE BELEM	SANTRO ANDRE DE RIBE...	0450002	15400-212	RIO GRANDE...	025.165.454-...	0052525001
EMANÉL FILHO DA SILVA	SÃO JO'SE DO RIO PRETO	BALSANETI DE MIRANTES	PARACARUARI	0215250	12155-002	MATO GROS...	002.165.565-...	0224500245

☒ Busca Especifica

☐ Busca Generalizada

Todos

Alterar

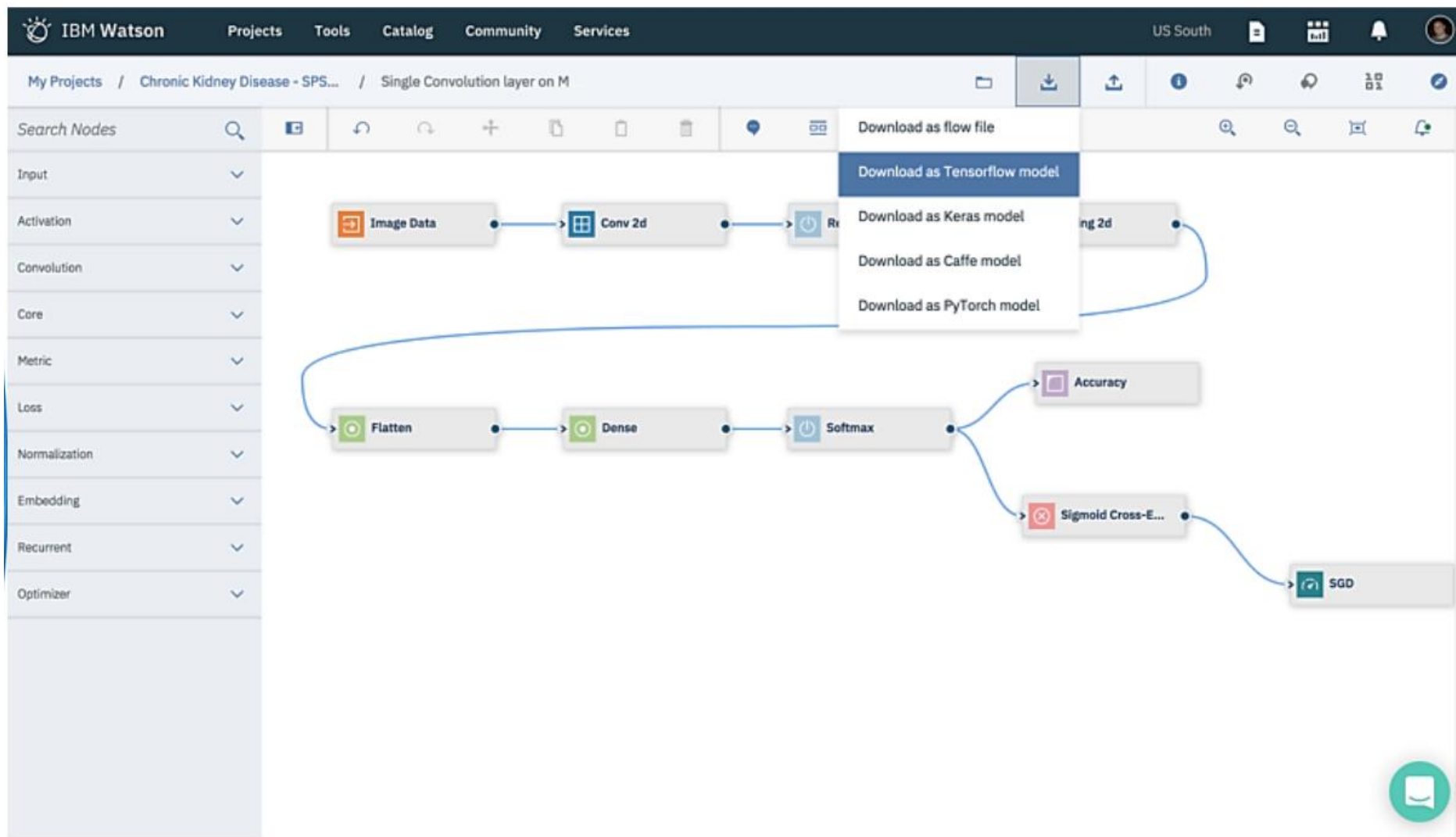
Excluir

Status do Cliente

☐ Cliente Ativo

☐ Cliente Não Ativo

# Observe...





# Observe...

The screenshot shows the WordPress dashboard for a website titled "Restaurant World Tou...". The interface includes a top navigation bar with links for "Upgrade to Pro", "New Post", a user profile for "Dave", and a search icon. Below this is a "Screen Options" and "Help" menu. A left sidebar contains a "Dashboard" link and a menu of site management tools: Home, Comments I've Made, Site Stats, Akismet Stats, My Blogs, Blogs I Follow, Store, Posts, Media, Links, Pages, Comments, Feedbacks, Appearance, Users, Tools, Settings, and a "Collapse menu" button. The main content area is titled "Dashboard" and is divided into several sections. The "Right Now" section provides a snapshot of site activity: 8 Posts, 1 Page, 5 Categories, and 52 Tags under the "CONTENT" heading; 9 Comments, 9 Approved, 0 Pending, and 0 Spam under the "DISCUSSION" heading. It also notes the current theme is "Hemingway" with 7 widgets and reports that Akismet has protected the site from 786 spam comments, with no items currently in the spam queue. The "STORAGE SPACE" section shows 3,072MB of space allowed and 0.08MB (0%) currently used. The "Recent Comments" section displays two entries: one from "Dave" on "Arctic Char #" and another from "Mandy" on the same post. On the right side, the "QuickPress" section offers a prompt to try a new quick post form, a title input field, an "Add Media" button, a content area, a "Tags" input field, and "Save Draft", "Reset", and "Publish" buttons. Below this, the "Recent Drafts" section states there are no drafts at the moment. The "Stats" section at the bottom right indicates that no stats are available for the current time period.

Restaurant World Tou... Upgrade to Pro New Post Dave Search

Screen Options Help

Dashboard

Home  
Comments I've Made  
Site Stats  
Akismet Stats  
My Blogs  
Blogs I Follow  
Store  
Posts  
Media  
Links  
Pages  
Comments  
Feedbacks  
Appearance  
Users  
Tools  
Settings  
Collapse menu

### Dashboard

#### Right Now

CONTENT	DISCUSSION
8 Posts	9 Comments
1 Page	9 Approved
5 Categories	0 Pending
52 Tags	0 Spam


Theme Hemingway with 7 Widgets


Akismet has protected your site from 786 spam comments already. There's nothing in your spam queue at the moment.

#### STORAGE SPACE

3,072MB Space Allowed	0.08MB (0%) Space Used
-----------------------	------------------------

#### Recent Comments

 From Dave on Arctic Char #  
Yes, it's a much less fishier fish than salmon. I've not heard of these two restaurants though. I'll have to ...

 From Mandy on Arctic Char #  
I agree arctic char is a great fish! It's really similar looking to

#### QuickPress

Have you tried our new home page quick post form yet? [Try it now](#) →

Enter title here

Add Media

Tags (separate with commas)

Save Draft Reset Publish

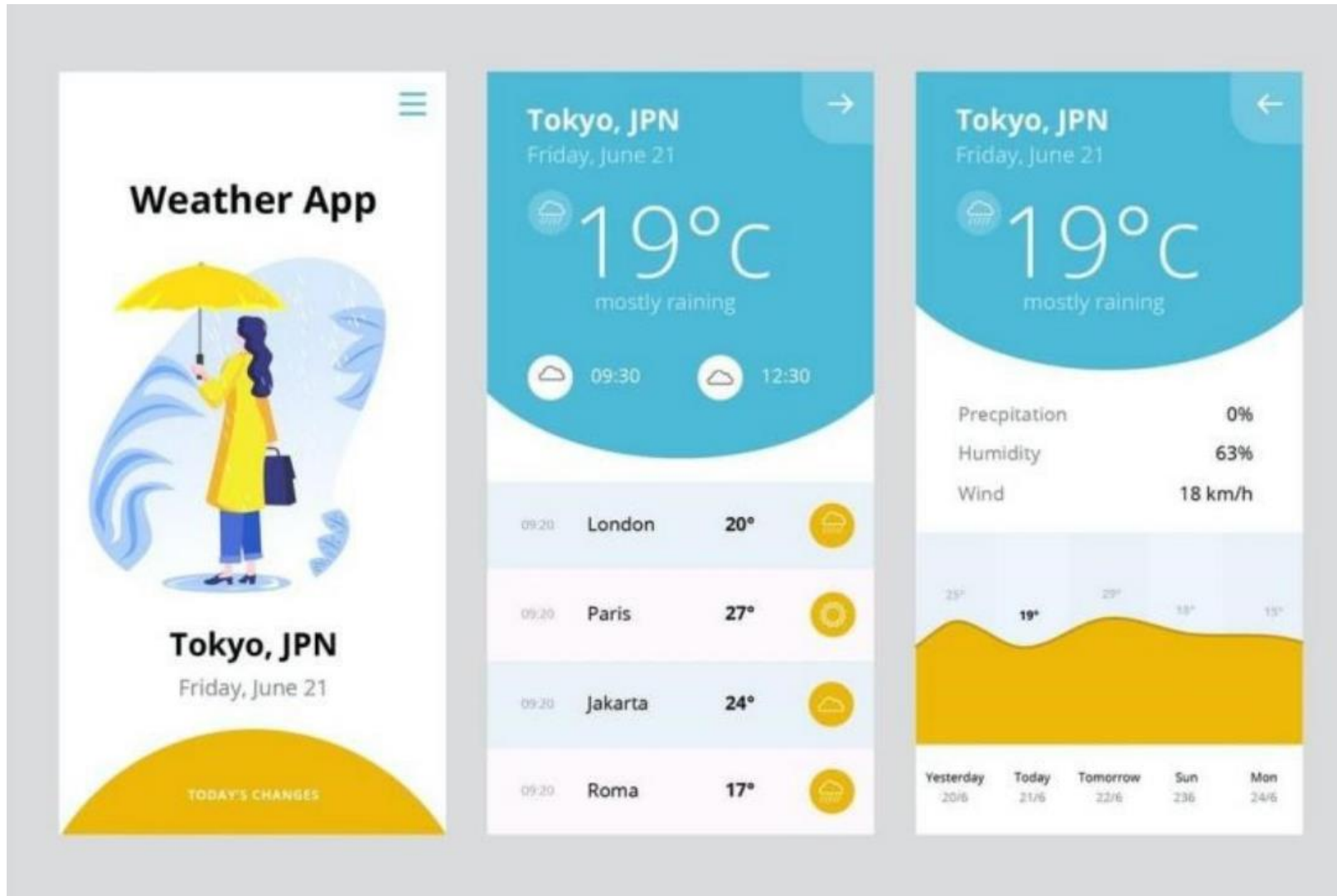
#### Recent Drafts

There are no drafts at the moment

#### Stats

No stats are available for this time period.

# Observe...

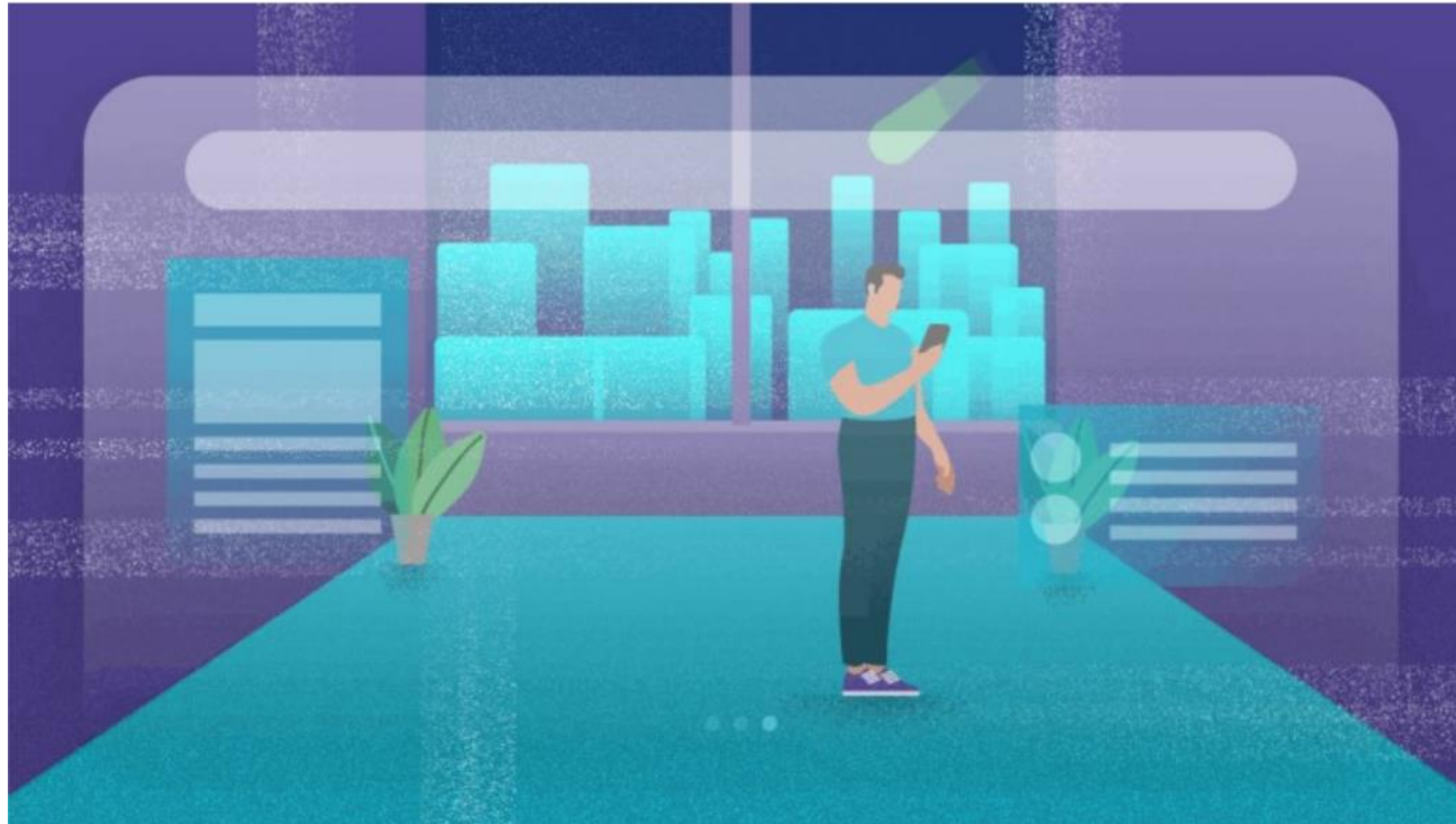


# MOTIVOS PARA DIVERSIDADE DE LINGUAGENS:

- 1. DOMÍNIOS DE PROGRAMAÇÃO

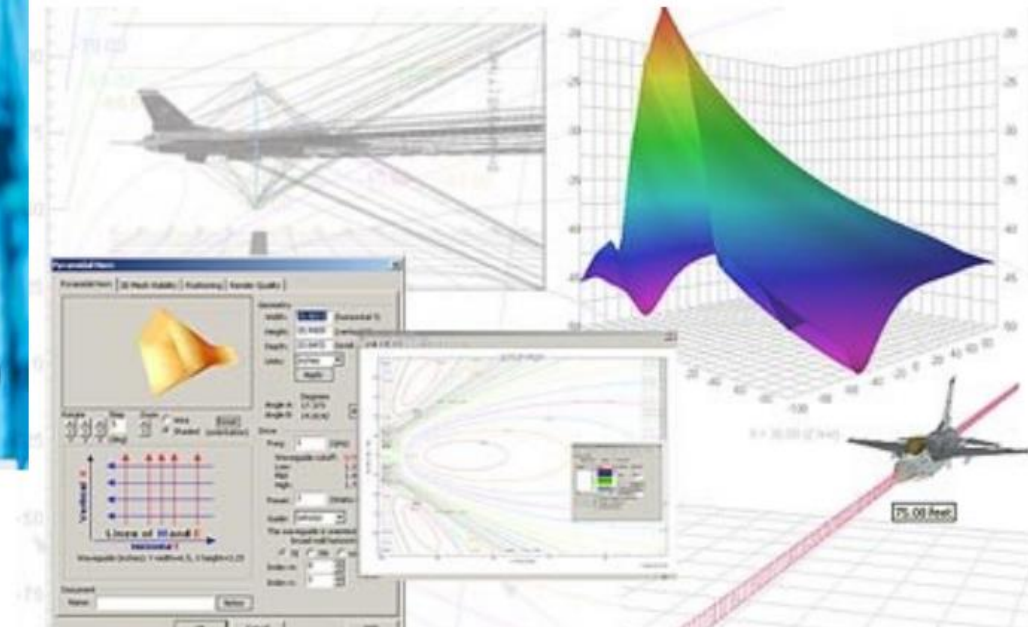
# Domínios de Programação

- Grandes Categorias de Software
  - Suas características: influenciam as linguagens



# Domínios de Programação

- Aplicações científicas
  - Primeiro tipo de aplicações (ALGOL60, FORTRAN, C)
  - Focada em cálculos e eficiência computacional



# Domínios de Programação

- Aplicações comerciais/empresariais
  - Após 1ª guerra: bancos, empresas... (COBOL, Java, C#)
  - Foco em cálculos decimais, geração de relatórios.





# Domínios de Programação

- Aplicações de IA
  - Inferências e deduções (Prolog, Lisp, C, Python)
  - Computação simbólica e associações.





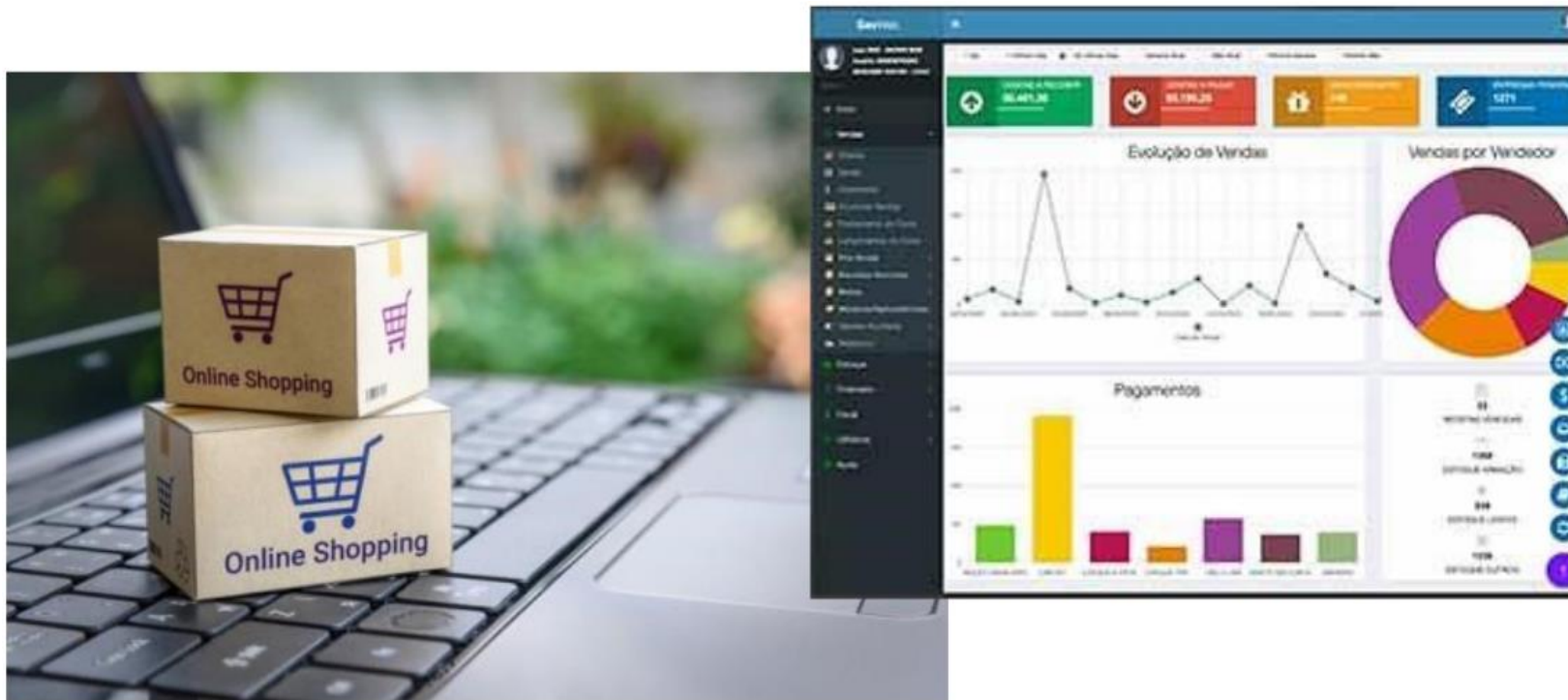
# Domínios de Programação

- Sistemas Básicos
  - Lidar diretamente como hardware (C, Assembly)
  - Foco em eficiência e baixo consumo de recursos.



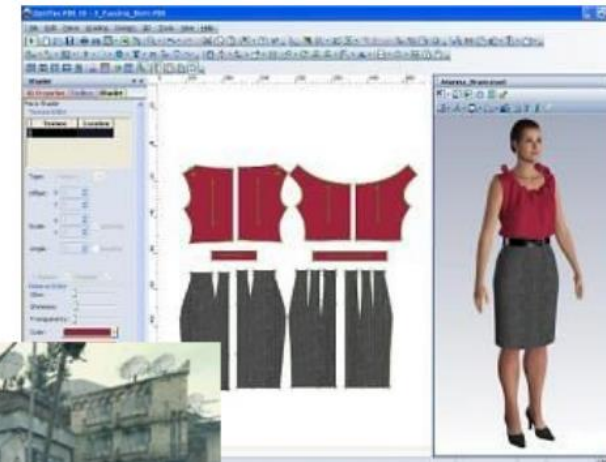
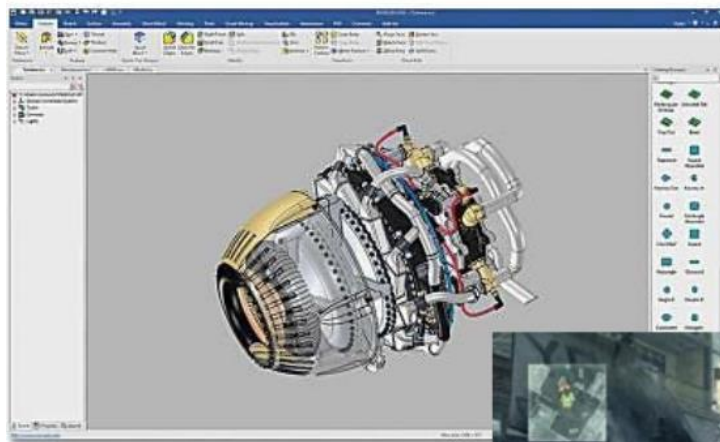
# Domínios de Programação

- Aplicações para Web
  - Acesso universal (JavaScript, Java)
  - Foco em difusão de conteúdo dinâmico.



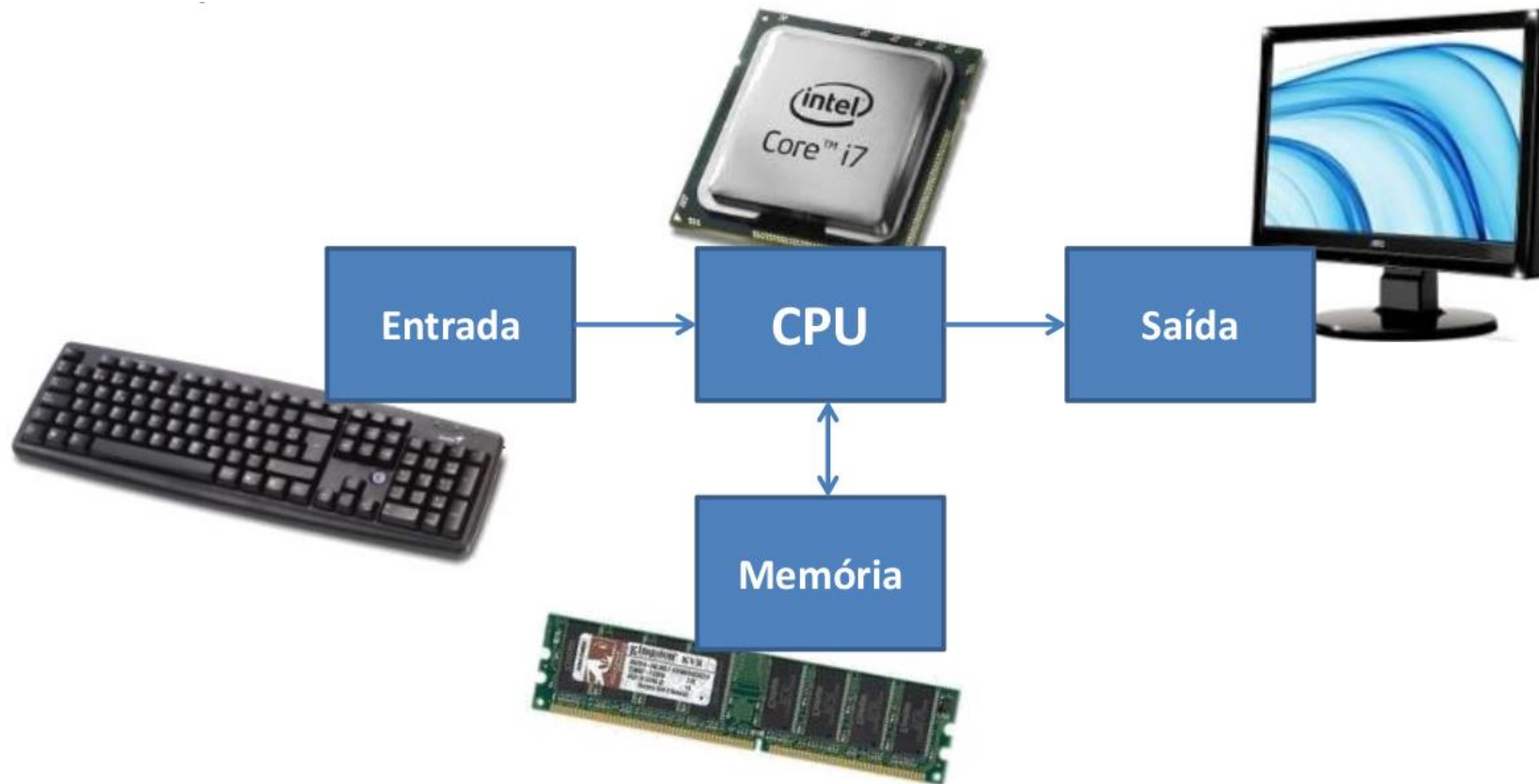
# Domínios de Programação

- Outros... Engenharia, jogos etc...
  - Combinação complexa de requisitos (C, Java, Python)
  - Linguagens de “propósito geral”



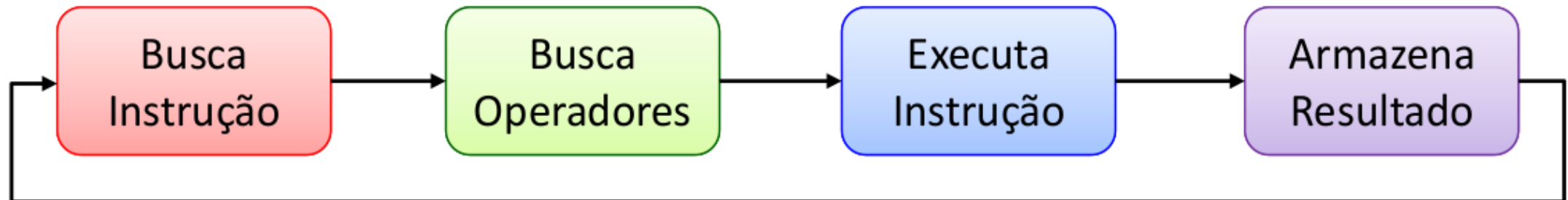
# Arquitetura de Computadores

- Lógica da máquina → lógica da linguagem
- Arquitetura de von Neumann



# Arquitetura de Computadores

- Arquitetura de von Neumann
  - Programas ficam na memória, como os dados
  - Procedimentos sequenciais para cálculo
  - Armazenamento de resultados na memória.
- Programas x Dados
  - Executar x Armazenar/Recuperar

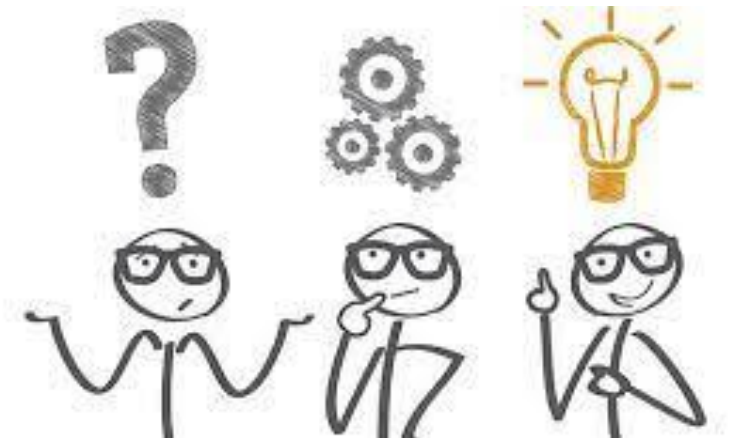


# MOTIVOS PARA DIVERSIDADE DE LINGUAGENS:

- 3. METODOLOGIAS DE PROJETO

# Software: Resolver um Problema

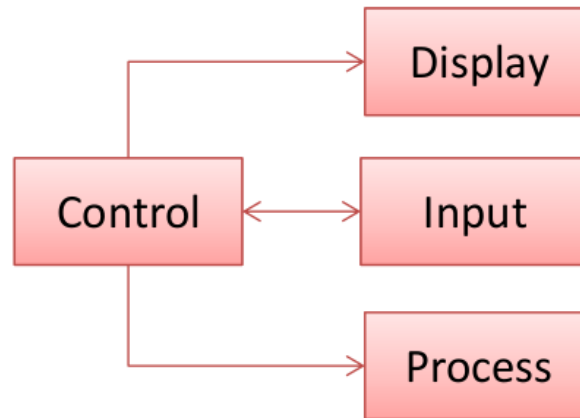
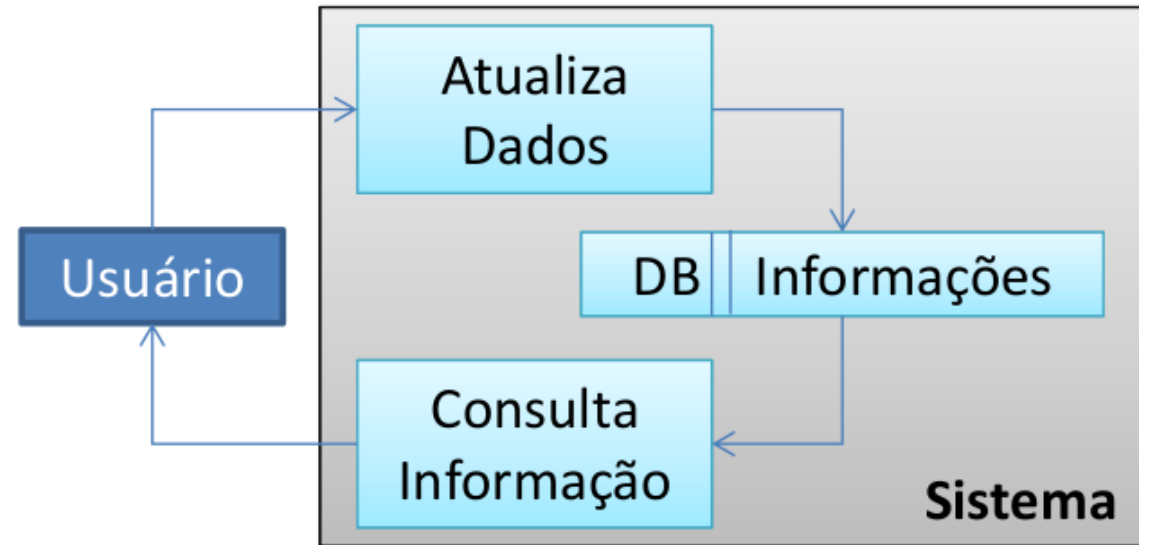
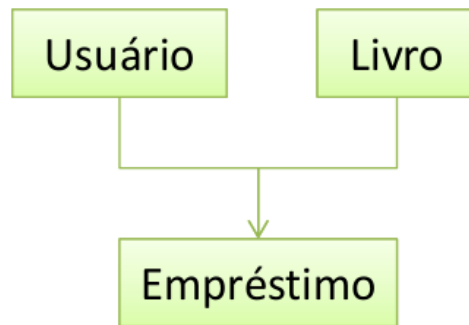
- Problemas Complexos x Custo x Tempo
  - “Sentar e programar” → Projetar
  - Análise e Projeto.
- Como implementar um sistema?
  - Compreender o domínio do problema
  - Propor modelo simplificado
  - Propor modelo detalhado
  - Implementar
  - Testar
  - Implantar.





# Análise de Sistemas

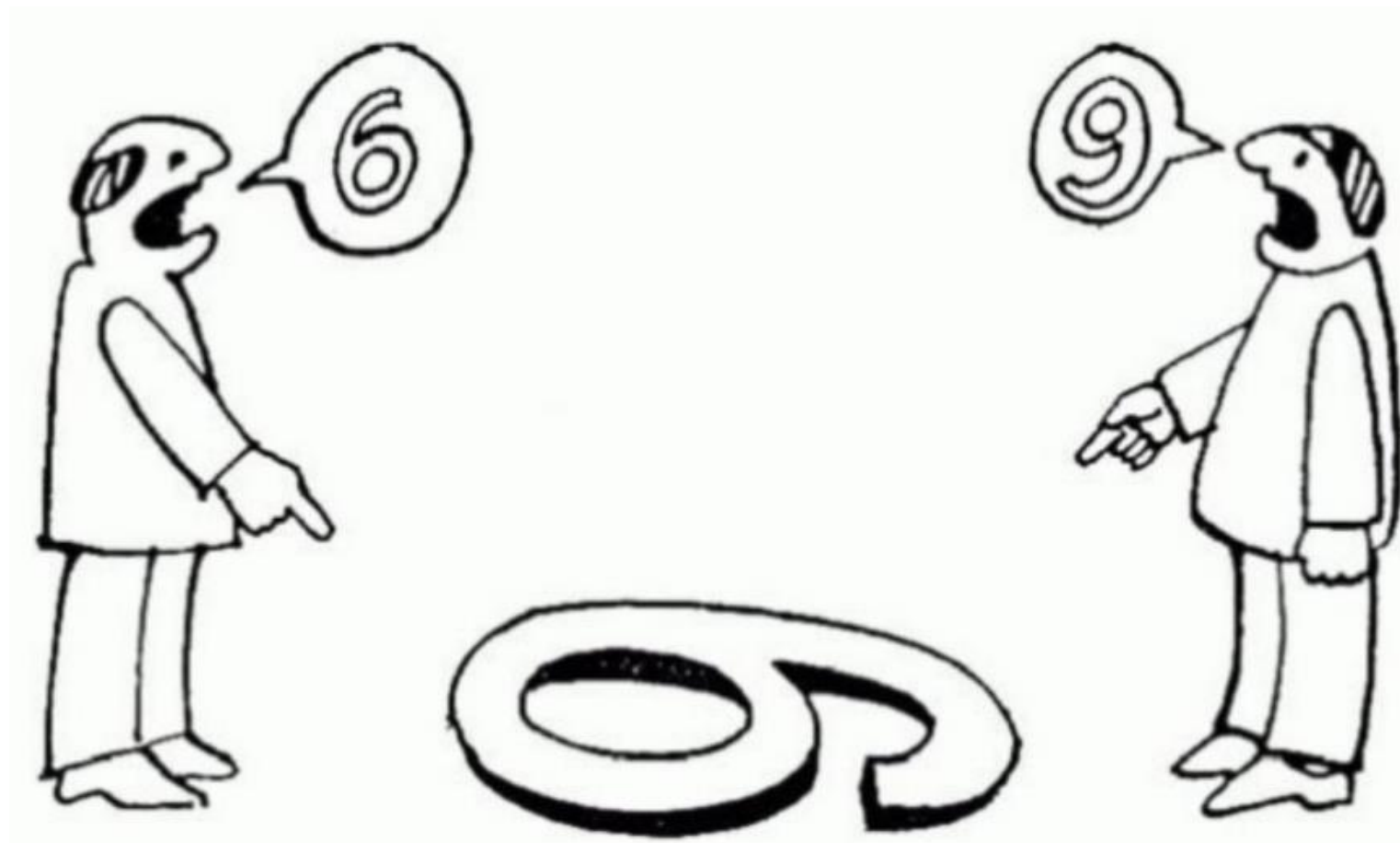
- LM & ASM: Fluxogramas
- Análise Estruturada Usuário
  - Fluxo de Dados (DFD)
  - Decomposição Funcional
- Análise Orientada a Objetos



- O que muda menos ao longo do tempo??

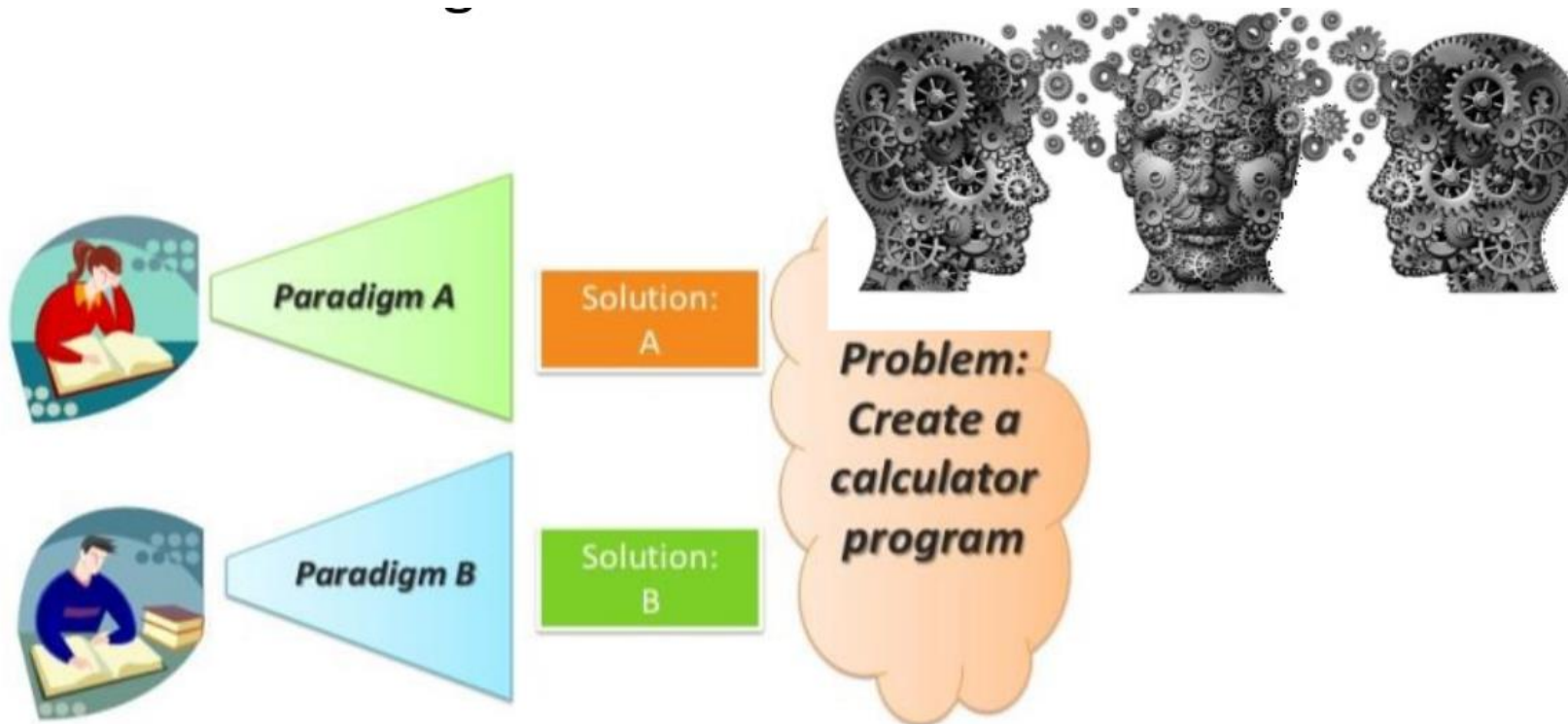
# CATEGORIAS DE LINGUAGENS

O que é um Paradigma?



# O que é um Paradigma?

- Forma de interpretar e pensar o mundo
  - Como interpretar problemas complexos....
  - Como imaginar uma solução.



# Paradigmas de Linguagem

- Linguagens Imperativas
  - Influência: arquitetura de von Neumann
  - Variáveis e procedimentos
- Linguagens imperativas podem ser
  - Estruturadas/Procedurais
    - Influência: análise estruturada
    - Ex.: COBOL, FORTRAN, C, Pascal...
  - Orientada a Objetos
    - Influência: an. orientada a objetos
    - Ex.: Smalltalk, C++, Python, Java, C#

# Paradigmas de Linguagem

- Linguagens Declarativas
  - Em oposição às imperativas
  - Abstraem a ideia de variável e/ou sequência
- Linguagens declarativas podem ser:
  - Funcionais
    - Influência: funções matemáticas
    - Ex.: Haskell, Erlang, R, XSLT
  - Lógicas
    - Influência: lógica matemática
    - Ex.: Prolog, LISP

# Paradigmas de Linguagem

- Na prática...
  - Muitas linguagens são multiparadigma
    - C: imperative, procedural
    - C++: imperative, object-oriented, generic, functional style(not functional)
    - C#: imperative, declarative, functional, generic, object-oriented(class-based), component-oriented
    - Java: concurrent, class-based, functional(Java8)
    - JavaScript: imperative, functional, object-oriented
    - Python: imperative, functional, procedural, object-oriented
    - Ruby: imperative, functional, object-oriented
    - SQL: declarative, data-driven



# Exemplo: Imperativa Estruturada



```
#include <stdio.h>
#include <assert.h>

//
// Função mdc
//
int mdc(int num1, int num2) {

    int resto;

    do {
        resto = num1 % num2;

        num1 = num2;
        num2 = resto;

    } while (resto != 0);

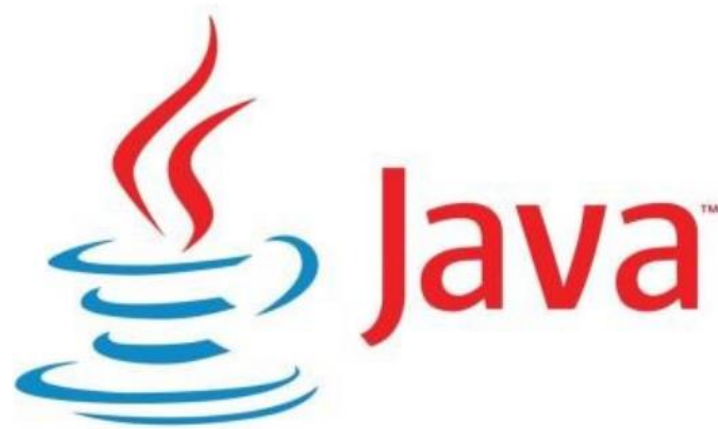
    return num1;
}

//
// Testes
//
int main() {

    assert(3 == mdc(24, 9));
    assert(10 == mdc(30, 20));

    return 0;
}
```

# Exemplo: Imperativa O.O.



```
class Lamp {
    boolean isOn;

    void turnOn() {
        // initialize variable with value true
        isOn = true;
        System.out.println("Light on? " + isOn);
    }

    void turnOff() {
        // initialize variable with value false
        isOn = false;
        System.out.println("Light on? " + isOn);
    }
}

class Main {
    public static void main(String[] args) {

        // create objects l1 and l2
        Lamp l1 = new Lamp();
        Lamp l2 = new Lamp();

        // call methods turnOn() and turnOff()
        l1.turnOn();
        l2.turnOff();
    }
}
```

# Exemplo: Imperativa 0.0.

```
class ComplexNumber:
    def __init__(self, r=0, i=0):
        self.real = r
        self.imag = i

    def get_data(self):
        print(f'{self.real}+{self.imag}j')

# Create a new ComplexNumber object
num1 = ComplexNumber(2, 3)

# Call get_data() method
# Output: 2+3j
num1.get_data()

# Create another ComplexNumber object
# and create a new attribute 'attr'
num2 = ComplexNumber(5)
num2.attr = 10

# Output: (5, 0, 10)
print((num2.real, num2.imag, num2.attr))

# but c1 object doesn't have attribute 'attr'
# AttributeError: 'ComplexNumber' object has no attribute 'attr'
print(num1.attr)
```



# Exemplo: Declarativa Funcional

```
module Main (main) where

numDivs :: Integer -> Integer
numDivs n
    = toInteger $ length [x | x <- [2 .. ((n `quot` 2) + 1)], n `rem` x == 0] + 2

triaList :: [Integer]
triaList = [foldr (+) 0 [1 .. n] | n <- [1 ..]]
triaList2 = go 0 1
    where go cs n = (cs + n) : go (cs + n) (n + 1)

sol :: Integer -> Integer
sol n = head $ filter (\ x -> numDivs (x) > n) triaList2
main = print $ sol 150
```



# Exemplo: Declarativa Lógica

```
(defun encode (lista)
  (if (eql lista nil)
      nil
      (cons (list (length (pega lista)) (car lista)) (encode (tira lista)))
  )
)

(defun pega (lista)
  (cond ((eql lista nil) nil)
        ((eql (cdr lista) nil) lista)
        ((equal (car lista) (cadr lista))
         (cons (car lista) (pega (cdr lista))))
        (t (list (car lista)))
  )
)

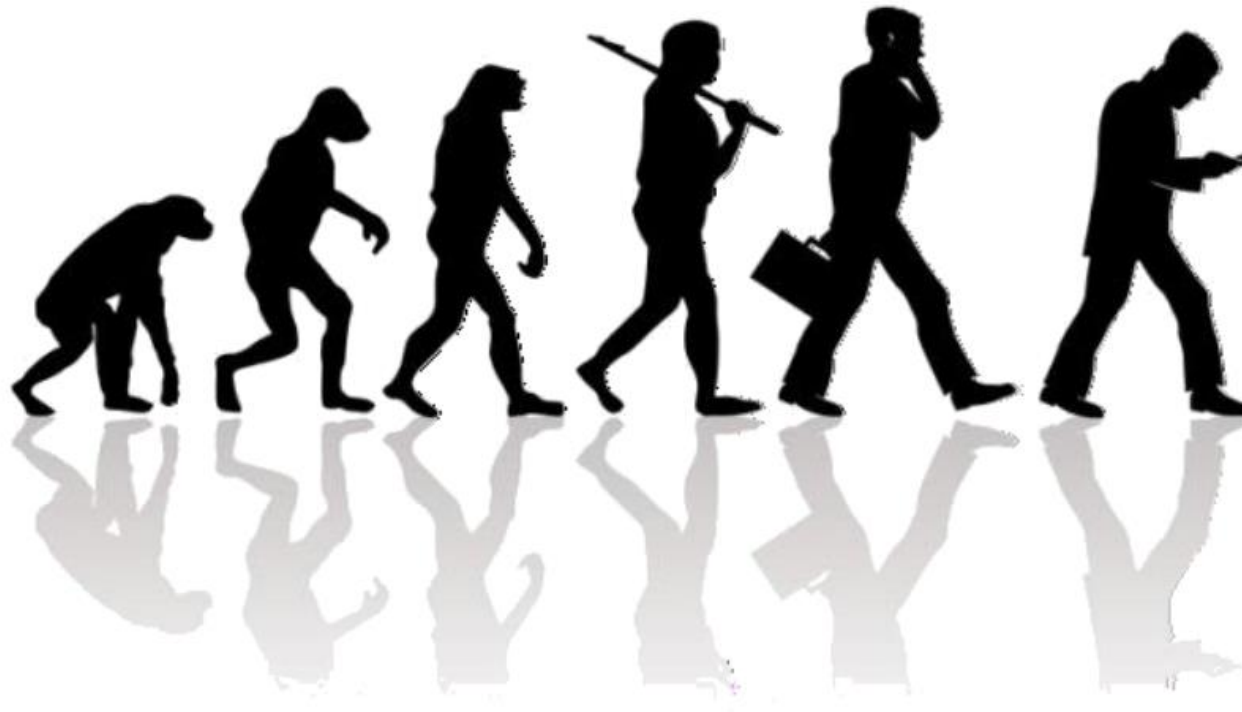
(defun tira (lista)
  (cond ((eql lista nil) nil)
        ((eql (cdr lista) nil) nil)
        ((equal (car lista) (cadr lista))
         (tira (cdr lista)))
        (t (cdr lista))
  )
)
```



# CRITÉRIOS PARA ESCOLHA DE LINGUAGEM

# Critérios de Avaliação

- Será que os critérios mudam com o tempo?





# Critérios de Avaliação



- **No início:**
  - Computadores caros e lentos
  - Aplicativos simples
  - Principal critério: eficiência

- **Posteriormente:**
  - Computadores baratos e rápidos
  - Aplicativos complexos
  - Critérios
    - Eficiência de desenvolvimento
    - Facilidade de manutenção





# Critérios de Avaliação

- Em que podemos traduzir?
  - Eficiência de desenvolvimento
  - Facilidade de manutenção



# Critérios de Avaliação

- Critérios práticos
  - Legibilidade
  - Facilidade de escrita
  - Confiabilidade
  - Custo
  - Portabilidade
- “Pesos” variam caso a caso!



# Critérios de Avaliação

- Legibilidade

- Facilidade de compreensão
- Simplicidade geral
  - Muitas formas de fazer a mesma coisa pode prejudicar

```
contador = contador + 1  
contador += 1  
contador++  
++contador
```

**C++ / Java**

- E simplicidade extrema?
  - Também é problema!  
Ex.: assembly!

- Ortogonalidade

- Poucas primitivas, muitas combinações é bom!

## **Assembly i8080**

```
MOV r, r  
MVI r, n
```

## **Assembly Z80**

```
LD a, b
```

# Critérios de Avaliação

- Legibilidade

- Tipos de dados
  - Adequação torna a leitura mais clara
- Projeto da sintaxe
  - Palavras especiais... Símbolos...

**C / C++ / Java**  
while (x<10) {  
    x = x + 1;  
}

**Pascal / Delphi**  
while x<10 do  
    begin  
        x := x + 1;  
    end

**Pyhton**  
while x<10:  
    x = x + 1

- Palavras especiais... Símbolos...
  - Significados mutantes são ruins
    - static em C/C++

**BASIC**  
logado = 1

**Java / Python**  
logado = **true**

# Critérios de Avaliação

- Facilidade de Escrita
  - Adequação de uma linguagem ao domínio
  - Simplicidade e Ortogonalidade
    - Possibilidade de ser mais sucinto
  - Expressividade
    - Várias maneiras de expressar as coisas

```
C / C++ / Java  
cont = cont + 1;  
cont++;
```

```
C / C++ / Java  
x = 0;  
while (x<10) {  
    x++;  
}
```

```
C / C++ / Java  
for (x=0; x<10;x++) {  
    ...  
}
```

Facilidade de Escrita   x   Facilidade de Leitura

# Critérios de Avaliação

- Confiabilidade
  - Comportamento conforme especificação
  - Legibilidade e facilidade de escrita
    - Certamente evita erros!
  - Verificação de tipos
    - Cadastrar cliente não cadastra um inteiro! (Java, C...)
  - Tratamento de exceções
    - Obrigar a tratar situações de erro (Java, C++, C#, Python)
- Apelidos
  - Perigo: vários nomes para mesmo valor na memória

# Critérios de Avaliação

- Custo
  - Custo de treinamento (simplicidade, ortogonalidade)
  - Custo de escrita (facilidade de escrita)
  - Custo de compilação
  - Custo de execução
  - Custo de implementação (ambiente)
  - Custo da baixa confiabilidade
  - Custo de manutenção (legibilidade e facilidade de escrita).

# Critérios de Avaliação

- Portabilidade
- Capacidade de transportar para outros sistemas
- Padronização da linguagem